Logics for Computer Science: Classical and Non-Classical

Anita Wasilewska

January 9, 2021

For my daughter Agatha
Light of my life

Contents

Chapter 1

Introduction: Paradoxes and Puzzels

1.1 Mathematical Paradoxes

Until recently, till the end of the 19th century, mathematical theories used to be built in an intuitive or axiomatic way. In other words, they were based either intuitive ideas concerning basic notions of the theory - ideas taken from the reality - or on the properties of these notions expressed in systems of axioms. The historical development of mathematics has shown that it is not sufficient to base theories on an intuitive understanding of their notions only. This fact became especially obvious in set theory. The basic concept of this theory, *set*, is certainly taken from reality, for there we come across many examples of various sets, all of which are finite. But in mathematics it is also necessary to consider infinite sets, such as the set of all integers, the set of all rational numbers, the set of all segments, the set of all triangles.

By a set, we mean intuitively, any collection of objects. For example, the set of all even integers or the set of all students in a class. The objects that make up a set are called its members (elements). Sets may themselves be members of sets for example, the set of all sets of integers has sets as its members. Most sets are not members of themselves; the set of all students, for example, is not a member of itself, because the set of all students is not a student. However, there may be sets that do belong to themselves - for example, the set of all sets. However, a simple reasoning indicates that it is necessary to impose some limitations on the concept of a set.

Russell, 1902 Consider the set A of all those sets X such that X is not a member of X. Clearly, by definition, A is a member of A if and only if

A is not a member of A. So, if A is a member of A, the A is also not a member of A; and if A is not a member of A, then A is a member of A. In any case, A is a member of A and A is not a member of A.

This paradox arises because the concept of set was not precisely defined and was too liberally interpreted. Russell noted the self-reference present in his paradox (and other paradoxes, two of them stated below) and suggested that every object must have a definite non-negative integer as its *type*. Then an expression *x is a member of the set y* is *meaningful* if and only if the type of y is one greater than the type of x. So, according to the theory of types, it is meaningless to say that a set belongs to itself, there can not be such set A, as stated in the Russell paradox.

The paradoxes concerning the notion of a set are called *logical paradoxes (antinomies)*. Two of the most known (besides the Russell's) *logical paradoxes* are Cantor and Burali-Forti antinomies. Both were stated at the end of 19th century. The Cantor paradox involves the theory of *cardinal numbers*, Burali-Forti paradox is the analogue to Cantor's in the theory of *ordinal numbers.* They will make real sense only to those already familiar with both of the theories, but we will state them here because they do have an intuitive meaning and had very important consequences.

The cardinal number $cardX$ of a set$X$ intuitively corresponds, in a case of finite sets, to a *number of elements of the set X.* $cardX$ is formally defined to be the set of all sets Y that are *equinumerous* with X (i.e., for which there a one-to-one correspondence between X and Y). We define $cardX \leq cardY$ to mean that X is equinumerous with a subset of Y; by $cardX < cardY$ we mean $cardX \leq cardY$ and $cardX \neq cardY$. Cantor proved that *if $\mathcal{P}(X)$ is the set of all subsets of X, then $cardX < card\mathcal{P}(X)$*. The cardinal numbers behave similarly to natural numbers in many cases, in particular Schröder- Berstein proved that *if $cardX \leq cardY$ and $cardY \leq cardX$, then $cardX = cardY$.*

The *ordinal numbers* are the numbers assigned to sets in a similar way as cardinal numbers but they deal with *ordered* sets.

Cantor, 1899 Let C be the universal set - that is, the set of all sets. Now, $\mathcal{P}(C)$ is a subset of C, so it follows easily that $card\mathcal{P}(C) \leq cardC$. On the other hand, by Cantor theorem, $cardC < card\mathcal{P}(C)$, so also $cardC \leq card\mathcal{P}(C)$ and by Schröder- Berstein theorem we have that $card\mathcal{P}(C) = cardC$, what contradicts $cardC < card\mathcal{P}(C)$.

Burali-Forti, 1897 Given any ordinal number, there is a still larger ordinal number. But the ordinal number determined by the set of all ordinal numbers is the largest ordinal number.

The approach of eliminating logical paradoxes, known as the theory of types, was systematized and developed by Whitehead and Russell in 1910 - 1913. It is successful, but difficult in practice and has certain other drawbacks as well. A different criticism of the logical paradoxes is aimed at their assumption that the notion of a set is defined in such a way that, for every property $P(x)$, there exists a corresponding set of all objects x that satisfy $P(x)$. If we reject this assumption, then the logical paradoxes are no longer derivable. Russell's Paradox then simply proves that there is no set A of all sets that do not belong to themselves; the paradoxes of Cantor and Burali-Forti show that there is no universal set and no set that contains all ordinal numbers.

It became obvious that the paradoxes described above, as well as many similar ones occurring in intuitive set theory cannot be avoided by referring to intuition only. The solution looked for was to characterize the intuitive concept of set by a suitable set of *axioms*. If we do so, we obtain an *axiomatic set theory* without such antinomies.

The first such axiomatic set theory was invented by Zermello in 1908. However, in 1922 Fraenkel pointed out some shortcomings of Zermello's axiomatizations and proposed improvements. The result is called *Zermello-Frankel set theory* ZF, or ZFC, where C stands for axiom of choice. Today ZFC is the standard form axiomatic set theory. ZFC is considered the most common foundation of mathematics sufficient to include all actual mathematical theories.

A more radical interpretation of the paradoxes has been advocated by Brouwer and his *intuitionist* school. They refuse to accept the universality of certain basic logical laws, such as the law of *excluded middle*: A or not A. Such a law, they claim, is true for finite sets, but it is invalid to extend it to all sets. It means that the intuitionists' view of the concept of infinite set differs from that of most mathematicians. Intuitionists reject the idea of infinite set as a closed whole. They look upon an infinite set as something which is constantly in a state of formation. It means that, for example, the set of all positive integers is not looked upon as a closed whole. It is infinite in a sense that to any given finite set of positive integers it is always possible to add one more positive integer. The notion of the set of all subsets of the set of all positive integers is not regarded meaningful. Obviously, intuitionists' view-point on the meaning of the basic logical and set-theoretical concepts used in mathematics is different from that of most mathematicians in their research. The basic difference lies in the interpretation of the word *exists*. For example, let $P(n)$ be a statement in the arithmetic of positive integers. For the mathematicians the sentence *the exists n, such that P(n)* is true if it can be *deduced* (proved) from the axioms of arithmetic by means of classical logic. If the mathematician proves it, this does not mean that he is able to indicate a method of *construction* of a positive integer n such that $P(n)$ holds. On the contrary, for the intuitionist the sentence *the exists n, such that P(n)* is true only if he is able to *construct* a number n such that $P(n)$ is true. In the intuitionists' universe we are justified in asserting the existence of an object having a certain property only if we know an effective method for constructing or finding such an object. The paradoxes are, in this case, not derivable (or even meaningful), but so are many theorems of everyday mathematics, and for this reason, intuitionism has found few converts among mathematicians. But, because of its constructive flavor, it has found some applications in computer science and this is the reason to discuss some of it here. An exact exposition of the basic ideas of intuitionism is outside the range of this book, but we will study *intuitionists logic*, which is a sort of reflection of intuitionists ideas formulated in formalized deductive system.

As we can see, the axiomatic method was the first step leading to greater precision in the construction of mathematical theories. In intuitive mathematical theories the borderline between that which is obvious and that which requires proof is not exact. In axiomatic theories a system of primitive notions is assumed which are characterized by a set of axioms. Other notions can be defined by means of the primitive notions. All statements which are consequences of the axioms are called theorems of the theory. All properties of any notion of the theory which are not expressed in the axioms, require proof.

For some time this degree of exactness in the construction of theories seemed sufficient. However, it turned out that the assumption of a consistent set of axioms does not prevent the occurrence of another kind of paradoxes, called *semantic paradoxes.*

For instance, let us consider the arithmetic based on the well known system of axioms due to Peano (to be discussed in chapter 11) and let's consider the following simple reasoning.

Berry, 1906 Let A denote the set of all positive integers which can be defined in the English language by means of a sentence containing at most 1000 letters. The set A is finite since the set of all sentences containing at most 1000 letters is finite. Hence, there exist positive integers which do not belong to A. The sentence:

n is the least positive integer which cannot be defined by means of a sentence of the English language containing at most 1000 letters

contains less than 1000 letters and defines a positive integer n. Therefore n belongs to A. On the other hand, n does not belong to A by the definition of n. This contradicts the first statement.

It is obvious that the reason for this paradox is that in its construction we used some notions (e.g the English language, letters, sentences) which do not belong to pure arithmetic. Usually we do not introduce definitions like the above in mathematics. The paradox resulted entirely from the fact that we did not say precisely what notions and sentences *belong* to the arithmetic and what notions and sentences *concern* the arithmetic, examined as a fix and closed deductive system. Intuitively we conceive the arithmetic to be a set of sentences expressing certain properties of positive integers and of other notions defined by means of the notion of integer. For example, a sentence: *for every integer n,* $n^2 \leq 0$ certainly belongs to arithmetic.

On the other hand we can also talk *about* the arithmetic. That is, assuming that all sentences in the arithmetic are formulated in the English language, we can formulate sentences concerning the problem how many integers we defined in the arithmetic by means of at most 1000 letters. However, such sentences about the arithmetic do not *belong* to arithmetic. They belong to another theory, which *examines* the arithmetic as a new subject of investigation. This theory is usually called *meta-arithmetic.* In particular, the Berry sentence does not belong to arithmetic; it belongs to meta-arithmetic and the paradox arises from the luck of distinction between a theory (language) and its meta-theory (metalanguage).

For a similar reason in well defined theory the following paradoxes can not appear.

The Liar Paradox (Greek philosopher Eubulides of Miletus, 400 BC)
A man says: *I am lying.* If he is lying, then what he says is true, and so he is not lying. If he is not lying, then what he says is not true, and so he is lying. In any case, he is lying and he is not lying.

Löb, 1955 Let A be any sentence. Let B be a sentence: *If this sentence is true, then A.* So, B asserts: *If B is true then A.* Now consider the following argument: Assume B is true. Then, by B, since B is true, A is true. This argument shows that, if B is true, then A. But this is exactly what B asserts. Hence, B is true. Therefore, by B, since B is true, A is true. Thus every sentence is true.

In these cases the paradox arises because the concepts of " I am true", " this sentence is true, " I am lying"" should not occur in the language (theory). It belong to a metalanguage (meta-theory).

The Liar Paradox is a corrected version of a following paradox stated in antiquity by a Cretan philosopher Epimenides, 600 BC.

Cretan " Paradox" (The Cretan philosopher Epimenides paradox, 600 BC)

Epimenides, a Cretan said: *All Cretans are liars.* If what he said is true, then, since Epimenides is a Cretan, it must be false. Hence, what he said is false. Thus, there must be some Cretan who is not a liar.

Note that the conclusion that there must be some Cretan who is not a liar is not logically impossible, so we do not have a genuine paradox. However, the fact that the utterance by Epimenides of the false sentence could imply the existence of some Cretan who is not a liar is rather unsettling.

It follows from above *semantic paradoxes* that in order to exclude them from an axiomatic theory, it is necessary to describe its *language* precisely, i.e. the set of sentences of the theory and the set of signs used to build these sentences. In this way we avoid contradictions caused by a collision between the theory and meta-theory, that is, by including meta-theoretical statements in the theory. This inclines us to introduce still greater precision in the construction of mathematical (and others) theories and leads to the concept of *formalized theories*, in which not only the properties of primitive notions are given in an exact way by means of a set of *axioms*, but also the *language of the theory* is precisely defined. The formalization of the language also gives the following advantage: it permits us to describe precisely the logical means assumed in the theory, i.e. to define the process of *deduction* precisely.

In formalized mathematical theories, e.g. in formalized arithmetic and formalized set theory, the paradoxes as those mentioned above cannot be constructed. On the other hand, a mathematician (or a computer scientist) following good intuitions in every -day investigations does not reach a contradiction even though the language of the theory and the logical means he employs are not precisely described. This is explained by the fact that his investigations can always in practice be repeated in a corresponding formalized theory. Thus he avoids practical difficulties of formalized theories, the formalized language of of which is complicated and very inconvenient in every day practice. Consequently, in mathematical practice we build theories axiomatically but always in such a way that they can be easily formalized, i.e. that all the reasonings can be repeated in a corresponding formalized theory. However, the formalization of the language and the logical means are necessary if we intend to develop the meta-theory of a given practical theory because only in this way such important notions as the existence of a proof of a given statement or the set of all theorems of the theory can be *precisely* defined. In practical, non-formalized axiomatic theories those notions are far from this precision.

Whatever approach one takes to the paradoxes, it is always necessary first to examine the language of logic and mathematics or given domain of computer science, to see what *symbols* may be used, to determine the way ways in which these symbols are put together to form terms, formulas, sentences, and proofs, and to find out what can and cannot be proved if certain axioms and rules of inference are assumed.

This is the basic task of mathematical logic, and, until it is done, there is no basis of talking about foundations of logic, mathematics or computer science.

This approach is already almost a century old - the first formalized theory was built by Frege in 1891. The deep and devastating results of Gödel, Tarski, Church, Rosser, Kleene and many others followed. They created what is called a modern mathematical logic and have earned for it its status as an independent branch of mathematics.

1.2 Computer Science Puzzles

Logical and semantical paradoxes have lead the mathematicians to the development of a modern classical logic as an independent domain of mathematics. They have also, as we could see, led to the development of the intuitionistic logic as rival to the classical one. The classical and intuitionistic logic differ on the interpretation of the meaning of the word *exists*, but also, as we will see later, in the interpretation of logical implication, i.e. the truthfulness of the sentences of the form *if A then B* is decided differently in both logics.

In 1918, an American philosopher, C.I. Lewis proposed yet another interpretation of lasting consequences, of the logical implication. In an attempt to avoid, what some felt, the paradoxes of implication (a false sentence implies any sentence) he created a *modal logic*. The idea was to distinguish two sorts of truth: *necessary* truth and mere *possible (contingent)* truth. A *possibly* true sentence is one which, though true, could be false.

More recently, modal logic has become much-used in computer science as a tool for analyzing such notions as knowledge, belief, tense.

The logics other than the classical propositional or predicate logic are usually called *non-standard logics*. The use of classical logic in computer science is known, indisputable, and well established.The existence of PROLOG and Logic Programming as a separate field of computer science is the best example of it. But the non-standard logics have been imported into a great many areas of computer science and, in particular into the research about the specification and verification of programs, the analysis of behavior of distributed systems and into almost all areas of artificial intelligence. Even in Logic Programming, once we start to talk *about* logic programming programs we run immediately into some non-standard logics.

Modal logic, for example, has been employed in form of *Dynamic logic* (Harel 1979) to facilitate the statement and proof of properties of programs.

Temporal Logics were created for the specification and verification of concurrent programs Harel, Parikh, 1979, 1983), for a specification of hardware circuits Halpern, Manna and Maszkowski, (1983), to specify and clarify the concept of causation and its role in commonsense reasoning (Shoham, 1988).

Intuitionistic logic, in the form of Martin-Löf's theory of types (1982), provides a complete theory of the process of program specification, construction, and verification. A similar theme has been developed by Constable (1971) and Beeson (1983).

The great strength of dynamic and temporal logics relates to their expressive power. In such systems it is possible to express properties of programs in an elegant and natural way. This is in large part due to enriched language of such logics over the classical predicate calculus. In the case of intuitionistic logic the motivation for their employment, as it was mentioned before, is different. The proponents of intuitionistic logic and mathematics claim that *constructive* mathematics is, generally, a more appropriate framework for computer science than classical logic and mathematics.

Fuzzy logic, Many valued logics were created and developed to reasoning with incomplete information. Most expert systems are forced to take decisions when not all the facts pertaining to decision are available. In such context it is natural to employ logics which, unlike classical logic, are suited to reasoning with such incomplete information.

The development of different logics and the applications of logic to different areas of computer science or even artificial intelligence only is beyond the scope of our investigations.

We present some of the most known motivations (*computer science puzzles*), which played a similar role in the development of the reasoning about knowledge in *distributed systems* and *artificial intelligence*, as logical and semantical paradoxes played in the development of logic and foundations of mathematics.

1.2.1 Reasoning about knowledge in distributed systems

The major complexities in designing, understanding and reasoning about distributed systems arise from the uncertainties inherent in the system, particularly with regard to message delivery and possible faulty or unexpected behavior of processors. A protocol must be designed (and proved) to function properly even if it is possible for messages to be lost, for messages to arrive out of order, or for some processor to fail. It is difficult (probably impossible) for one node to *know* everything about the rest of the network. Yet we are often interested in situations in which everyone in a group (every processor in the network) knows a certain fact. But even the state of knowledge in which everyone knows what everyone knows does not suffice for a number of applications. In some cases we also need to consider the state in which simultaneously everyone knows a fact F, everyone knows that everyone knows F, everyone knows that everyone knows that everyone knows F, and so on. In this case it is said that the group has *common knowledge* of F. The relationship between the common knowledge, simultaneous agreement and coordinate action is nicely put together in the *coordinated attack problem*, from the distributed system folklore.

Grey, 1978. Halpern, Moses, 1984 Two divisions of an army are camped on two hilltops overlooking a common valley. In the valley awaits the enemy. It is clear that if both divisions attack the enemy simultaneously they will win the battle, whereas if only one division attacks it will be defeated. The divisions do not initially have plans for launching an attack on the enemy, and the commanding general of the first division wishes to coordinate a simultaneous attack (at some time the next day). Neither

> general will decide to attack unless he is sure that the other will attack with him. The generals can only communicate by means of a messenger. Normally, it takes a messenger one hour to get from one encampment to the other. However, it is possible that he will get lost in the dark or, worst yet, be captured by the enemy. Fortunately, on this particular night, everything goes smoothly. How long will it take them to coordinate an attack?

Suppose the messenger sent by General A makes it to General B with a message saying *Attack at dawn.* Will B attack? No, since A does not know B got the message, and thus may not attack. So B sends the messenger back with an acknowledgment. Suppose the messenger makes it. Will A attack? No, because now A is worried that B does not know A got the message, so that B thinks A may think that B did not get the original message, and thus not attack. So A sends the messenger back with an acknowledgment. But of course, this is not enough either. It is not difficult to be convinced that no amount of acknowledgments sent back and forth will ever guarantee agreement. Even in a case that the messenger succeeds in delivering the message every time. All that is required in this (informal) reasoning is the possibility that the messenger doesn't succeed.

This rather convoluted reasoning was expressed formally by Halpern and Moses in 1985 in terms of *a propositional modal logic with m agents*. They proved this logic to be essentially a multi-agent version of the *modal logic S5* which we present in chapter 7.

They also showed that not only is common knowledge (formally defined!) not attainable in systems where communication is not guaranteed, it is also not attainable in systems where communication *is* guaranteed, as long as there is some uncertainty in message delivery time. Thus, in practical distributed systems , common knowledge is not attainable. This holds for systems of communicating humans as well as processors. What is going on here? After all, we often do reach agreement! It was shown that common knowledge (as formally defined) is attainable in such models of reality where we assume, for example, events can be guaranteed to happen simultaneously. It turns also out that even we can't always make this assumption in practice, there are some variants of the definition of common knowledge that are attainable under more reasonable assumptions, and these variants are indistinguishable in certain cases from the "true" common knowledge, as originally defined.. So, finally, we can *prove* that in fact we often do reach agreement!

1.2.2 Reasoning in Artificial Intelligence

A key property of intelligence, one may agree, is *flexibility.* This flexibility is intimately connected with the defeasible nature of commonsense inference; we are all capable of drawing conclusions, acting on them, and then retracting them if necessary in the face of new evidence. If our computer programs are to act intelligently, they will need to be similarly flexible.

A large portion of the work in artificial intelligence (AI) on reasoning or deduction involves the development of formal systems that describe this process.

The most usual example of such a flexible inference is the following flying birds example.

Reiter, 1987 Consider a statement *Birds fly.* Tweety, we are told, is a bird. From this, and the fact that birds fly, we conclude that Tweety can fly.

This conclusion, however is *defeasible*: Tweety may be an ostrich, a penguin, a bird with a broken wing, or a bird whose feet have been set in concrete.

The inference here is *non-monotonic*: on learning a new fact (that Tweety has a broken wing), you are forced to retract your conclusion that he could fly. This original conclusion didn't follow logically (in a sense if classical logic) from the facts that birds typically fly and that Tweety is a bird; it followed from these facts *together* with the *assumption* that Tweety is a typical bird. When we learn more about Tweety, we may discover that this assumption is unjustified.

It means, by a *non-monotonic reasoning* (logics) we understand reasonings (logics) in which the introduction of a new information (facts) can invalidate old theorems.

The inference described here is also called a *default reasoning.*

It means, by *default reasoning* we mean the drawing of plausible inferences from less-then-conclusive evidence in the absence of information to the contrary.

Consider now the following example.

Moore, 1983 Consider my reason for believing that I do not have an older brother. It is surely not that one of my parents once casually remarked, *You know, you don't have any older brothers*, nor have I pieced it together by carefully sifting other evidence. I simply *believe* that if I did have an older brother I would know about it; therefore, since I don't know of any older brothers of mine, I must not have any.

This type of reasoning is not a form of default reasoning at all; it rather seems to be more like reasoning about one's own knowledge or belief. Hence it is called an *auto-epistemic reasoning*, i.e. the reasoning about one's own beliefs or knowledge.

The auto-epistemic reasoning is intended to model the reasoning of an ideally rational agent reflecting upon his beliefs or knowledge. Logics which describe it are called *auto-epistemic logics.*

In addition to application to the understanding of common-sense reasoning, non-monotonic reasoning (*non-monotonic logics*) has been shown to be important in other areas. There are applications to logic programming, to planning and reasoning about action, and to automated diagnosis. As the formal work matures, increasing effort is being devoted to applying the improved understanding to the solution of practical problems.

We end this introduction by McCarthy discussion of a much used in AI puzzle *Missionaries and Cannibals*, as a proof of a need of another "stretch " from classical logic.

McCarthy, 1985 Consider the *Missionaries and Cannibals* puzzle.

> *Three missionaries and three cannibals come to the river. A rowboat that seats two is available. If the cannibals ever outnumber the missionaries on either bank of the river, the missionaries will be eaten. How shall they cross the river?*

Obviously the puzzler is expected to devise a strategy of rowing the boat back and forth that gets them all across and avoids the disaster.

Ammarel considered several representations of the problem and discussed criteria whereby the following representation is preferred for purposes of AI, because it leads to the smallest state space that must be explored to find the solution. A state is a triple comprising the number of missionaries, cannibals and boats on the starting bank of the river. The initial state is 331, the desired state is 000, and one solution is given by the sequence: 331, 220, 321, 300,311, 110, 221, 020, 031, 010, 021, 000.

We are not presently concerned with the heuristic of the problem but rather with the *correctness of the reasoning* that goes from the English statement of the problem to Amerel's state space representation. A generally intelligent computer program should be able to carry out this reasoning. Of course, there are the well known difficulties in making computers understand English, but suppose the English sentences describing the problem have already been rather directly translated into first order logic. The correctness of Amarel's representation is not an ordinary logical consequence of these sentences for two further reasons. First, nothing has been said about the properties of boats or even the fact that rowing across the river doesn't change the number of missionaries or cannibals or the capacity of the boat. Indeed it hasn't been stated that situations change as a result of action. These facts follow from common sense knowledge, so let us imagine that common sense knowledge, or at least the relevant part of it, is also expressed in first order logic.

The second reason we can't *deduce* the propriety of Amarel's representation is deeper. Imagine giving someone a problem, and after he puzzles for a while, he suggests going upstream half a mile and crossing a bridge.

> *What bridge?* you say. *No bridge is mentioned in the statement of the problem.* And this dunce replies, *Well, they don't say there isn't a bridge.* You look at the English and even at the translation of the English into first order logic, and you must admit that *they don't say* there is no bridge. So you modify the problem to exclude the bridges and pose it again, and the dunce proposes a helicopter, and after you exclude that, he proposes a winged horse or that the others hang onto the outside of the boat while two row.
>
> You now see that while a dunce, he is an inventive dunce. Despairing of getting him to accept the problem in the proper puzzler's spirit, you tel him the solution. To your further annoyance, he attacks your solution on the grounds that the boat might have a leak or lack oars. After you rectify that omission from the statement of the problem, he suggests that a sea monster may swim up the river and may swallow the boat. Again you are frustrated, and you look for a mode of reasoning that will settle his hash once and for all.

McCarthy proposes *circumscription* as a technique for solving his puzzle. He argues that it is a part of common knowledge that a boat can be used to cross the river *unless there is something with it or something else prevents using it*, and if our facts do not require that there be something that prevents crossing the river, *circumscription* will generate the conjecture that there isn't. Among the various competing approaches to model the common sense reasoning *circumscription* appears to have the most advocates.

One of the serious difficulties is that the circumscription axiom itself involves a quantification over predicates, and there is therefore a sentence in *second-order logic.* Little is known about automated deduction using second-order sentences, but on the other hand Lifschits has shown in 1987 that in some special cases the circumscription is equivalent to a first order sentence. In this way we can go back, in those cases, to our secure and well known classical logic.

1.3 Homework Problems

1. Write definition of *logical* and *semantical* paradox.
2. Give an example of a *logical paradox.*
3. Give an example of a *semantical paradox.*
4. Describe a difference between *logical* and *semantical* paradoxes.
5. Describe a role of paradoxes in the development of classical logic and foundations of mathematics.
6. Write a definition os a *non-standard logic.*

7. Give an example of some non-standard logics.
8. Describe a difference between classical and intuitionistic logic.
9. Give two examples of Computer Science Puzzles.
10. What a *default reasoning* is? Give an example.
11. What a *non - monotonic reasoning* is? Give an example.
12. What an *auto-epistemic reasoning* is? Give an example.

Chapter 2

Introduction to Classical Logic

.

Logic builds symbolic models of our world. It builds them in such a way as to be able to describe formally the ways we reason in and about it. It also poses questions about correctness of such models and develops tools to answer them. Classical Logic was created to describe the reasoning principles of mathematics and hence reflects the "black" and "white" qualities of mathematics; we expect from mathematical theorems to be always either true or false and the reasonings leading to them should guarantee this without any ambiguity. It hence admits only two logical values and is sometimes called a two-valued logic.

The models we build are based on a principle that the language in which we reason uses *sentences*. These sentences are built up from *basic assertions* about the world using special words or phrases like "not", "not true" "and", "or", " implies", "if ..... then", "from the fact that .... we can deduce", " if and only if", "equivalent", "every", "for all", "any", "some"," exists". Basically, it is the behavior of these words we want to study. Most of these words and phrases have accepted intuitive meanings and we want our models to formalize these meanings. To do so we first define a notion of a *symbolic language* and then define a formal meaning of its symbols, called *semantics.*

We use *symbols*: $\neg$, for "not", "not true", $\cap$ for "and", $\cup$ for "or", $\Rightarrow$ for " implies" , "if ..... then", "from the fact that... we can deduce", and a symbol $\Leftrightarrow$ for " if and only if", "equivalent". We call these symbols *propositional connectives.* There are other symbols for propositional connectives and there are other propositional connectives as well that we will introduce later.

We use *symbols*: $a, b, c, p, r, q, \ldots,$ with indices, if necessary to represent the basic assertions, called *propositions.* Hence we call the symbols $a, b, c, p, r, q, \ldots$ *propositional variables.*

We use *symbols*: $\forall$ for "every", "any", and $\exists$ for "some" ," exists". The symbols $\forall, \exists$ are called *quantifiers.*

Restricting our attention to the role of propositional connectives yields to what is called *propositional logic* with the a *propositional language* and a *propositional semantics* as its basic components. This is a quite simple model to justify, describe and develop and we will devote first few chapters to it. We do it both for its own sake, and because it provides a good background for developing and understanding more difficult logics to follow.

Consideration and study of the role of propositional connectives and quantifiers leads to what is called a *predicate logic* with its *predicate language* and *semantics.* This is a much more complicated model and we will develop and study it in full formality in chapters following the introduction and examination of the formal propositional logic model.

In this chapter we provide motivation for and description of both propositional and predicate languages, and discuss their semantics.

2.1 Propositional Language: Motivation and Description

The propositional language is a quite simple symbolic language into which we can translate (represent) natural language sentences. For example, let's consider a natural language sentence " If 2+2 = 5, then 2+2 = 4". To translate it into the propositional language we replace "$2 + 2 = 5$" by a propositional variable, let's say a, and "$2 + 2 = 4$" by a propositional variable b and we write a connective $\Rightarrow$ for "if ..... then". As a result we obtain a propositional language *formula* $(a \Rightarrow b)$. A sentence "If $2 + 2 \neq 4$ and $2 + 2 = 5$, then $2 + 2 = 4$" translates into a formula $((\neg b \cap a) \Rightarrow b)$. A sentence " fact that it is not true that at the same time $2 + 2 = 4$ and $2 + 2 = 5$ implies that $2 + 2 = 4$" translates into a propositional formula $(\neg(b \cap a) \Rightarrow b)$.

A formal description of symbols and the definition of the set of formulas is called a *syntax* of a symbolic language. We use the word syntax to stress that the formulas carry neither formal meaning nor a logical value. We assign the meaning and logical value to syntactically defined formulas in a separate step. This next, separate step is called a *semantics.* A given symbolic language can have different semantics and different semantics can define different logics.

We first describe the syntax of the propositional language. The syntax of the predicate language is much more complex and will be defined later.

The smallest "building blocks" of a propositional language are propositional variables that represent the the basic assertions called *propositions.* Historically, we define propositions as basic, declarative sentences (assertions) that can always be evaluated as *true* or *false.* For example, a statement: " $2 + 2 = 4$" is a proposition as we assume that it is a well known and agreed upon truth. A statement: "$2 + 2 = 5$" is also a classical proposition (false). A statement: $2 + n = 5$ according to the historical definition is not a proposition; it might be true for some n, for example n=3, false for other n, for example n= 2, and moreover, we don't know what n is. Sentences of this kind are called *propositional functions.* We treat propositional functions within propositional model as propositions and represent them by the propositional variables.

Similar examples can be found in natural language rather then in mathematical language. For example we tend to accept a statement: "The earth circulates the sun" as a proposition while a statement: "Ann is pretty", even if we accept it as a proposition by assuming that is always has exactly one logical value, could also be treated as ambiguous; Ann may be found pretty by some people and not pretty by others. If we try to improve the situation by saying for example: "Ann seems to be pretty", " I am sure Ann is pretty" or even "I know that Ann is pretty" the ambiguity increases rather then decreases.

To deal with these and other ambiguities many *non-classical logics* were and are being invented and examined by philosophers, computer scientists, and even by mathematicians. We will present and study some of them later. Nevertheless we accept all these and similar statements within classical propositional model as propositions and represent them by the propositional variables.

Observe that one can think about a *natural language* as a set $\mathcal{W}$ of all words and sentences based on a given alphabet $\mathcal{A}$. This leads to a simple, abstract model of a *natural language* NL as a pair

$$NL = (\mathcal{A},\ \mathcal{W}).$$

Some natural languages share the same alphabet, some have different alphabets. All of them face serious problems with a proper recognition and definitions of accepted words and complex sentences. We do not want the symbolic languages to share the same difficulties. We define their components precisely and in such a way that their recognition and correctness will be easily decided. In order to distinguish them from natural languages we call their words and sentences *formulas* and denote the set of all *formulas* by $\mathcal{F}$. We call a pair

$$SL = (\mathcal{A},\ \mathcal{F}). \tag{2.1}$$

a *symbolic language.*

We distinguish two categories of symbolic languages: *propositional* and *predicate.* We first define the *propositional language.* The definition of the *predicate language*, with its much more complicated structure will follow.

Definition 2.1

By a **propositional language** $\mathcal{L}$ *we understand a pair*

$$\mathcal{L} = (\mathcal{A}, \mathcal{F}), \tag{2.2}$$

where $\mathcal{A}$ is called propositional **alphabet***, and $\mathcal{F}$ is called a set of all well formed propositional* **formulas** *of $\mathcal{L}$.*

Components the language $\mathcal{L}$ are defined as follows.

1. Alphabet $\mathcal{A}$

The alphabet $\mathcal{A}$ consists of a countably infinite set VAR of propositional variables, a finite set of propositional connectives, and a set of two parenthesis.

We denote the propositional variables by letters *a, b, c, p, q, r, ...,* with indices if necessary. It means that we can also use $a_1, a_2, ..., b_1, b_2,$ ... etc... as symbols for propositional variables.

Propositional connectives are: $\neg$, $\cap$, $\cup$, $\Rightarrow$, and $\Leftrightarrow$. The connectives have well established names. We use names *negation, conjunction, disjunction, implication* and *equivalence* or *biconditional* for the connectives $\neg$, $\cap$, $\cup$, $\Rightarrow$, and $\Leftrightarrow$, respectively. Parenthesis are $($, $)$.

2. Set $\mathcal{F}$ of formulas

Formulas are expressions build by means of elements of the alphabet $\mathcal{A}$. We denote formulas by capital letters $A, B, C, ...$, with indices, if necessary.

The set $\mathcal{F}$ of all formulas of $\mathcal{L}$ is defined recursively as follows.

1. *Base step:* all propositional variables are formulas. They are called **atomic formulas**.

2. *Recursive step:* for any already defined formulas A, B, the expression: $\neg A$, $(A \cap B)$, $(A \cup B), (A \Rightarrow B), (A \Leftrightarrow B)$ are also formulas.

3. Only those expressions are formulas that are determined to be so by means of conditions 1. and 2.

We often say that the set $\mathcal{F}$ is the set of all **well-formed formulas (wff)** to stress exactness of the definition.

By the definition, any propositional variable is a formula. Let's take, for example two variables a and b. They are atomic formulas.

By the recursive step we get that

$$(a \cap b), (a \cup b), (a \Rightarrow b), (a \Leftrightarrow b), \neg a, \neg b$$

are formulas. Recursive step applied again produces for example formulas

$$\neg(a \cap b), ((a \Leftrightarrow b) \cup \neg b), \neg\neg a, \neg\neg(a \cap b).$$

These are not all formulas we can obtain in the second recursive step. Moreover, as the recursive process continue we obtain a countably infinite set of all non-atomic formulas.

Remark that we put parenthesis within the formulas in a way to avoid *ambiguity.* The expression $a \cap b \cup a$ is ambiguous. We don't know whether it represents a formula $(a \cap b) \cup a$, or a formula $a \cap (b \cup a)$.

Exercise 2.1

Consider a following set

$$\mathcal{S} = \{\neg a \Rightarrow (a \cup b),\ ((\neg a) \Rightarrow (a \cup b)),\ \neg(a \Rightarrow (a \cup b)), \neg(a \rightarrow a)\}.$$

1. Determine which of the elements of $\mathcal{S}$ are, and which are not well formed formulas (wff) of $\mathcal{L} = (\mathcal{A}, \mathcal{F})$.

2. For any $A \notin \mathcal{F}$ re-write it as a correct formula and write in the natural language what it says.

Solution
The formula $\neg a \Rightarrow (a \cup b)$ is not a well formed formula. A correct formula is $(\neg a \Rightarrow (a \cup b))$. The corrected formula says: "If a is not true , then we have a or b ". Another correct formula is $\neg(a \Rightarrow (a \cup b))$. This corrected formula says: "It is not true that a implies a or b ".

The formula $((\neg a) \Rightarrow (a \cup b))$ is not correct; $(\neg a) \notin \mathcal{F}$. The correct formula is $(\neg \Rightarrow (a \cup b))$. The formula $\neg(a \Rightarrow (a \cup b))$ is correct. The formula $\neg(a \rightarrow a) \notin \mathcal{F}$ as the connective $\rightarrow$ does not belong to the language $\mathcal{L}$. It is a correct formula of another propositional language; the one that uses a symbol $\rightarrow$ for implication.

Exercise 2.2

Given a sentence S

"If a natural number a is divisible by 3, then from the fact that a in not divisible by three we can deduce that a is divisible by 5."

Write a formula corresponding to the sentence S.

Solution
First we write our sentence in a more "logical way" as follows:

" *If* a natural number a is divisible by 3, *then* (*if not*(a is divisible by three) *then* a is divisible by 5). We denote the sentence: "a natural number a is divisible by 3 " by a, and the sentence "a is divisible by 5" by b, and we rewrite our sentence as: "*If* a, *then* (*if not* a, *then* b)".

We replace expressions *If ... then* and *not* by symbols $\Rightarrow$ and $\neg$, respectively and we follow the definition of the set of formulas to obtain a formula

$$(a \Rightarrow (\neg a \Rightarrow b))$$

which corresponds to our natural language sentence S.

Observe that for a given logical sentence there is only one schema of a logical formula corresponding to it. One can replace a by d and b by c and get a formula $(d \Rightarrow (\neg d \Rightarrow c))$, or we get a formula $(b \Rightarrow (\neg b \Rightarrow a))$ by replacing a by b and b by a. We can, in fact, construct as many of those formulas as we wish, but all those formulas will have the same form as the formula $(a \Rightarrow (\neg a \Rightarrow b))$. They will differ only on a choice of names for the propositional variables assigned corresponding to logical sentences. The same happens, when we want to do the "inverse" transformation from a given formula A to a logical sentence corresponding to it. There may be as many of them as we can invent, but they all will be built in the same way; the way described by the formula A.

Exercise 2.3

Write following natural language statement:

"One likes to play bridge or from the fact that the weather is good we conclude the following: one does not like to play bridge or one likes swimming."

as a formula of $\mathcal{L} = (\mathcal{A}, \mathcal{F})$.

Solution
First we identify the needed components of the alphabet $\mathcal{A}$ as follows.

Propositional variables: a, b, c.

a denotes statement: *One likes to play bridge*, b denotes a statement: *the weather is good*, c denotes a statement: *one likes swimming.*

Connectives: $\cup, \Rightarrow, \cup$.

Then we write the **formula** of $\mathcal{L}$ as $(a \cup (b \Rightarrow (\neg a \cup c)))$.

Exercise 2.4

Given a formula $(a \cap (\neg a \cup b))$.

Write 2 natural language sentences which correspond to this formula.

Solution
Let propositional variables a, b denote sentences 2+2 = 4 and $2 > 1$, respectively. In this case the corresponding sentence is:

$2 + 2 = 4$ *and we have that* $2 + 2 \neq 4$ *or* $2 > 1$.

If we assume that the propositional variables a, b denote sentences $2 > 1$ and $2 + 2 = 4$, respectively, then the corresponding natural language statement is:

$2 > 1$ *and we have that* $2 \not> 1$ *or* $2 + 2 = 4$.

Symbols for Connectives

The symbols for connectives used in book are not the only used in mathematical, logical, or computer science literature.

Other symbols employed for these most important propositional connectives are listed in the table below.

Negation	Disjunction	Conjunction	Implication	Equivalence
$-$A	$(A \cup B)$	$(A \cap B)$	$(A \Rightarrow B)$	$(A \Leftrightarrow B)$
NA	DAB	CAB	IAB	EAB
$\overline{A}$	$(A \vee B)$	(A & B)	$(A \to B)$	$(A \leftrightarrow B)$
$\sim A$	$(A \vee B)$	$(A \cdot B)$	$(A \supset B)$	$(A \equiv B)$
A'	$(A + B)$	$(A \cdot B)$	$(A \to B)$	$(A \equiv B)$

The first of these systems of notation is the closest to ours and is drawn mainly from the algebra of sets and lattice theory. The second comes from the Polish logician *J. Łukasiewicz.* In this notation the binary connectives *precede* the formulas and are *not inserted* between them; this enables us to dispense with parenthesis; Łukasiewicz's notation is usually called the *Polish notation* and it is a *parenthesis-free notation.* The third was used by D. Hilbert. The fourth comes from Peano and Russell, while the fifth goes back to Schröder and Pierce.

2.2 Propositional Semantics: Motivation and Description

We present here definitions of propositional connectives in terms of two logical values *true* or *false* and discuss their motivations.

The resulting definitions are called *a semantics* for the classical propositional connectives. As we consider only two logical values, the semantics is also called 2 valued semantics. The semantics presented here is fairly informal. The formal definition of classical propositional semantics will be presented in chapter 4.

Classical Connectives

Our language $\mathcal{L}$ contains five connectives called conjunction, disjunction, implication, equivalence, and negation. We divide the connectives into two groups: one and two argument connectives. Negation is the one argument connective. Conjunction, disjunction, implication, equivalence are two argument connectives. We define their semantics, i.e. their definitions in terms of two logical values, and give a motivation justifying the definitions as follows.

We denote a statement A *is false* by $A = F$, what stands for *the logical value* of a formula A is F. We denote a statement A *is true* by $A = T$, what stands for *the logical value* of a formula A is T.

Negation motivation and definition.

In accordance with the intuition, the negation of a true formula is a false formula, and the negation of a false formula is a true formula. Moreover, the logical value of $\neg A$ depends on the logical values of A in a way which can be express in the form of the following table.

Negation Table

A	$\neg A$
T	F
F	T

(2.3)

Conjuncion motivation and definition.

In accordance with intuition, a conjunction $(A \cap B)$ is a *true* formula if both of its factors are *true* formulas. If one of the factors, or both, are *false* formulas, then the conjunction is a *false* formula.

The logical value of a conjunction depends on the logical values of its factors in a way which can be express in the form of the following table.

Conjunction Table

A	B	$(A \cap B)$
T	T	T
T	F	F
F	T	F
F	F	F

(2.4)

Disjunction motivation and definition.

In everyday language the word *or* is used in two different senses. In the first, a statement of the form A *or* B is accepted as true if at least one of the statements A and B is true; in the other, the compound statement A *or* B is accepted as true if one of the statements A and B is true, and the other is false. In mathematics the word *or* is used in the former sense.

Hence, we adopt the convention that a *disjunction* $(A \cup B)$ is *true* if at least one of the formulas A and B is true. This convention is called a classical semantics for the disjunction and is expressed in the following table.

Disjunction Table

A	B	$(A \cup B)$
T	T	T
T	F	T
F	T	T
F	F	F

(2.5)

As in the case of the other connectives, the logical value of a disjunction depends only on the logical values of its factors.

Implication motivation and definition.

The symbol $\Rightarrow$ is used instead of the statements of the form *if A, then B*, *A implies B*, and is called an **implication** connective. The formula $(A \Rightarrow B)$ and is called an *implication* and A is called its *antecedent*, B is called its *consequent.*

The semantics of the implication needs some discussion. In everyday language the implication statement *if A, then B* is interpreted to mean that B can be *inferred* from A. This interpretation *differs* from that given to it in mathematics, and hence in classical semantics. The following example explains the meaning of the statement *if A, then B* as understood in mathematics. It hence justifies our semantics for the implication.

Consider the following **arithmetical theorem:**

For every natural number n,

$$if \quad 6 \quad DIVIDES \quad n, \quad then \quad 3 \quad DIVIDES \quad n. \tag{2.6}$$

The above implication (2.6) is **true for any natural number**, hence, in particular, for 2,3,6.

Thus the following propositions are **true**:

$$If \quad 6 \quad DIVIDES \quad 2, \quad then \quad 3 \quad DIVIDES \quad 2. \tag{2.7}$$

$$If \quad 6 \quad DIVIDES \quad 3, \quad then \quad 3 \quad DIVIDES \quad 3. \tag{2.8}$$

$$If \quad 6 \quad DIVIDES \quad 6, \quad then \quad 3 \quad DIVIDES \quad 6. \tag{2.9}$$

It follows from (2.7) that an implication $(A \Rightarrow B)$ in which both the *antecedent* A and the *consequent* B are *false* statements is interpreted as a **true** statement.

It follows from (2.8) that an implication $(A \Rightarrow B)$ in which *false antecedent* A and *true consequent* B is interpreted as a **true** statement.

Finally, it follows from (2.9) that an implication $(A \Rightarrow B)$ in which both the *antecedent* A and the *consequent* B are *true* statements is interpreted as a **true** statement.

Thus one case remains to be examined, namely that in which the *antecedent* of an implication is a *true* statement, and the *consequent* is a *false* statement.

For example consider the statement:

If 6 DIVIDES 12, then 6 DIVIDES 5.

In accordance with arithmetic of natural numbers, this statement is interpreted as **false**.

The above examples justifies adopting the following semantics for the implication $\Rightarrow$. An implication $(A \Rightarrow B)$ is interpreted to be a *false* statement if and only if its *antecedent* A is a *true* statement and its *consequent* is a *false* statement. In the remaining cases such an implication is interpreted as a *true* statement.

We expressed it in a form of the following table.

Implication Table

A	B	$(A \Rightarrow B)$
T	T	T
T	F	F
F	T	T
F	F	T

(2.10)

Equivalence motivation and definition.

An equivalence $(A \Leftrightarrow B)$ is, in accordance with intuition, interpreted as *true* if both formulas A and B have the same logical value, that is, are either *both true* or *both false.* This is expressed in the following table.

Equivalence Table

A	B	$(A \Leftrightarrow B)$
T	T	T
T	F	F
F	T	F
F	F	T

(2.11)

We summarize the tables for propositional connectives in the following one table. We call it a **truth table definition** of propositional; connectives and hence we call the semantics defined here a **truth tables semantics**.

A	B	$\neg A$	$(A \cap B)$	$(A \cup B)$	$(A \Rightarrow B)$	$(A \Leftrightarrow B)$
T	T	F	T	T	T	T
T	F	F	F	T	F	F
F	T	T	F	T	T	F
F	F	T	F	F	T	T

(2.12)

The table (2.12) indicates that the logical value of of propositional connectives depends only on the logical values of its factors; i.e. it is *independent* of the formulas A, B. We write the table in a "formula in depended" form as a set of the following equations.

$$\neg T = F, \quad \neg F = T;$$
$$(T \cap T) = T, \quad (T \cap F) = F, \quad (F \cap T) = F, \quad (F \cap F) = F;$$
$$(T \cup T) = T, \quad (T \cup F) = T, \quad (F \cup T) = T, \quad (F \cup F) = F; \tag{2.13}$$
$$(T \Rightarrow T) = T, \quad (T \Rightarrow F) = F, \quad (F \Rightarrow T) = T, \quad (F \Rightarrow F) = T;$$
$$(T \Leftrightarrow T) = T, \quad (T \Leftrightarrow F) = F, \quad (F \Leftrightarrow T) = F, \quad (T \Leftrightarrow T) = T.$$

We use the above set of equations (2.13) to evaluate logical values of formulas.

Example 2.1

Given a formula $(A \Rightarrow (\neg A \cap B))$, such that logical values of its basic components, i.e. the propositional formulas A, B are: A=T, and B=F. We calculate the logical value of the formula $(A \Rightarrow (\neg A \cap B))$ by substituting the logical values for the formulas A, B and applying the equations (2.13) as follows.

$$(T \Rightarrow (\neg T \cap F)) = (T \Rightarrow (F \cap F)) = (T \Rightarrow F) = F.$$

Exercise 2.5

Given a formula A: $(((a \cup b) \cap \neg c) \Rightarrow a)$. Evaluate the logical value of A for the following sets of logical values of its basic components, i.e. for the propositional variables a, b: 1. a=T, b=F, c=F, and 2. a=F, b=T, c=T.

Solution
1. Let a=T, b=F, c=F. We evaluate the logical value of A as follows.

$$(((T \cup F) \cap \neg F) \Rightarrow T) = ((T \cap \neg F) \Rightarrow T) = ((T \cap T) \Rightarrow T) = (T \Rightarrow T) = T.$$

2. Let a=F, b=T, c=T. We evaluate the logical value of A as follows.

$$(((F \cup T) \cap \neg T) \Rightarrow F) = ((T \cap \neg T) \Rightarrow T) = ((T \cap F) \Rightarrow T) = (F \Rightarrow T) = T.$$

Extensional Connectives

We observe that our connectives are such that the logical value of a given formula build by means of its connectives depends only of logical values of its factors. Connectives with this property are called extensional. We hence adopt the following definition.

Definition 2.2

We call a propositional connective **extensional** *if the logical value of a given formula build by means of this connective depends only of logical values of its factors.*

Fact 2.1

All connectives $\neg, \cup, \cap, \Rightarrow$*, and* $\Leftrightarrow$ *are* **extensional.**

In everyday language there are expressions which are propositional connectives but are not extensional. They do not play any role in mathematics and so they are not discussed in classical logic.

Other Extensional Connectives

The propositional classical connectives $\cap, \cup, \Rightarrow, \Leftrightarrow, \neg$ are not the only extensional connectives. We define here all possible unary and binary two valued extensional connectives.

An extensional *unary connective* $\bigtriangledown$ enables us to form from any formula A, a new formula $\bigtriangledown A$, whose logical value is defined in terms of the logical value of A only, i.e. by means of a table of a type (2.3).

Thus there as many *unary connectives* as there are functions f from the set $\{T, F\}$ to the set $\{T, F\}$, that is $2^2 = 4$.

All Unary Connectives

A	$\bigtriangledown_1 A$	$\bigtriangledown_2 A$	$\neg A$	$\bigtriangledown_4 A$
T	F	T	F	T
F	F	F	T	T

(2.14)

An extensional *binary connective* $\circ$ permits us to form, of any two formulas A and B, a new formula $(A \circ B)$, whose logical value is defined from the logical values A and B only, i.e. by means of a table similar to (2.4), (2.5), (2.48), (2.11).

So, there are as many *binary connectives* as many functions f from a set $\{T, F\} \times \{T, F\}$ (four elements) to a set $\{T, F\}$ (two elements) that is, $2^4 = 16$.

All Binary Connectives

A	B	$(A \circ_1 B)$	$(A \cap B)$	$(A \circ_3 B)$	$(A \circ_4 B)$
T	T	F	T	F	F
T	F	F	F	T	F
F	T	F	F	F	T
F	F	F	F	F	F
A	B	$(A \downarrow B)$	$(A \circ_6 B)$	$(A \circ_7 B)$	$(A \leftrightarrow B)$
T	T	F	T	T	T
T	F	F	T	F	F
F	T	F	F	T	F
F	F	T	F	F	T
A	B	$(A \circ_9 B)$	$(A \circ_{10} B)$	$(A \circ_{11} B)$	$(A \cup B)$
T	T	F	F	F	T
T	F	T	T	F	T
F	T	T	F	T	T
F	F	F	T	T	F
A	B	$(A \circ_{13} B)$	$(A \Rightarrow B)$	$(A \uparrow B)$	$(A \circ_{16} B)$
T	T	T	T	F	T
T	F	T	F	T	T
F	T	F	T	T	T
F	F	T	T	T	T

(2.15)

Functional Dependency

It can be proved that all propositional connectives, as defined by tables (2.14) and (2.15), i.e. whether unary or binary, can be defined in terms of *disjunction* and *negation.*

This property of defining a set of connectives in terms of its proper subset is called **a functional dependency** of connectives.

There are also two other *binary* connectives which suffice, each of them separately, to define *all two valued* connectives, whether unary or binary. These connectives play a special role and are denoted in our table (2.15) by $\downarrow$ and $\uparrow$, respectively.

The connective $\uparrow$ was discovered in 1913 by H.M. Sheffer, who called it *alternative negation.* Now it is often called simply as *Sheffer's connective.* The formula $(A \uparrow B)$ is read: *not both A and B.*

The connective $\downarrow$ was discovered by J. Łukasiewicz and named *joint negation.* The formula $(A \downarrow B)$ is read: *neither A nor B.*

We define formally and examine the functional dependency of connectives in Chapter 3. We state here some important facts to be proved in Chapter 3.

Fact 2.2 *All two-valued propositional connectives and in particular our connec-*

tives $\neg, \cup, \cap, \Rightarrow$, *and* $\Leftrightarrow$ *are* **a functionally dependent**.

In particular, we prove the following.

Fact 2.3

The alternative negation connective $\uparrow$, *and the joint negation.* $\downarrow$ *suffice, each of them separately, to define all propositional connectives , whether unary or binary.*

The following was proved n 1925 by a Polish mathematician E. Żyliński.

Fact 2.4

No propositional connective other than $\uparrow$ *and* $\downarrow$ *suffices to define all the remaining connectives.*

We show now as examples how to define some of our connectives $\neg, \cup, \cap, \Rightarrow$, and $\Leftrightarrow$ in terms of $\uparrow$ or $\downarrow$ leaving the definability of other connectives as an exercise.

Example 2.2

Definition of negation $\neg$ *in terms of* $\uparrow$.

This is an interesting example as it shows that one can define a one argument connective in terms of a two argument connective.

Let's now look at Sheffer's *alternative negation* connective $\uparrow$.

Alternative Negation $\uparrow$

A	B	$(A \uparrow B)$
T	T	F
T	F	T
F	T	T
F	F	T

(2.16)

We now write the table (3.3) in the "formula independed" form of the following equations.

$$(T \uparrow T) = F,\ (T \uparrow F) = T,\ (F \uparrow T) = T,\ (F \uparrow F) = T \tag{2.17}$$

Observe that $(T \uparrow T) = F$ and $(F \uparrow F) = T$. This means that logical value of a formula $(A \uparrow A)$ is the same as logical value of a formula $\neg A$, for any logical value the formula A can take. We write it following our notation as as

$$\neg A = (A \uparrow A) \tag{2.18}$$

and call it a *definition* of $\neg$ in terms of $\uparrow$. We verify its correctness of of by building the table below.

A	$\neg A$	$(A \uparrow A)$ *computation*	$(A \uparrow A)$
T	**F**	$(T \uparrow T) = F$	**F**
F	**T**	$(T \uparrow T) = F$	**T**

(2.19)

The table shows that the logical value of a formula $\neg A$ is the same as logical value of a formula $(A \uparrow A)$, for any logical value their basic component A can take, i.e. that our definition (2.18) is correct.

Example 2.3

Definition of conjunction $\cap$ *in terms of* $\uparrow$.

Observe now that the Sheffer's connective table (3.3) looks as a negation of the conjunction table (2.4). It means that the logical value a formula $(A \cap B)$ is the same as logical value of a formula $\neg(A \uparrow B)$, for all logical values of A and B. We write it as

$$(A \cap B) = \neg(A \uparrow B). \tag{2.20}$$

We have just proved the formula (2.18) to be true for any formula and hence for the formula $\neg(A \uparrow B)$, i.e. we get that $\neg(A \uparrow B) = (A \uparrow B) \uparrow (A \uparrow B)$. The formula (2.24) becomes $(A \cap B) = (A \uparrow B) \uparrow (A \uparrow B)$.

We call the equality

$$(A \cap B) = (A \uparrow B) \uparrow (A \uparrow B) \tag{2.21}$$

the *definition* of conjunction in terms of negation and Sheffer's connective.

Let's now examine the Łukasiewicz' s *joint negation connective* $\downarrow$. The formula $A \downarrow B$ is read: *neither A nor B*. As it is a special connective we re-write its truth table separately.

Joint Negation $\downarrow$

A	B	$(A \downarrow B)$
T	T	F
T	F	F
F	T	F
F	F	T

(2.22)

We now write the table (3.30) in an "formula independed" form of the following equations.

$$(T \downarrow T) = F,\ (T \downarrow F) = F,\ (F \downarrow T) = F,\ (F \downarrow F) = T \tag{2.23}$$

Observe that $T \downarrow T = F$ and $F \downarrow F = T$. This means that logical value of a formula $(A \downarrow A)$ is the same as logical value of a formula $\neg A$, for any logical value the formula A can take. We write it as

$$\neg A = (A \downarrow A) \tag{2.24}$$

and call it a *definition* of $\neg$ in terms of $\downarrow$. We verify its correctness of of by building the table below.

A	$\neg A$	$(A \downarrow A)$ *computation*	$(A \downarrow A)$
T	**F**	$(T \downarrow T) = F$	**F**
F	**T**	$(F \downarrow F) = T$	**T**

(2.25)

The table shows that the logical value of a formula $\neg A$ is the same as logical value of a formula $(A \downarrow A)$, for any logical value their basic component A can take, i.e. that our definition (2.24) is correct.

Exercise 2.6

Prove that the equality

$$(A \cup B) = ((A \downarrow B) \downarrow (A \downarrow B)) \tag{2.26}$$

defines $\cup$ in terms of $\downarrow$.

Solution
To prove the correctness of the equation (2.26) we construct a table below.

A	B	$(A \cup B)$	$((A \downarrow B) \downarrow (A \downarrow B))$
T	T	**T**	$((T \downarrow T) \downarrow (T \downarrow T)) = (F \downarrow F) = \mathbf{T}$
T	F	**T**	$((T \downarrow F) \downarrow (T \downarrow F)) = (F \downarrow F) = \mathbf{T}$
F	T	**T**	$((F \downarrow T) \downarrow (F \downarrow T)) = (F \downarrow F) = \mathbf{T}$
F	F	**F**	$((F \downarrow F) \downarrow (F \downarrow F)) = (T \downarrow T) = \mathbf{F}$

(2.27)

The table shows that the logical value of a formula $(A \cup B)$ is the same as logical value of a formula $((A \downarrow B) \downarrow (A \downarrow B))$, for any logical value the formulas can take depending of logical values of their basic components A, B, i.e. that our definition (2.26) is correct.

2.3 Examples of Propositional Tautologies

Now we connect syntax (formulas of a given language $\mathcal{L}$) with semantics (assignment of truth values to the formulas of $\mathcal{L}$). In logic we are interested in those propositional formulas that must be always true because of their syntactical structure without reference to the meaning of the propositions they represent. Such formulas are called *propositional tautologies*.

Example 2.4

Given a formula $(A \Rightarrow A)$*. Lets now evaluate its logical value for all possible logical values of its basic component A, i.e. for A=T, and A=F. We put our calculation in a form of a table below.*

A	$(A \Rightarrow A)$ *computation*	$(A \Rightarrow A)$
T	$(T \Rightarrow T) = T$	**T**
F	$(F \Rightarrow F) = T$	**T**

(2.28)

The logical value of the formula $(A \Rightarrow A)$ is always T, what means that it is a **propositional tautology**. The table (11.27) is called a **truth table** for the formula $(A \Rightarrow A)$.

Example 2.5

We construct a **truth table** *for a formula* $(A \Rightarrow B)$ *as follows.*

A	B	$(A \Rightarrow B)$ *computation*	$(A \Rightarrow B)$
T	T	$(T \Rightarrow T) = T$	**T**
T	F	$(T \Rightarrow F) = F$	**F**
F	T	$(F \Rightarrow T) = T$	**T**
F	F	$(F \Rightarrow F) = T$	**T**

(2.29)

The logical value of the formula $(A \Rightarrow B)$ is F for A=T and B=F what means that it is *not a propositional tautology.* We put these ideas in a form of the following definition.

Definition 2.3

For any formula A of a propositional language $\mathcal{L}$*, we say that A is a propositional* **tautology** *if and only if the logical value of A is T (we write it A=T) for all possible logical values of its basic components. We write*

$$\models A$$

to denote that A is a **tautology**.

Examples of Propositional Tautologies

Given any formula A of $\mathcal{L} = \mathcal{L}_{\{\neg,\, \cap,\, \cup,\, \Rightarrow,\, \Leftrightarrow\}}$. Here are some basic classical propositional tautologies, the first of which we have just proved as the example by constructing the table (11.27). We leave the proofs of others as an easy exercise.

Identity for Implication

$$\models (A \Rightarrow A) \tag{2.30}$$

Identity for Equivalence

$$\models (A \Leftrightarrow A) \tag{2.31}$$

Excluded Middle

$$\models (\neg A \cup A) \tag{2.32}$$

One of the most frequently used classical tautologies are the laws of detachment for implication and equivalence. The implication law was already known to the Stoics (3rd century BC) and a rule of inference, based on it is called *Modus Ponens*, so we use the same name here.

Modus Ponens

$$\models ((A \cap (A \Rightarrow B)) \Rightarrow B) \tag{2.33}$$

Detachment

$$\models ((A \cap (A \Leftrightarrow B)) \Rightarrow B) \tag{2.34}$$

$$\models ((B \cap (A \Leftrightarrow B)) \Rightarrow A)$$

Mathematical and not only mathematical theorems are usually of the form of an implication, so we will discuss some terminology and more properties of implication.

Sufficient Given an implication $(A \Rightarrow B)$, A is called a *sufficient condition* for B to hold.

Necessary Given an implication $(A \Rightarrow B)$, B is called a *necessary condition* for A to hold.

Simple The implication $(A \Rightarrow B)$ is called a *simple implication.*

Converse Given a simple implication $(A \Rightarrow B)$, the implication $(B \Rightarrow A)$ is called a *converse implication* to $(A \Rightarrow B)$.

Opposite Given a simple implication $(A \Rightarrow B)$, the implication $(\neg B \Rightarrow \neg A)$ is called an *opposite implication* to $(A \Rightarrow B)$. It is also often called a *contrapositive* implication.

Contrary Given a simple implication $(A \Rightarrow B)$, the implication $(\neg A \Rightarrow \neg B)$ is called a *contrary implication* to $(A \Rightarrow B)$.

Each of the following pairs of implications: *a simple* and *an opposite*, and *a converse* and *a contrary* are equivalent, i.e. the following formulas are tautologies:

Laws of contraposition (1)

$$\models ((A \Rightarrow B) \Leftrightarrow (\neg B \Rightarrow \neg A)), \tag{2.35}$$
$$\models ((B \Rightarrow A) \Leftrightarrow (\neg A \Rightarrow \neg B)).$$

The laws of contraposition (2.35) make it possible to replace, in any deductive argument, a sentence of the form $(A \Rightarrow B)$ by $(\neg B \Rightarrow \neg A)$, and conversely. The relationships between all implications involved in the contraposition laws are usually shown graphically in a following form, which is called the *square of opposition.*

$$\begin{tikzcd}
(A \implies B) \arrow[r, dash, dashed, "\text{converse}"] \arrow[d, dash, dashed, "\text{contrary}"'] \arrow[dr, dash, dashed, "\textbf{contrapositive}"] & (B \implies A) \arrow[d, dash, dashed, "\text{contrary}"] \arrow[dl, dash, dashed] \\
(\neg A \implies \neg B) \arrow[r, dash, dashed, "\text{converse}"'] & (\neg B \implies \neg A)
\end{tikzcd}$$

Equivalent implications are situated at the vertices of one and the same diagonal. It follows from the contraposition laws that to prove all of the following implications: $(A \Rightarrow B)$, $(B \Rightarrow A)$, $(\neg A \Rightarrow \neg B)$, $(\neg B \Rightarrow \neg A)$, it suffices to prove any pairs of those implications which are situated at one and the same side of the square, since the remaining two implications are equivalent to those already proved to be true.

Consider now the following tautology:

$$\models ((A \Leftrightarrow B)) \Leftrightarrow ((A \Rightarrow B) \cap (B \Rightarrow A))). \tag{2.36}$$

The above tautology (2.36) says that in order to prove a theorem of a form of $(A \Leftrightarrow B)$ it suffices to prove two implications: the *simple* one $(A \Rightarrow B)$ and the *converse* one $(B \Rightarrow A)$. Conversely, if a formula $(A \Leftrightarrow B)$ is a theorem, then the implications $(A \Rightarrow B)$ and $(B \Rightarrow A)$ are also theorems.

In other words, B is then a *necessary condition* for A, and at the same time B is a *sufficient condition* for A. Accordingly, we say that a theorem of the form of a formula $(A \Leftrightarrow B)$ is often formulated as: " B is necessary and sufficient condition for A".

Other laws developed by the Stoics are the *hypothetical syllogism* and *modus tollendo ponens*. We present them here in a form of logical tautology, not as the rule of reasoning as it was developed. The relationship between those two approaches is quite obvious and will be discussed in detail in the proof theory chapter.

Hypothetical syllogism

$$\models (((A \Rightarrow B) \cap (B \Rightarrow C)) \Rightarrow (A \Rightarrow C))$$

$$\models ((A \Rightarrow B) \Rightarrow ((B \Rightarrow C) \Rightarrow (A \Rightarrow C))) \tag{2.37}$$

$$\models ((B \Rightarrow C) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C))).$$

Modus tollendo ponens

$$\models (((A \cup B) \cap \neg A) \Rightarrow B) \tag{2.38}$$

$$\models (((A \cup B) \cap \neg B) \Rightarrow A)$$

Here are some other tautologies with a history centuries old. First is called *Duns Scotus Law* after an eminent medieval philosopher who lived at the turn of the 13th century. Second is called *Clavius Law* , after Clavius, a Euclid commentator who lived in the late 16th century. The reasonings based on this law were already known to Euclid, but this type of inference became popular in scholarly circles owing to Clavius, hence the name. The third is called *Frege Laws* after G. Frege who was first to give a formulation of the classical propositional logic as a formalized axiomatic system in 1879, adopting the second of them as one of his axioms.

Duns Scotus

$$\models (\neg A \Rightarrow (A \Rightarrow B)) \tag{2.39}$$

Clavius

$$\models ((\neg A \Rightarrow A) \Rightarrow A) \tag{2.40}$$

Frege

$$\models (((A \Rightarrow (B \Rightarrow C)) \cap (A \Rightarrow B)) \Rightarrow (A \Rightarrow C)) \tag{2.41}$$

$$\models ((A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C)))$$

Double Negation

$$\models (\neg\neg A \Leftrightarrow A) \tag{2.42}$$

Next set of tautologies deal with **apagogic proofs** which are the proofs by **reductio ad absurdum**. The method of apagogic proof consists in negating the theorem which is to be proved. If the assumption that the theorem is false yields a contradiction, then we conclude that the theorem is true. The correctness of this reasoning is based on the following tautology.

Reductio ad Absurdum

$$\models ((\neg A \Rightarrow (B \cap \neg B)) \Rightarrow A) \tag{2.43}$$

If the theorem to be proved by reductio ad absurdum is of the form of an implication $(A \Rightarrow B)$, then the prove often follows a following pattern: it is assumed that $\neg(A \Rightarrow B)$ is true, and we try to deduce a contradiction from this assumption. If we succeed in doing so, then we infer that the implication $(A \Rightarrow B)$ is true. The correctness of this reasoning is based on the following version the **reductio ad absurdum** tautology (2.43).

$$\models (((\neg(A \Rightarrow B) \Rightarrow (C \cap \neg C)) \Rightarrow (A \Rightarrow B)).$$

Sometimes to prove $(A \Rightarrow B)$ it is assumed that $(A \cap \neg B)$ is true and if the assumption leads to contradiction, then we deduce that the implication $(A \Rightarrow B)$ is true. In this case a tautology, which guarantee the correctness this kind of argument is:

$$\models (((A \cap \neg B) \Rightarrow (C \cap \neg C)) \Rightarrow (A \Rightarrow B)).$$

Often, when assuming $(A \cap \neg B)$, we arrive, by deductive reasoning, at the conclusion $\neg A$. Then we need the following tautology:

$$\models (((A \cap \neg B) \Rightarrow \neg A) \Rightarrow (A \Rightarrow B)).$$

Sometimes, on assuming $(A \cap \neg B)$ we arrive by deductive reasoning at the conclusion B. The following tautology is then applied:

$$\models (((A \cap \neg B) \Rightarrow B) \Rightarrow (A \Rightarrow B)).$$

The proofs based on the application of the laws of contraposition (2.35) are also classed as apagogic. Instead of proving a simple theorem $(A \Rightarrow B)$ we prove the opposite theorem $(\neg B \Rightarrow \neg A)$, which is equivalent to the simple one. The following two tautologies, also called laws of contraposition, are used, respectively, when the hypothesis or the thesis of the theorem to be proved is in the form of a negation.

Laws of Contraposition (2)

$$\models ((\neg A \Rightarrow B) \Leftrightarrow (\neg B \Rightarrow A)), \tag{2.44}$$

$$\models ((A \Rightarrow \neg B) \Leftrightarrow (B \Rightarrow \neg A)).$$

We present now some tautologies characterizing basic properties of *conjunction*, *disjunction*, *quivalence*, and their interactions.

Conjunction

$$\models ((A \cap B) \Rightarrow A), \quad \models ((A \cap B) \Rightarrow B),$$
$$\models (((A \Rightarrow B) \cap (A \Rightarrow C)) \Rightarrow (A \Rightarrow (B \cap C))),$$
$$\models (((A \Rightarrow B) \cap (C \Rightarrow D)) \Rightarrow ((A \cap C) \Rightarrow (B \cap D))),$$
$$\models (A \Rightarrow (B \Rightarrow (A \cap B))).$$

Disjunction

$$\models ((A \Rightarrow (A \cup B)), \quad \models ((B \Rightarrow (A \cup B)),$$
$$\models (((A \Rightarrow B) \cap (B \Rightarrow C)) \Rightarrow ((A \cup B) \Rightarrow C)),$$
$$\models (((A \Rightarrow B) \cap (C \Rightarrow D)) \Rightarrow ((A \cup C) \Rightarrow (B \cup D))).$$

Here are some more important and frequently used equivalence tautologies, called also the equivalence laws.

Idempotence

$$\models ((A \cap A) \Leftrightarrow A), \qquad \models ((A \cup A) \Leftrightarrow A),$$

Associativity

$$\models (((A \cap B) \cap C) \Leftrightarrow (A \cap (B \cap C))),$$
$$\models (((A \cup B) \cup C) \Leftrightarrow ((A \cup (B \cup C))).$$

Commutativity

$$\models ((A \cap B) \Leftrightarrow (B \cap A)), \qquad \models ((A \cup B) \Leftrightarrow (B \cup A)).$$

Distributivity

$$\models ((A \cap (B \cup C)) \Leftrightarrow ((A \cap B) \cup (A \cap C))), \tag{2.45}$$

$$\models ((A \cup (B \cap C)) \Leftrightarrow ((A \cup B) \cap (A \cup C))). \tag{2.46}$$

De Morgan

$$\models (\neg(A \cup B) \Leftrightarrow (\neg A \cap \neg B)), \quad \models (\neg(A \cap B) \Leftrightarrow (\neg A \cup \neg B)). \tag{2.47}$$

Implication
$$\models ((A \Rightarrow B) \Leftrightarrow (\neg A \cup B)). \tag{2.48}$$

Negation of Implication

$$\models (\neg(A \Rightarrow B) \Leftrightarrow (A \cap \neg B)).$$

Negation of Equivalence

$$\models (\neg(A \Leftrightarrow B) \Leftrightarrow (A \cap \neg B) \cup (B \cap \neg A)).$$

Double Negation
$$\models (\neg\neg A \Leftrightarrow A). \tag{2.49}$$

Exportation and Importation

$$\models (((A \cap B) \Rightarrow C) \Leftrightarrow (A \Rightarrow (B \Rightarrow C))).$$

De Morgan laws (2.47) are named after A. De Morgan (1806 - 1871), an English logician, who discovered analogous laws for the algebra of sets. They stated that for any sets A,B the complement of their union is the same as the intersection of their complements, and vice versa, the complement of the intersection of two sets is equal to the union of their complements. The laws of the propositional calculus were formulated later, but they are usually also called De Morgan Laws.

2.4 Predicate Language Description and Application to Artificial Intelligence

We define a *predicate language* $\mathcal{L}$ following the pattern established by the symbolic and propositional languages definitions (2.1), (7.44). The predicate language $\mathcal{L}$ is much more complicated in its structure. Its *alphabet* $\mathcal{A}$ is much richer. The definition of its set of *formulas* $\mathcal{F}$ is more complicated. In order to define the set $\mathcal{F}$ we introduce an additional set $\mathbf{T}$, called a set of *terms* of the predicate language $\mathcal{L}$. We single out this set not only because we need it for the definition of formulas, but also because of its role in the development of other notions of predicate logic.

Definition 2.4

By a **predicate language** $\mathcal{L}$ *we understand a triple*

$$\mathcal{L} = (\mathcal{A}, \mathbf{T}, \mathcal{F}), \tag{2.50}$$

where $\mathcal{A}$ *is a predicate* **alphabet***,* $\mathbf{T}$*, is the set of* **terms***, and* $\mathcal{F}$ *is a set of* **formulas***.*

Alphabet $\mathcal{A}$

The components of $\mathcal{A}$ are as follows.

1. *Propositional connectives*: $\neg$, $\cap$, $\cup$, $\Rightarrow$, $\Leftrightarrow$.

2. *Quantifiers*: we adopt two quantifiers; $\forall$ (for all, the universal quantifier) and $\exists$ (there exists, the existential quantifier).

In a case of the classical logic it is possible to adopt only *one quantifier* and to define the other in terms of it and propositional connectives. But the two quantifiers express better the common intuition, so we assume that we have two of them.

3. *Parenthesis*: (and).

4. *Variabes:* we assume that we have, as we did in the propositional case a countably infinite set VAR of *variables.* The variables now have a different meaning than they had in the propositional case. We hence call them *variables*, or *individual variables* to distinguish them from the *propositional variables.* We also denote denote them by different symbols, namely by letters $x, y, z, ...,$ with indices, if necessary. We express it by writing $VAR = \{x_1, x_2, ...\}$.

5. *Constants:* the constants represent in "real life" concrete elements of sets. We denote constants by by $c, d, e...$, with indices, if necessary. We assume that we have a countably infinite set $\mathbf{C} = \{c_1, c_2, ...\}$ of *constants.*

6. *Predicate symbols:* the predicate symbols represent "real life" relations. We denote them by $P, Q, R, ...$ with indices, if necessary. We use symbol $\mathbf{P}$ for the set of all predicate symbols. We assume that $\mathbf{P}$ it countably infinite.

In "real life" we write symbolically $x < y$ to express that element x is smaller then element y according to the two argument order relation $<$. In our predicate language $\mathcal{L}$ we represent the relation $<$ as a two argument predicate $P \in \mathbf{P}$ and write $P(x, y)$, where now x, y are *individual variables* from the set VAR.

Mathematical statements $n < 0$, $1 < 2$, $0 < m$ are represented in $\mathcal{L}$ by $P(x, c_1)$, $P(c_, c_3)$, $P(c_1, y)$, respectively. Here c_1, c_2, c_3 are any *constants* and x, y any *variables.*

7. *Function symbols:* the function symbols represent "real life" functions. We denote function symbols by $f, g, h, ...$, with indices, if necessary. We use symbol $\mathbf{F}$ for the set of all function symbols. We assume that the set $\mathbf{F}$ is countably infinite.

Set T of terms

Terms are expressions built out of function symbols and variables. They describe how we build compositions of functions. We define the set $\mathbf{T}$ of *terms* recursively as follows.

1. All *variables* are *terms.*

2. All *constants* are *terms*.

3. For any *function symbol* f representing a function on n variables, and any terms $t_1, t_2, ..., t_n$, the expression $f(t_1, t_2, ..., t_n)$ is a *term*.

4. The set $\mathbf{T}$ of *terms* is the smallest set that fulfills the conditions 1. - 3.

Consider a "real life" function given by a formula $sin(x+y)$. It is a composition of two functions defined by formulas $sinx$ and $x + y$. The sin is one argument function and we represent it as *term* $f(x)$ for $f \in \mathbf{F}$. The $+$ is a two argument function and we represent it as a *term* $g(x, y)$ for $g \in \mathbf{F}$. The "real life" function $sin(x+y)$ is hence represented by a *term* $f(g(x, y))$, where x, y are any *individual variables* from the set VAR. Observe that to obtain the predicate language representation of for example $x + y$ we can first write the real two argument function formula $x + y$ as $+(x, y)$ and then replace the addition symbol $+$ by any two argument function symbol $g \in \mathbf{F}$ and get the *term* $g(x, y)$.

Here are some more *terms* of $\mathcal{L}$.

$$h(c_1),\ f(g(c, x)),\ g(f(f(c)), g(x, y)),\ f_1(c, g(x, f(c))),\ g(g(x, y), g(x, h(c)))\ ....$$

Set $\mathcal{F}$ of formulas

Formulas are now expressions built out of elements of the *alphabet* $\mathcal{A}$ and the set $\mathbf{T}$ of *terms*. We denote them, as in propositional case by $A, B, C, ...$ with indices, if necessary. We build them, as before in recursive steps, the fist of them says as in the propositional case: all *atomic formulas* are formulas. The *atomic formulas* are the simplest formulas as the *propositional variables* were in the case of *propositional language*. We define them as follows.

Definition 2.5 *An* **atomic formula** *is any expression of the form* $R(t_1, t_2, ..., t_n)$ *where R is any predicate* $R \in \mathbf{P}$ *and* $t_1, t_2, ..., t_n$ *are terms, i.e.* $t_1, t_2, ..., t_{\in}\mathbf{T}$.

To represent a mathematical statement $x + y = 5$ in $\mathcal{L}$ we first observe that $=$ as a two argument relation and $+$ is a two argument function, x, y are variables and 5 is a number. We represent, as before, $+$ by a two argument function symbol $g \in \mathbf{F}$, the relation $=$ by a predicate symbol $P \in \mathbf{P}$, the number 5 by a constant $c \in \mathbf{C}$. We re-write $x + y = 5$ as $= (+(x, y), 5)$, replace mathematical symbols by corresponding $\mathcal{L}$ symbols and get an *atomic formula* $P(g(x, y), c)$ representing in $\mathcal{L}$ the statement $x + y = 5$. We have used the same letters x, y to represent mathematical and and atomic formula variables. We can also use any other letters for individual variables in the atomic formula re[presenting $x + y = 5$. For example $P(g(x_1, x_2), c)$, $P(g(y, x), c)$.

Here are some more **atomic formulas** of $\mathcal{L}$.

$$Q(c),\ Q(x),\ Q(g(x_1, x_2)),\ R(c, d),\ R(x, f(c)),\ R(g(x, y), f(g(c, z))), ...$$

Definition 2.6

The set $\mathcal{F}$ of formulas of $\mathcal{L}$ is the smallest set meeting the following conditions.

1. *All* **atomic formulas** *(definition 10.2) are formulas;*

2. *If A, B are formulas, then $\neg A, (A \cap B), (A \cup B), (A \Rightarrow B), (A \Leftrightarrow B)$ are formulas;*

3. *If A is a formula, then $\forall xA$, $\exists xA$ are formulas for any variable $x \in VAR$.*

Here are some formulas of $\mathcal{L}$.

$$R(c,d), \quad \exists yR(y, f(c)), \quad R(x,y), \quad (\forall xR(x, f(c)) \Rightarrow \neg R(x,y)),$$

$$(R(c,d) \cap \forall zR(z, f(c))), \quad \forall yR(y, \; g(c, g(x, f(c)))), \quad \forall y \neg \exists xR(x,y).$$

Let's look now closer at the following formulas.

$$R(c_1, c_2), \; R(x,y), \; (R(y,d) \Rightarrow R(a,z)), \; \exists xR(x,y), \; \forall yR(x,y), \exists x \forall yR(x,y).$$

Here are some simple observations.

1. Some formulas are without quantifiers.
For example formulas $R(c_1, c_2)$, $R(x,y)$, $(R(y,d) \Rightarrow R(a,z))$. A formula without quantifiers is called an **open formula**.

Variables x, y in $R(x,y)$ are called **free variables**. The variables y in R(y, d) and z in R(a,z) are also **free**.

2. Quantifiers **bind** variables within formulas.

The variable x is **bounded** by $\exists x$ in the formula $\exists xR(x,y)$, the variable y is **free**. The variable y is **bounded** by $\forall y$ in the formula $\forall yR(x,y)$, the variable y is **free**.

3. The formula $\exists x \forall yR(x,y)$ does not contain any free variables, neither does the formula $R(c_1, c_2)$. A formula without any free variables is called a **closed formula** or a **sentence**.

Sometimes in order to distinguish more easily *which variable* is **free** and which is **bound** in the formula we might use the bold face type for the quantifier bound variables and write the formulas as follows.

$$(\forall \mathbf{x} Q(\mathbf{x}, y), \quad \exists \mathbf{y} P(\mathbf{y}), \quad \forall \mathbf{y} R(\mathbf{y}, g(c, g(x, f(c)))),$$

$$(\forall \mathbf{x} P(\mathbf{x}) \Rightarrow \exists \mathbf{y} Q(x, \mathbf{y})), \quad (\forall \mathbf{x}(P(\mathbf{x}) \Rightarrow \exists \mathbf{y} Q(\mathbf{x}, \mathbf{y})))$$

Observe that the formulas $\exists \mathbf{y} P(\mathbf{y})$, $(\forall \mathbf{x}(P(\mathbf{x}) \Rightarrow \exists \mathbf{y} Q(\mathbf{x}, \mathbf{y})))$ are **closed**. We call a close formula a **sentence**.

Example 2.6

Consider **atomic formulas***:* $P(y), Q(x,c), R(z), P_1(g(x,y),z)$*. Here are some non atomic formulas formed out of them.*

1. $(P(y) \cup \neg Q(x,c)) \in \mathcal{F}$*. This is an* **open** *formula A with two free variables x,y. We denote A this as formula* $A(x,y)$*.*

2. $\exists \mathbf{x}(P(y) \cup \neg Q(\mathbf{x},c)) \in \mathcal{F}$*. We write* $\mathbf{x}$ *to denote that x is a* **bound** *variable. The variable y is* **free***. This is a formula B with one free variable y. We denote B as a formula* $B(y)$*.*

3. $\forall \mathbf{y}(P(\mathbf{y}) \cup \neg Q(x,c)) \in \mathcal{F}$*. The variable y is* **bound***, the variable x is* **free***. We denote this formula by for example* $A_1(x)$*.*

4. $\forall \mathbf{y}\exists \mathbf{x}(P(y) \cup \neg Q(\mathbf{x},c)) \in \mathcal{F}$ *has no free variables. It is a* **closed** *formula called also a* **sentence***.*

Exercise 2.7

Given the following formulas of $\mathcal{L}$*:*

$P(x, f(c,y)),\ \exists c P(x, f(c,y)),\ \forall x f(x, P(c,y)),\ \exists x P(x, f(c,y)) \Rightarrow \forall y P(x, f(c,y))$.

1. Indicate whether they are, or are not well formed formulas of $\mathcal{F}$*. For those which are not in* $\mathcal{F}$ *write a correct formula. 2. For each correct, or corrected formula identify all components: connectives, quantifiers, predicate and function symbols, and list all its terms. 3. For each formula identify its s free and bound variables. State which are open and which are closed formulas (sentences), if any.*

Solution

Formula $A_1 = P(x, f(c,y))$.

It is a correct **atomic** formula. P is a 2 argument *predicate* symbol, f is a 2 argument *function* symbol, c is a *constant.* We write it symbolically: $P \in \mathbf{P},\ f \in \mathbf{F},\ c \in \mathbf{C}$. It is an **open** formula with two *free variables* x,y. We denote it by $A_1(x,y)$. It has no *bound variables.*

Formula $A_2 = \exists c P(x, f(c,y))$.

It is a not a correct formula, i.e. $\exists c P(x, f(c,y)) \notin \mathcal{F}$. The expression $\exists c$ has no meaning because c is a constant, not a variable.

The corrected formulas are: $B_1 = \exists x P(x, f(c,y))$, $B_2 = \exists y P(x, f(c,y))$, and formulas $B = \exists z P(z, f(c,y))$ for any variable z different then x and y.

None of the correct formulas are open. Variable y is free in $B_1 = B_1(y)$, variable x is free in $B_2 = B_2(x)$, both variables x and y are free in all formulas $B = B(x,y)$. All formulas are nether close, nor open. The *terms* appearing in any of them are the same as in $A_1 = P(x, f(c,y))$ and are: $x, y, c, f(c,y)$.

Formula $A_3 = \forall x f(x, P(c, y))$.

It is a not a correct formula, i.e. $\forall x f(x, P(c, y)) \notin \mathcal{F}$. The function symbol f in front $f(x, P(c, y))$ indicate a term and terms are not formulas. Moreover, the atomic formula $P(c, y)$ can't be put inside a term!

Formula $A_4 = \exists x P(x, f(c, y)) \Rightarrow \forall y P(x, f(c, y))$

It is a not a correct formula. The correct formula is $A = (\exists x P(x, f(c, y)) \Rightarrow \forall y P(x, f(c, y)))$. It has two free variables x and y and we write it as $A = A(x, y)$.

We often use logic symbols, while writing mathematical statements in a more symbolic way. For example, mathematicians to say "all natural numbers are greater then zero and some integers are equal 1" often write

$$x \geq 0, \forall_{x \in N} \text{ and } \exists_{y \in Z}, y = 1.$$

Some of them who are more "logic oriented" would write it as

$$\forall_{x \in N}\ x \geq 0\ \cap\ \exists_{y \in Z}\ y = 1,$$

or even as

$$(\forall_{x \in N}\ x \geq 0\ \cap\ \exists_{y \in Z}\ y = 1).$$

Observe that none of the above symbolic statements, not even the last one, are formulas of the *predicate language*. These are mathematical statements written with mathematical and logic symbols. They are written with different degrees of "logical precision", the last being, from a logician point of view, the most precise.

Our goal now is to "translate " mathematical and natural language statement into correct logical formulas, i.e. into formulas of the *predicate language* $\mathcal{L}$. Let's start with some observations about the statements above.

The quantifiers in $\forall_{x \in N}$ and $\exists_{y \in Z}$ used in all of them are not the one used in logic. In our language $\mathcal{L}$ we use only quantifiers $\forall x$ and $\exists y$, for any variables $x, y \in VAR$. The quantifiers $\forall_{x \in N}$, $\exists_{y \in Z}$ are called *quantifiers with restricted domain*. The first is restricted to the domain of natural numbers, the second to the integers. The restriction of the quantifier domain can, and often is given by more complicated statements. For example we say "for all $x > 2$" and write $\forall_{x > 2}$, or we say "exists $x > 2$ and at same time $x + 2 < 8$" and write symbolically $\exists_{(x > 2 \cap x + 2 < 8)}$. We introduce the quantifiers with restricted domain into our predicate logic language by expressing them within the language $\mathcal{L}$ as follows.

Definition 2.7

The quantifiers $\forall_{A(x)}$, $\exists_{A(x)}$ *are called* **quantifiers with restricted domain***, or* **restricted quantifiers***, where* $A(x) \in \mathcal{F}$ *is any formula with any free variable* $x \in VAR$.

A formula $\forall_{A(x)}B(x)$ *stands for a formula* $\forall x(A(x) \Rightarrow B(x)) \in \mathcal{F}$. *We write it symbolically as*

$$\forall_{A(x)}\ B(x) \equiv \forall x(A(x) \Rightarrow B(x)). \tag{2.51}$$

A formula $\exists_{A(x)}B(x)$ *stands for a formula* $\exists x(A(x) \cap B(x)) \in \mathcal{F}$. *We write it symbolically as*

$$\exists_{A(x)}\ B(x) \equiv \exists x(A(x) \cap B(x)) \tag{2.52}$$

The definition 2.7 of restricted quantifiers is obviously faithful to our intuitive meaning of quantifiers. We use informally a symbol $\equiv$ to stress that we they are in a sense equivalent. We call (8.18) and (8.19) **transformations rules** for restricted quantifiers.

We carry our translations of mathematical statements written with logical symbols into a formula of predicate language $\mathcal{L}$ a sequence of steps. Given a mathematical statement **S** written with logical symbols. We obtain a corresponding formula A that is our translation into $\mathcal{L}$ by conducting the following steps.

Step 1. We identify *basic statements* in **S**, i.e. mathematical statements that involve only relations. They will be translated into *atomic formulas.*

We identify the *relations* in the basic statements and choose the *predicate symbols* as their names.

We identify all *functions* and *constants* (if any) in the basic statements and choose the *function symbols* and *constant symbols* as their names.

Step 2. We write the basic statements as *atomic formulas* of $\mathcal{L}$.

Remember that in the predicate language $\mathcal{L}$ we write function symbol in front of the function arguments, not between them as we write in mathematics. The same applies to relation symbols when we form atomic formulas. For example a basic mathematical statement $x + 2 > y$ could be re-written as $> (+(x, 2), y)$, and then we could immediately write it as an *atomic formula* $P(f(x, c), y)$, where $P \in \mathbf{P}$ stands for two argument relation $>$, f stands for two argument function $+$, and c stands for the number (constant) 2.

Step 3. We re-write the statement **S** a logical formula with restricted domain quantifiers.

Step 4. We apply equivalences (8.18) and (8.19) to the formula from Step 3 and obtain a formula A of $\mathcal{L}$ as a translation, i.e. a representation of the given mathematical statement.

When we conduct a translation from mathematical statement written without logical symbols we add a Step 0 to this list to first write the mathematical statement with logical symbols.

Step 0. We identify *logical connectives* and *quantifiers* and write the statement using them that is as close to the structure of a logical formula as possible.

Exercise 2.8

Given a mathematical statement **S** *written with logical symbols*

$$(\forall_{x\in N}\ x \geq 0\ \cap\ \exists_{y\in Z}\ y = 1)$$

1. *Translate it into a proper logical formula with restricted domain quantifiers i.e. into a formula of $\mathcal{L}$ that uses the restricted domain quantifiers.* **2.** *Translate your restricted domain quantifiers logical formula into a correct logical formula* **without** *restricted domain quantifiers, i.e. into a formula of $\mathcal{L}$.*

Solution

Step 1. The basic statements in **S** are: $x \in N,\ x \geq 0,\ y \in Z,\ y = 1$. The relations are: $\in N,\ \in Z,\ \geq,\ =$. We use one argument predicate symbols N, Z for $\in N, \in Z$, respectively. We use two argument predicate symbols G for $\geq$, and E for $=$. There are no functions. We have two constant symbols c_1, c_2 for numbers 0 and 1, respectively.

Step 2. We write $N(x), Z(x)$ for $x \in N, x \in Z$, respectively. $G(x, c_1)$ for $x \geq 0$ and $E(y, c_2)$ for $y = 1$. These are all atomic formulas.

Step 3. The statement **S** becomes a restricted quantifiers formula:

$$(\forall_{N(x)}\ G(x, c_1)\ \cap\ \exists_{Z(y)}\ E(y, c_2)).$$

Step 4. A formula $A \in \mathcal{F}$ that corresponds to **S** is

$$(\forall x\ (N(x) \Rightarrow G(x, c_1))\ \cap\ \exists y\ (Z(y) \cap E(y, c_2))).$$

Here is a perfectly acceptable **short solution** to exercise 2.8. We presented the long solution in order to explain all steps needed to be performed when one writes the short solution.

Example 2.7

Given a mathematical statement **S** *written with logical symbols*

$$(\forall_{x\in N}\ x \geq 0\ \cap\ \exists_{y\in Z}\ y = 1)$$

We translate it into a proper formula of $\mathcal{L}$ as follows.

The basic statements in **S** *are: $x \in N,\ x \geq 0,\ y \in Z, y = 1$. The corresponding atomic formulas of $\mathcal{L}$ are: $N(x),\ G(x, c_1),\ Z(y),\ E(y, c_2)$, respectively.*

The statement **S** *becomes becomes restricted quantifiers formula $(\forall_{N(x)} G(x, c_1)\ \cap\ \exists_{Z(y)}\ E(y, c_2))$. Applying restricted quantifiers definition 2.7 and transformation rules (8.18), (8.19) we get a following formula $A \in \mathcal{F}$*

$$(\forall x(N(x) \Rightarrow G(x, c_1)) \cap \exists y(Z(y) \cap E(y, c_2))).$$

Exercise 2.9

Here is a mathematical statement **S***:*

"For all real numbers x the following holds: If $x < 0$, then there is a natural number n, such that $x + n < 0$."

1. *Re-write* **S** *as a symbolic mathematical statement SF that only uses mathematical and logical symbols.* **2.** *Translate the symbolic statement SF into to a corresponding formula $A \in \mathcal{F}$ of the predicate language $\mathcal{L}$.*

Solution
The symbolic mathematical statement SF is : $\forall_{x \in R}(x < 0 \Rightarrow \exists_{n \in N}\ x + n < 0)$. We write R(x) for $x \in R$, N(y) for $n \in N$, and atomic formula L(x, c) for the basic statement $x < 0$. We write f(x,y) for the function $+(x, n)$ and a constant c for the number 0. We write atomic formula L(f(x,y), c) for $x + n < 0$. The symbolic statement SF becomes $\forall_{R(x)}(L(x, c) \Rightarrow \exists_{N(y)} L(f(x, y), c))$. The corresponding formula $A \in \mathcal{F}$ is $\forall x(N(x) \Rightarrow (L(x, c) \Rightarrow \exists y(N(y) \cap L(f(x, y), c)))$.

There are various kinds of non-mathematical statements, that obviously cannot be justified on the basis of propositional logic. Consider for example a statement

"Any friend of Mary is a friend of John and Peter is not John's friend. Hence Peter is not May's friend. "

Intuitively, what it says is always true, but translating it it into a propositional language we get a formula $((a \cap \neg b) \Rightarrow \neg c)$ that can be false. The validity of the reasoning described by the statement follows from a more complexed structure provided by the predicate language. We will discuss the notion of validity of predicate language formulas, i.e. a semantics for predicate logic later. Natural language statements and reasoning with them also play a special role in creation of *non-classical logics* and in *Artificial Intelligence* research and applications.

Exercise 2.10

Translate a natural language statement **S***: "Any friend of Mary is a friend of John and Peter is not John's friend. Hence Peter is not May's friend." into a formula $A \in \mathcal{F}$ of the predicate language $\mathcal{L}$.*

Solution

1. We identify the basic relations and functions (if any) and translate them into atomic formulas.

We have only one relation of "being a friend". It is a two argument relation. We write atomic formula F(x, y) for "x is a friend of y". We use constants m, j, p for Mary, John, and Peter, respectively. We have the following atomic formulas: F(x, m) for "x is a friend of Mary", F(x, j) for "x is a friend of John", F(p, j) for "Peter is a friend of John".

2. Statement "Any friend of Mary is a friend of John" translates into a restricted quantifier formula $\forall_{F(x,m)}\ F(x,j)$. Statement "Peter is not John's friend" translates into $\neg F(p,j)$, and "Peter is **not** May's friend" translates into $\neg F(p,m)$.

3. Restricted quantifiers formula for **S** is

$$((\forall_{F(x,m)}F(x,j)\ \cap \neg F(p,j)) \Rightarrow \neg F(p,m))$$

and the formula $A \in \mathcal{F}$ of $\mathcal{L}$ is

$$((\forall x(F(x,m) \Rightarrow F(x,j))\ \cap \neg F(p,j)) \Rightarrow \neg F(p,m)).$$

Here are simple steps we follow in order to perform translations from natural language to the symbolic predicate language $\mathcal{L}$. They are similar to the steps we used in the translations of mathematical formulas nevertheless we voice them separately and call them *rules of translation.*

Rules of translation to $\mathcal{L}$.

1. Identify the basic relations and functions (if any) and translate them into *atomic formulas.*
2. Identify propositional connectives and use symbols $\neg, \cup, \cap, \Rightarrow, \Leftrightarrow$ for them.
3. Identify quantifiers. Restricted $\forall A(x)$, $\exists A(x)$ and non-restricted $\forall x$, $\exists x$.
4. Use the symbols from **1. - 3.** and restricted quantifiers transformation rules (8.18) and (8.19) to write $A \in \mathcal{F}$ of the predicate language $\mathcal{L}$.

Example 2.8

Given a natural language statement **S***: "For any bird one can find some birds that are white." The translation of* **S** *into a formula of the predicate language* $\mathcal{L}$ *is*

$$\forall x(B(x) \Rightarrow \exists x(B(x) \cap W(x))).$$

We follow the rules of translation and get the following.

1. Atomic formulas: B(x), W(x). We write one argument predicate B(x) for " x is a bird" and one argument predicate W(x) for " x is white".

2. There is no propositional connectives in **S**.

3. Restricted quantifiers: $\forall_{B(x)}$ for "any bird " and $\exists_{B(x)}$ for "one can find some birds". A restricted quantifiers formula for **S** is $\forall_{B(x)}\exists_{B(x)}\ W(x)$.

4. By the transformation rules we get a required non-restricted formula of the predicate language $\mathcal{L}$, i.e. the formula $\forall x(B(x) \Rightarrow \exists x(B(x) \cap W(x)))$.

Observe that the quantifier $\forall x$ *binds* the variable x only in the first B(x), even if its scope covers the second appearance of B(x) as well. It happens because the second appearance of B(x) is *bounded* by the quantifier $\exists x$. Let's re-write the formula A using **x** to indicate this fact

$$\forall \mathbf{x}(B(\mathbf{x}) \Rightarrow \exists x(B(x) \cap W(x))).$$

In this case, and in the similar cases we can apply a predicate logic law of quantifiers, called *Rename Variables Law* to our formula A and get a formula B that is logically equivalent to A. It means that the formula B states exactly the same what A states but is written in a more comprehensible form:

$$\forall x(B(x) \Rightarrow \exists y(B(y) \cap W(y))).$$

We will discuss and study *Laws of Quantifiers* in the next section. There is another important law, one of the *Distributivity Laws* that allows us to transform B into a formula $\forall x \exists y(B(x) \Rightarrow (B(y) \cap W(y)))$. We express it as the following example.

Example 2.9

Given a natural language statement **S***: "For any bird one can find some birds that white." The translation of* **S** *into a formula of the predicate language* $\mathcal{L}$ *is*

$$\forall x \exists y(B(x) \Rightarrow (B(y) \cap W(y))).$$

Exercise 2.11

Translate into $\mathcal{L}$ *a natural language statement*
S*: " Some patients like all doctors."*

Solution.

1. Atomic formulas: P(x), D(x), L(x, y). We write one argument predicate P(x) for " x is a patient", one argument predicate D(x) for " x is a doctor", and two argument predicate L(x,y) for " x likes y".

2. There is no propositional connectives in **S**.

3. Restricted quantifiers: $\exists_{P(x)}$ for "some patients " and $\forall_{D(x)}$ for "all doctors". Observe that we can't write L(x, D(y)) for "x likes doctor y". D(y) is a predicate, not a term and hence L(x, D(y)) is not a formula. We have to express the statement " x likes all doctors y" in terms of restricted quantifiers and predicate L(x,y) only. The statement " x likes all doctors y" means " all doctors y are liked by x", i.e. "for all doctors y, x likes y". This translates to $\forall_{D(y)}L(x, y)$ and the statement **S** translates to $\exists_{P(x)}\forall_{D(x)}L(x, y)$.

4. By the transformation rules we get the following translation of **S** into $\mathcal{L}$.

$$\exists x(P(x) \cap \forall y(D(y) \Rightarrow L(x, y))).$$

Translations to Logic in Artificial Intelligence

In Artificial Intelligence (AI) we usually deal with what is called an **intended interpretation**. It means we use logic symbols to describe, similarly as we do in mathematics, concrete, specific universes with specific relations and functions, or constants. In logic we use general symbols without any meaning because the logic is created to define statements (formulas) and methods of reasoning that are universally applicable (tautologically true) and hence independent of any particular domain. In AI we use as symbols *intended* names for relations, functions, and constants. The symbolic language we use is still a symbolic language, even if intended names are used. In the AI language we can write, for example , an atomic formula *Like(John, Mary)* instead of a formula $L(c_1, c_2)$ of $\mathcal{L}$. We write *greater(x, y)*, or $> (x, y)$ instead of $R(x, y)$. We leave it as an exercise to formally define the AI language you would like to use.

Example 2.10

AI formulas corresponding to a statement

S*: "For every student there is a student that is an elephant."*

are as follows.

1. Restricted quantifiers AI formula:

$$\forall_{Student(x)}\exists_{Student(x)}\ Elephant(x).$$

2. Non-restricted quantifiers AI formula :

$$\forall x(Student(x) \Rightarrow \exists x(Student(x) \cap Elephant(x))).$$

3. Re-name variables AI formula:

$$\forall x(Student(x) \Rightarrow \exists y(Student(y) \cap Elephant(y))).$$

4. AI formula after applying the the Distributivity Laws:

$$\forall x\exists y(Student(x) \Rightarrow (Student(y) \cap Elephant(y))).$$

Observe that a proper formulas of the predicate language $\mathcal{L}$ corresponding the example 2.10 statement "For every student there is a student that is an elephant." are the same as the formulas corresponding to the natural language statement "For any bird one can find some birds that white." of the example 2.9, namely
1. Restricted quantifiers $\mathcal{L}$ formula: $\forall_{P(x)}\exists_{P(x)}\ R(x)$.
2. Non-restricted quantifiers $\mathcal{L}$ formula : $\forall x(P(x) \Rightarrow \exists x(P(x) \cap Rx)))$.
3. Re-name variables $\mathcal{L}$ formula: $\forall x(P(x) \Rightarrow \exists y(P(y) \cap R(y)))$.
4. $\mathcal{L}$ formula after applying the the Distributivity Laws

$$\forall x\exists y(P(x) \Rightarrow (P(y) \cap R(y))).$$

The predicate symbols P, R, Student, Elephant denote in all cases one argument predicates but AI predicate symbols Student, Elephant (of a slightly different language than $\mathcal{L}$) impose a particular meaning called the **intended interpretation**. The predicate symbols P, R and any elements of the set of all predicate symbols **P** of $\mathcal{L}$.

Exercise 2.12

Translate a natural language statement "Any friend of Mary is a friend of John and Peter is not John's friend. Hence Peter is not Mary's friend." into a formula A of the predicate AI language (of your choice).

Solution
Statement "Any friend of Mary is a friend of John" translates into a restricted quantifier AI formula $\forall_{Friend(x,Mary)}\ Friend(x, John)$.
Statement "Peter is not John's friend" translates into $\neg Friend(Peter, John)$, and "Peter is not Mary's friend" translates into $\neg Friend(Peter, Mary)$.

Restricted quantifiers AI formula for **S** is $((\forall_{Friend(x,Mary)}\ Friend(x, John) \cap \neg Friend(Peter, John)) \Rightarrow \neg Friend(Peter, Mary))$.

The AI formula is $((\forall x(Friend(x, Mary) \Rightarrow Friend(x, John)) \cap \neg Friend(Peter, John)) \Rightarrow \neg Friend(Peter, Mary))$.

The AI formulas are very useful, as they "read" as natural language statements but it is very important to remember that they *do not carry any meaning*, as the natural language statements do to the reader. An atomic formula *Friend(Peter, John)* is just an atomic formula of a symbolic AI language as $P(c, d)$ is in $\mathcal{L}$. We assign a meaning to them i.e. their *semantics* in a separate step as we did in the propositional case. The first step in this process is an assignment of an *interpretation* in a non-empty set U of the predicate, functional and constant symbols. Each symbol can have many *interpretations* in a given set and we can can define the interpretations an many sets. The AI *intended interpretation* of the two argument predicate named *Friend* and constants *Peter, John* is to define the set U and a relation *Friend.* This relation must hold between elements *Peter, John* and other elements of U in a way we want to define what "friendship" means in the set U. This is called in AI a *conceptualization.*

2.5 Predicate Semantics: Description and Laws of Quantifiers

The notion of predicate tautology is much more complicated then that of the propositional. We define it formally in later chapters. Predicate tautologies are also called *valid formulas*, or *laws of quantifiers* to distinguish them from the propositional case. We provide here a motivation, examples and an intuitive definition of the predicate tautology. We also list and discuss the most used and useful tautologies and equational laws of quantifiers.

The formulas of the predicate language $\mathcal{L}$ have meaning only when an *interpretation* is given for the symbols. We define the interpretation I in a set $U \neq \emptyset$ by interpreting predicate, functional symbols as a concrete relation, function defined in the universe U, and constants symbols as elements of the set U. The set U is called the *universe* of the *interpretation* I. These two items specify a *model structure* for $\mathcal{L}$. We write it as a pair $\mathbf{M} = (U, I)$.

Given a formula A of $\mathcal{L}$, and the *model structure* $\mathbf{M} = (U, I)$. Let's denote by A_I a statement written with logical symbols determined by the formula A and the interpretation I in the universe U. When A is a *closed formula*, it means it is a *sentence*, formula without free variables, A_I represents a proposition that is *true* or *false*. When A is not a sentence it contains free variables and may be *satisfied* (i.e. true) for some values in the universe U and *not satisfied* (i.e. false) for the others. Lets look at few simple examples.

Example 2.11

Let A be a formula $\exists x P(x, c)$ and consider a model structure $\mathbf{M}_1 = (N, I_1)$. The universe of the interpretation I_1 is the set N of natural numbers and we define I_1 as follows: we interpret the predicate P as relation $<$ and the constant c as number 5, i.e we put $P_{I_1} := and c_{I_1} : 5$.

The formula A: $\exists x P(x, c)$ under the interpretation I_1 becomes a mathematical statement $\exists x\ x < 0$ defined in the set N of natural numbers. We write it for short

$$A_{I_1} :\ \exists_{x \in N}\ x = 5.$$

A_{I_1} is obviously a true mathematical statement and say that the formula A: $\exists x P(x, c)$ is *true* under the interpretation I_1 in $\mathbf{M}_1$ or that A is *true* in $\mathbf{M}_1$. We write it symbolically as

$$\mathbf{M}_1\ \models\ \exists x P(x, c)$$

and say that $\mathbf{M}_1$ is a *model* for the formula A.

Example 2.12

Consider now a model structure $\mathbf{M}_2 = (N, I_2)$ and the formula A: $\exists x P(x, c)$. We interpret now the predicate P as relation $<$ in the set N of natural numbers and the constant c as number 0, i.e. we put $P_{I_2} :<$ and $c_{I_2} : 0$.

The formula A: $\exists x P(x, c)$ under the interpretation I_2 mathematical statement $\exists x\ x < 0$ defined in the set N of natural numbers. We write it for short

$$A_{I_2} :\ \exists_{x \in N}\ x < 0.$$

A_{I_2} is obviously a *false* mathematical statement. We say that the formula A: $\exists x P(x,c)$ is *false* under the interpretation I_2 in $\mathbf{M}_2$ or that A is *false* in $\mathbf{M}_2$. We write it symbolically as

$$\mathbf{M}_2 \not\models \exists x P(x,c)$$

and say that $\mathbf{M}_2$ is a *counter-model* for the formula A.

Example 2.13

Consider now a model structure $\mathbf{M}_3 = (Z, I_3)$ *and the formula A:* $\exists x P(x,c)$. *We define an interpretation* I_3 *in the set of all integers* Z *exactly as the interpretation* I_1*, i.e. we put* $P_{I_3} :<$ *and* $c_{I_3} :\ 0$.

In this case we get $A_{I_3} :\ \exists_{x \in Z}\ x < 0$ and A_{I_3} is obviously a true mathematical statement. The formula A is *true* under the interpretation I_3 in $\mathbf{M}_3$ (A is *satisfied, true* in $\mathbf{M}_3$). We write it symbolically as

$$\mathbf{M}_3 \models\ \exists x P(x,c).$$

$\mathbf{M}_3$ is yet another *model* for the formula A.

When a formula is not a closed (sentence) thing get more complicated. Given a model structure $\mathbf{M} = (U, I)$ a formula can be *satisfied* (i.e. true) for some values in the universe U and *not satisfied* (i.e. false) for the others.

Example 2.14

Consider the following formulas: 1. $A_1 : R(x,y)$, 2. $A_2 : \forall y R(x,y)$, 3. $A_3 : \exists x \forall y R(x,y)$. *We define a model structure* $\mathbf{M} = (N, I)$ *where R is interpreted as a relation* $\leq$ *defined in the set N of all natural numbers, i.e. we put* $R_I :\leq$.

In this case we get the following.
1. ${A_1}_I :\ x \leq y$ and $A_1 :\ R(x,y)$ is *satisfied* in $\mathbf{M} = (N, I)$ by all $n, m \in N$ such that $n \leq m$.
2. ${A_2}_I : \forall_{y \in N}\ x \leq y$ and $A_2 :\ \forall y R(x,y)$ is *satisfied* in $\mathbf{M} = (N, I)$ only by the natural number 0.
3. ${A_3}_I : \exists_{x \in N} \forall_{y \in N}\ x \leq y$ asserts that there is a smallest natural number and A_3 is a *true sentence* in $\mathbf{M} = (N, I)$, i.e. $\mathbf{M}$ is a *model* for A_3.

Observe that changing the universe of $\mathbf{M} = (N, I)$ to the set of all integers Z, we get a different a model structure $\mathbf{M}_1 = (Z, I)$. In this case ${A_3}_I : \exists_{x \in Z} \forall_{y \in Z}\ x \leq y$ asserts that there is a smallest integer and A_3 is a *false sentence* in $\mathbf{M}_1$, i.e. $\mathbf{M}_1$ is a *counter-model* for A_3.

We want predicate language *tautologies* to have the same property as the propositional, namely to be always true. In this case, we intuitively agree that it means that we want *predicate tautologies* to be formulas that are true under any interpretation in any possible universe.

A rigorous definition of the *predicate tautology* is provided in a later chapter on Predicate Logic. We construct it in the following steps.
1. We first define formally the notion of interpretation I of symbols of *calL* in a set $U \neq \emptyset$ i.e. the *model structure* $\mathbf{M} = (U, I)$ for the predicate language $\mathcal{L}$.
2. Then we define formally a notion " a formula A of $\mathcal{L}$ a is *true (valid)* in $\mathbf{M} = (U, I)$". We write it symbolically

$$\mathbf{M} \models A$$

and call the model structure $\mathbf{M} = (U, I)$ a *model* for A.
3. We define a notion *"A is a predicate tautology"* as follows.

Definition 2.8

For any formula A of predicate language $\mathcal{L}$,
A is a **predicate tautology (valid formula)** *if and only if* $\mathbf{M} \models A$ *for all model structures* $\mathbf{M} = (U, I)$ *for* $\mathcal{L}$*.*

4. We get immediately from the above definition 2.8 of a following definition of a notion *" A is not a predicate tautology"*.

Definition 2.9

For any formula A of predicate language $\mathcal{L}$,
A **is not** *a predicate tautology if and only if there is a model structure* $\mathbf{M} = (U, I)$ *for* $\mathcal{L}$*, such that* $\mathbf{M} \not\models A$*.*
We call such model structure $\mathbf{M}$ *a* **counter-model** *for A.*

The definition 2.9 says: to prove that A is not a predicate tautology one has to show a *counter- model*. It means one has to show a non-empty set U and define an interpretation I, such that we can prove that A_I is a false.

We use terms *predicate tautology* or *valid formula* instead of just saying a *tautology* in order to distinguish tautologies belonging to two very different languages. For the same reason we usually reserve the symbol $\models$ for propositional case. Sometimes symbols $\models_p$ or $\models_f$ are used to denote predicate tautologies, where "p" stands for "predicate" and "f" stands "first order". The predicate tautologies are also called *laws of quantifiers* and we will use both terms for them.

Here are some examples of predicate tautologies and counter models for formulas that are not tautologies.
For any formula $A(x)$ with a free variable x:

$$\models_p \ (\forall x \ A(x) \Rightarrow \exists x \ A(x)). \tag{2.53}$$

Observe that (2.53) represents an infinite number of formulas. It is a tautology for any formula $A(x)$ of $\mathcal{L}$ with a free variable x.
The inverse implication to (2.53) is not a predicate tautology.

$$\not\models_p \ (\exists x\ A(x) \Rightarrow \forall x\ A(x)) \tag{2.54}$$

To prove (2.54) we have to provide an example of a concrete formula $A(x)$ and construct a counter-model $\mathbf{M} = (U, I)$ for the formula $F : (\exists x\ A(x) \Rightarrow \forall x\ A(x))$. Let $A(x)$ be an atomic formula $P(x, c)$. We take as $\mathbf{M} = (N, I)$ for N set of natural numbers and $P_I :<,\ c_I : 3$. The formula F becomes an obviously *false* mathematical statement $F_I : (\exists_{n\in N} n < 3 \Rightarrow \forall_{n\in N} n < 3)$.

Observe that we have to be very careful when we deal with **quantifiers with restricted domain**. The most basic predicate tautology (2.53) fails when we use the quantifiers with restricted domain.

Example 2.15

Show that

$$\not\models_p \ (\forall_{B(x)}\ A(x) \Rightarrow \exists_{B(x)}\ A(x)). \tag{2.55}$$

Observe that (2.55) means that corresponding proper formula F of $\mathcal{L}$ obtained by the restricted quantifiers *transformations rules* (8.18), (8.19) is not a predicate tautology, i.e.

$$\not\models_p \ (\forall x(B(x) \Rightarrow A(x)) \Rightarrow \exists x(B(x) \cap A(x))). \tag{2.56}$$

We construct a *counter-model* $\mathbf{M}$ for (2.56) as follows. We take $\mathbf{M} = (N, I)$ where N is the set of real numbers, $B(x), A(x)$ are atomic formulas $Q(x, c), P(x, c)$ and the interpretation I is defined as $Q_I :<,\ P_I :>,\ c_I : 0$. The formula F of (2.56) becomes a mathematical statement

$$F_I : \quad (\forall_{n\in N}\ (n < 0 \Rightarrow n > 0) \Rightarrow \exists_{n\in N}(n < 0 \cap n > 0)).$$

F_I is a *false* because the statement $n < 0$ is false for all natural numbers and $F \Rightarrow B$ is a true implication for any logical value of B, so $\forall_{n\in N}\ (n < 0 \Rightarrow n > 0)$ is a true statement and $\exists_{n\in N}(n < 0 \cap n > 0)$ is obviously false.

Restricted quantifiers law corresponding to the predicate tautology (2.53) is:

$$\models_p \ (\forall_{B(x)}\ A(x) \Rightarrow (\exists x\ B(x) \Rightarrow \exists_{B(x)}\ A(x))). \tag{2.57}$$

We remind that (2.57) means that corresponding proper formula of $\mathcal{L}$ obtained by the restricted quantifiers *transformations rules* (8.18), (8.19) is a predicate tautology, i.e.

$$\models_p (\forall x(B(x) \Rightarrow A(x)) \Rightarrow (\exists x\ B(x) \Rightarrow \exists x\ (B(x) \cap A(x)))) \tag{2.58}$$

.

Another basic predicate tautology called a *dictum de omni law* is: For any formulas $A(x)$ with a free variable $x \in VAR$,

$$\models_p \ (\forall x \ A(x) \Rightarrow A(y)), \tag{2.59}$$

where $y \in VAR$ and A(y) is a result of substitution of y for all free occurrences of x in $A(x)$ (if any) and y is *free for* x in A(x), what means that no occurrence of a variable y becomes a bound occurrence in A(y). Restricted quantifiers law corresponding to the *dictum de omni* law (2.59) is:

$$\models_p \ (\forall_{B(x)} \ A(x) \Rightarrow (B(y) \Rightarrow A(y))), \tag{2.60}$$

where $y \in VAR$ satisfies the same condition as in (2.59).

Observe that we say A is restricted quantifiers law, or A is restricted quantifiers tautology as a shorthand to formally saying that a formula obtained from A by the *transformations rules* (8.18), (8.19) is a predicate tautology.

A more general version of (2.59) is:

$$\models_p \ (\forall x \ A(x) \Rightarrow A(t)), \tag{2.61}$$

where t is a term and A(t) is a result of substitution of t for all free occurrences of x in $A(x)$ and t is *free for* x in A(x), what means that no occurrence of a variable in t becomes a bound occurrence in A(t).

Here is another important tautology, called a *generalization law.*

$$\models_p \ (A(x) \Rightarrow \forall x \ A(x)). \tag{2.62}$$

The next important laws are the *Distributivity Laws.*

1. **Distributivity** of *existential quantifier* over *conjunction* holds only on one direction, namely the following is a predicate tautology.

$$\models_p \ (\exists x \ (A(x) \cap B(x)) \ \Rightarrow \ (\exists x A(x) \cap \exists x B(x))) \tag{2.63}$$

where $A(x), B(x)$ are any formulas with a free variable x. The inverse implication is not a predicate tautology, i.e. there are formulas $A(x), B(x)$ with a free variable x. such that

$$\not\models_p \ ((\exists x A(x) \cap \exists x B(x)) \Rightarrow \exists x \ (A(x) \cap B(x))). \tag{2.64}$$

To prove (2.64) means that we have to find a concrete formulas $A(x), B(x) \in \mathcal{F}$ and a model structure $\mathbf{M} = (U, I)$, where the interpretation I is the interpretation of all predicate, functional, and constant symbols in $A(x), B(x)$, such that it is a *counter- model* for the formula

$$F: \ ((\exists x A(x) \cap \exists x B(x)) \Rightarrow \exists x \ (A(x) \cap B(x))).$$

Take $\mathbf{M} = (R, I)$ where R is the set of real numbers, and $A(x), B(x)$ be atomic formulas $Q(x, c), P(x, c)$. We define the interpretation I as $Q_I :>, \ P_I :<, \ c_I : 0$. The formula F becomes an obviously *false* mathematical statement

$$F_I : ((\exists_{x \in R} \ x > 0 \cap \exists_{x \in R} \ x < 0) \Rightarrow \exists_{x \in R} \ (x > 0 \cap x < 0)).$$

2. **Distributivity** of *universal quantifier* over *disjunction* holds only on one direction, namely the following is a predicate tautology for any formulas $A(x), B(x)$ with a free variable x.

$$\models_p \ ((\forall x A(x) \cup \forall x B(x)) \Rightarrow \forall x \ (A(x) \cup B(x))). \tag{2.65}$$

The inverse implication *is not a predicate tautology*, i.e.there are formulas $A(x), B(x)$ with a free variable x. such that

$$\not\models_p \ (\forall x \ (A(x) \cup B(x)) \Rightarrow (\forall x A(x) \cup \forall x B(x))). \tag{2.66}$$

It means that we have to find a concrete formula $A(x), B(x) \in \mathcal{F}$ and a model structure $\mathbf{M} = (U, I)$ that is a *counter- model* for the formula

$$F: \ (\forall x \ (A(x) \cup B(x)) \Rightarrow (\forall x A(x) \cup \forall x B(x))).$$

Take $\mathbf{M} = (R, I)$ where R is the set of real numbers, and $A(x), B(x)$ be atomic formulas $Q(x, c), R(x, c)$. We define $Q_I :\geq, \ R_I :<, \ c_I : 0$. The formula F becomes an obviously *false* mathematical statement

$$F_I : (\forall_{x \in R} \ (x \geq 0 \cup x < 0) \Rightarrow (\forall_{x \in R} \ x \geq 0 \cup \forall_{x \in R} \ x < 0)).$$

The most frequently used laws of quantifiers have a form of a *logical equivalence*, symbolically written as $\equiv$. This not a new logical connective. This is a very useful symbol. It says that two formulas always have the same logical value, hence it can be used in the same way we use the equality symbol $=$. Formally we define it as follows.

Definition 2.10

For any formulas $A, B \in \mathcal{F}$ of the **predicate language** $\mathcal{L}$,

$$A \equiv B \quad \textit{if and only if} \quad \models_p \ (A \Leftrightarrow B).$$

We have also a similar definition for our *propositional language* $\mathcal{L}$ (definition 2.1) and propositional tautology (definition 2.3).

Equational Laws for Quantifiers

De Morgan
For any formula $A(x) \in \mathcal{F}$ with a free variable x,

$$\neg \forall x A(x) \equiv \exists x \neg A(x), \qquad \neg \exists x A(x) \equiv \forall x \neg A(x). \tag{2.67}$$

Definability
For any formula $A(x) \in \mathcal{F}$ with a free variable x,

$$\forall x A(x) \equiv \neg \exists x \neg A(x), \quad \exists x A(x) \equiv \neg \forall x \neg A(x). \tag{2.68}$$

Renaming the Variables
Let $A(x)$ be any formula with a free variable x and let y be a variable that *does not occur* in A(x). Let $A(x/y)$ be a result of *replacement* of each occurrence of x by y, then the following holds.

$$\forall x A(x) \equiv \forall y A(y), \quad \exists x A(x) \equiv \exists y A(y). \tag{2.69}$$

Alternations of Quantifiers
Let $A(x, y)$ be any formula with a free variables x and y.

$$\forall x \forall y\ (A(x, y)\ \equiv\ \forall y \forall x\ (A(x, y) \tag{2.70}$$

$$\exists x \exists y\ (A(x, y)\ \equiv\ \exists y \exists x\ (A(x, y) \tag{2.71}$$

Introduction and Elimination Laws
If B is a formula such that B *does not contain any free occurrence* of x, then the following logical equivalences hold.

$$\forall x(A(x) \cup B) \equiv (\forall x A(x) \cup B), \tag{2.72}$$

$$\exists x(A(x) \cup B) \equiv (\exists x A(x) \cup B), \tag{2.73}$$

$$\forall x(A(x) \cap B) \equiv (\forall x A(x) \cap B), \tag{2.74}$$

$$\exists x(A(x) \cap B) \equiv (\exists x A(x) \cap B), \tag{2.75}$$

$$\forall x(A(x) \Rightarrow B) \equiv (\exists x A(x) \Rightarrow B), \tag{2.76}$$

$$\exists x(A(x) \Rightarrow B) \equiv (\forall x A(x) \Rightarrow B), \tag{2.77}$$

$$\forall x(B \Rightarrow A(x)) \equiv (B \Rightarrow \forall x A(x)), \tag{2.78}$$

$$\exists x(B \Rightarrow A(x)) \equiv (B \Rightarrow \exists x A(x)). \tag{2.79}$$

Distributivity Laws

Let $A(x), B(x)$ be any formulas with a free variable x.
Distributivity of **universal quantifier** over **conjunction**.

$$\forall x\ (A(x) \cap B(x))\ \equiv\ (\forall x A(x) \cap \forall x B(x)) \tag{2.80}$$

Distributivity of **existential quantifier** over **disjunction**.

$$\exists x\ (A(x) \cup B(x))\ \equiv\ (\exists x A(x) \cup \exists x B(x)) \tag{2.81}$$

We also define the notion of logical equivalence $\equiv$ for the formulas of the —textitpropositional language (definition 2.1) and its semantics.

Definition 2.11

For any formulas $A, B \in \mathcal{F}$ of the **propositional language $\mathcal{L}$**,

$$A \equiv B \quad \textit{if and only if} \quad \models\ (A \Leftrightarrow B).$$

Moreover, we prove that any substitution of propositional tautology by a formulas of the predicate language is a predicate language tautology. The same holds for the logical equivalence. In particular, we transform the propositional *Implication* and *Double Negation* tautologies (2.48), (2.49) into the following predicate equivalences.

For any formulas A, B of the **predicate language** $\mathcal{L}$,

$$(A \Rightarrow B) \equiv (\neg A \cup B), \tag{2.82}$$

$$\neg\neg A \equiv A \tag{2.83}$$

We use (2.82) and (2.83) to prove the following De Morgan Laws for *restricted quantifiers.*

Restricted De Morgan
For any formulas $A(x), B(x) \in \mathcal{F}$ with a free variable x,

$$\neg\forall_{B(x)}\ A(x) \equiv \exists_{B(x)}\ \neg A(x), \qquad \neg\exists_{B(x)}\ A(x) \equiv \forall_{B(x)}\neg A(x). \tag{2.84}$$

Here is a poof of first equality. The proof of the second one is similar and is left as an exercise.

$$\neg\forall_{B(x)}\ A(x) \equiv \neg\forall x\ (B(x) \Rightarrow A(x)) \equiv \neg\forall x\ (\neg B(x) \cup A(x)) \equiv \exists x\ \neg(\neg B(x) \cup A(x))$$
$$\equiv \exists x\ (\neg\neg B(x) \cap \neg A(x)) \equiv \exists x\ (B(x) \cap \neg A(x)) \equiv \exists_{B(x)}\ \neg A(x)).$$

We also transform the propositional *Distributivity* tautologies (2.45), (2.46) into the following predicate equivalences.

For any formulas A, B of the **predicate language** $\mathcal{L}$,

$$(A \cap (B \cup C)) \equiv ((A \cap B) \cup (A \cap C)), \tag{2.85}$$

$$(A \cup (B \cap C)) \equiv ((A \cup B) \cap (A \cup C)) \tag{2.86}$$

We use (2.85) and (2.86) to prove the following Distributivity Laws for *restricted quantifiers.*

Restricted Distributivity Laws
We generalize the *Introduction and Elimination Laws* (10.32), (11.45), (11.46), (10.38) to the case of the the restricted quantifiers as folows.

Restricted Introduction and Elimination Laws

If B is a formula such that B *does not contain any free occurrence* of x, then the following logical equivalences hold for any formulas $A(x), B(x), C(x)$.

$$\forall_{C(x)}(A(x) \cup B) \equiv (\forall_{C(x)}A(x) \cup B), \tag{2.87}$$

$$\exists_{C(x)}\ (A(x) \cap B) \equiv (\exists_{C(x)}\ A(x) \cap B), \tag{2.88}$$

$$\forall_{C(x)}(A(x) \Rightarrow B) \equiv (\exists_{C(x)} A(x) \Rightarrow B), \tag{2.89}$$

$$\forall_{C(x)}(B \Rightarrow A(x)) \equiv (B \Rightarrow \forall_{C(x)} A(x)). \tag{2.90}$$

The proofs are similar to the proof of the restricted de Morgan Laws.
The similar generalization of the other *Introduction and Elimination Laws* (11.43), (11.44), (11.47), (10.39) fails. We can easily follow Example 2.15 and construct proper counter-models proving the following.

$$\exists_{C(x)}(A(x) \cup B) \not\equiv (\exists_{C(x)} A(x) \cup B),$$

$$\forall_{C(x)}(A(x) \cap B) \not\equiv (\forall_{C(x)} A(x) \cap B),$$

$$\exists_{C(x)}(A(x) \Rightarrow B) \not\equiv (\forall_{C(x)} A(x) \Rightarrow B),$$

$$\exists_{C(x)}(B \Rightarrow A(x)) \not\equiv (B \Rightarrow \exists x A(x)).$$

Nevertheless it is possible to correctly generalize them all as to cover quantifiers with restricted domain. We show it in a case of (11.43) and leave the other cases to the reader as an exercise.

Example 2.16

The restricted quantifiers version of (11.43) is the following.

$$\exists_{C(x)}(A(x) \cup B) \equiv (\exists_{C(x)} A(x) \cup (\exists x\ C(x) \cap B)). \tag{2.91}$$

We derive (8.74) as follows.

$$\exists_{C(x)}(A(x) \cup B) \equiv \exists x(C(x) \cap (A(x) \cup B)) \equiv \exists x((C(x) \cap A(x)) \cup (C(x) \cap B))$$

$$\equiv (\exists x(C(x) \cap A(x)) \cup \exists x(C(x) \cap B)) \equiv (\exists_{C(x)} A(x) \cup (\exists x\ C(x) \cap B)).$$

We leave it as an exercise to specify and write references to transformation or equational laws used at each step of our computation.

2.6 Homework Problems

Propositional Languages

1. For the following sentences write their corresponding formulas.

   (a) If Mr. Smith is happy, Mrs. Smith is not happy, and if If Mr. Smith is not happy, Mrs. Smith is not happy.

(b) If John doesn't know logic, then if he knows logic, he was born in the 12th century.

(c) If from the fact that all sides of a triangle ABC are equal we can deduce that all angles of the triangle ABC are equal and all angles of the triangle ABC are not equal, then all sides of a triangle ABC are equal.

(d) If it is not the fact that a line L is parallel to a line M or a line P is not parallel the line M, then the line L is not parallel to the line M or the line P is parallel the line M.

(e) If a number a is divisible by 3 and by 5, then from the fact that it is not divisible by 3, we can deduce that it is also not divisible by 5.

2. For each of the following formulas write 3 corresponding natural language sentences.

   (a) $(a \Rightarrow (\neg a \cap b))$

   (b) $(((p \cup q) \cap \neg p) \Rightarrow q)$

   (c) $((a \Rightarrow b) \Rightarrow (a \Rightarrow (b \cup c)))$

   (d) $\neg(p \cap (\neg p \cap q))$

   (e) $((a \Rightarrow ((\neg b \cap b) \Rightarrow c))$

3. Consider a following set $\mathcal{S}$

$$\mathcal{S} = \{(a \cap b) \Rightarrow \neg(a \cup b),\ ((\neg a) \Rightarrow (\neg a \Rightarrow b)),\ (\neg a \Rightarrow (a \cap \neg b))\}.$$

   1. Determine which of the elements of $\mathcal{S}$ are, and which are not well formed formulas (wff) of $\mathcal{L} = (\mathcal{A}, \mathcal{F})$.

   2. If $A \in \mathcal{S}$ is not a formula, i.e if $A \notin \mathcal{F}$ re-write it as a correct formula and write in the natural language what it says.

4. Write a full definition of a propositional language that uses Hilbert set of connectives. Give four examples of well form formulas of this language. List next to them corresponding formulas of our propositional language $\mathcal{L}$.

5. Write a full definition of a propositional language $\mathcal{L}$ that uses Łukasiewicz set of connectives. Give 4 examples of well form formulas of this language. Give 4 examples of well form formulas of this language. List next to them corresponding formulas of our propositional language $\mathcal{L}$.

Propositional Semantics

1. Given a formula A: $(((a \cap b) \cup \neg c) \Rightarrow b)$. Evaluate the logical value of A for the following sets of logical values of its basic components, i.e. variables a, b: 1. a=T, b=F, c=F and 2. a=F, b=T, c=T.

2. Given a formula A: $(((a \Rightarrow \neg b) \cup b) \Rightarrow a)$. Evaluate the logical value of A for all possible logical values of its variables.

3. Given a formula A: $(((a \downarrow \neg b) \cup b) \uparrow a)$. Evaluate the logical value of A for the following sets of logical values of its variables: 1. a=T, b=F and 2. a=F, b=F.

4. Find and prove an equality defining implication in terms of disjunction and negation.

5. Find and prove an equality defining conjunction in terms of disjunction and negation.

6. Find and prove an equality and a table defining conjunction in terms of implication and negation.

7. Prove that $\cup$ can be defined in terms of $\Rightarrow$ alone.

8. Find and prove an equality defining $\Rightarrow$ in terms of $\uparrow$.

9. Define $\Rightarrow$ in terms of $\neg$ and $\cap$.

10. Find an equality defining $\Rightarrow$ in terms of $\downarrow$.

11. Define $\cap$ in terms of $\Rightarrow$ and $\neg$.

12. Find an equality defining $\cap$ in terms of $\downarrow$ alone.

Propositional Tautologies

1. Prove 5 propositional tautologies of your choice.

2. Prove that a formula $(((\neg A \Rightarrow B) \cap (B \Rightarrow C)) \Rightarrow ((A \cap B) \Rightarrow C)$ is not a propositional tautology.

3. Show that "If a number is divisible by 3 and by 5, then from the fact that it is not divisible by 3, we can deduce that it is also not divisible by 5" is always a true statement.

4. Determine whether the following arguments *logically correct* by representing each sentence as propositional formula and checking whether the conclusion is logically implied by the conjunction of the assumptions. To do this assign logical value T to each formula representing assumption and F to the formula representing the conclusion, and determine whether a *contradiction* results.

   (a) If John is a Communist, John is atheist. John is an atheist. Hence John is a Communist.

(b) If the temperature and air pressure remained constant, there was a rain.The temperature did remain constant. Therefore, if there was a rain then the air pressure did not remain constant.

(c) If $a = 0$ or $b = 0$, then $ab = 0$. But $ab \neq 0$. Hence $a \neq 0$ or $b \neq 0$.

(d) If $a = 0$ and $b = 0$, then $ab = 0$. But $ab \neq 0$. Hence $a \neq 0$ or $b \neq 0$.

Predicate Language Description and Application to AI

1. Given the following formulas $A_1 - A_5$ of $\mathcal{L}$.

$$A_1 = R(x, y, g(c, x)),\ A_2 = \exists x P(x, f(x, y)),\ A_3 = \exists d R(x, y, g(c, d)),$$

$$A_4 = \forall z(f(x, P(c, y)),\quad A_5 = \exists y P(x, f(c, y)) \cup \forall y P(x, f(c, y)).$$

(a) Indicate whether they are, or are not well formed formulas of $\mathcal{F}$. For those which are not in $\mathcal{F}$ write a correct formula.

(b) For each correct, or corrected formula identify all components: connectives, quantifiers, predicate and function symbols, and list all its terms.

(c) For each formula identify its s free and bound variables. State which are open and which are closed formulas (sentences), if any.

2. For the following mathematical statements write their corresponding formulas of predicate language $\mathcal{L}$.

(a) $\forall_{n>1}(n + 3 < 8 \cup \exists_{x \in R}\ x + n > 8)$

(b) $\forall_{x \in R}\ \exists_{n \in N}(x + n > 0 \Rightarrow \exists_{m \in N}(m = x + n))$

(c) If all natural numbers are smaller then zero, then the sum of any two integers is smaller then zero.

(d) For all natural numbers The following implication holds for all natural numbers: if $n > 0$, then there is a real number x, such that $n + x = 0$ or there is an integer m, such that $m > 0$.

3. For the following natural language statements write their corresponding formulas of predicate language $\mathcal{L}$.

(a) Anyone who is persistent can learn logic.

(b) Some people are witty only if they are drunk.

(c) John hates everybody who does not hate himself.

(d) Everybody loves somebody and no one loves everybody.

4. For the following natural language statements write their corresponding formulas of AI language of your choice.

(a) Anyone who is lazy can't learn logic.

(b) Some people are happy only if they sing.

(c) John likes everybody who does not like Mary.

(d) Everybody with green eyes likes John.

5. For each of the following formulas (some with restricted quantifiers) write two corresponding natural language sentences.

(a) $\forall x(P(x) \Rightarrow \exists y Q(x,y))$.

(b) $\forall x \exists y(P(x) \cap \neg Q(x,y))$.

(c) $\forall_{A(x)} \exists_{A(y)} B(y)$.

(d) $\exists_{P(x)} \forall_{N(x)} R(x,y)$.

Predicate Semantics

1. For each of the formulas and each model structure **M** indicate for what values the formula is *satisfied* (if it contains free variables) or whether **M** is its *model* or *counter-model* (if it is a closed formula. i.e. a sentence).

Formulas are:

(a) $P(f(x,y),c)$

(b) $P(x,y) \Rightarrow P(y,x)$

(c) $\forall x \forall y \forall z((P(x,y) \cap P(y,z)) \Rightarrow P(x,z))$

Model structures **M** are:

$\mathbf{M}_1 = (N, I)$, for N set of natural numbers and $P_I :\geq, f_I$: multiplication, and $c_I : 2$

$\mathbf{M}_2 = (Z, I)$, for Z set of integers and $P_I :=,\ f_I : +$, and $c_I : 2$

$\mathbf{M}_3 = (2^Z, I)$, for 2^Z the set of all subsets of Integers, and $P_I : \subseteq,\ f_I : \cap$, and $c_I : \emptyset$

2. For a given model structure **M** and corresponding closed formulas determine for each of them whether **M** is its *model* or a *counter-model*.

(a) Model structure is $\mathbf{M} = (N, I)$, for N set of natural numbers and $P_I :=,\ g_I :\ +,\ f_I$: multiplication, and $c_I : 0,\ d_I : 1$.

Formulas are:

$A_1 :\ \forall x \exists y(P(x, g(y,y)) \cup P(x, g(g(y,y),d)))$

$A_2 :\ \forall x \forall y(P(f(x,y),c) \Rightarrow (P(x,c) \cup P(y,c)))$

$A_3 : \exists y\ P(g(y,y),d)$

(b) Model structure is $\mathbf{M} = (Z, I)$, for Z set of integers and $P_I :=,\ f_I : +$,

Formulas are:

$A_1 : \ \forall x \forall y\ P(f(x,y), f(y,x)))$

$A_2 : \ \forall x \forall y\ P(f(x,y), y)$

3. Prove that the following formulas are not predicate tautologies, i.e. find for each of them a *counter-model* $\mathbf{M}$.

   (a) $(\exists x\ A(x) \Rightarrow \forall x\ A(x))$

   (b) $(\forall x \exists y\ A(x,y) \Rightarrow \exists x \forall y\ A(x,y))$

   (c) $(\exists x \exists y\ A(x,y) \Rightarrow \exists y\ A(y,y))$

   (b) $(\forall x \exists y\ A(x,y) \Rightarrow \exists y\ A(y,y))$

   (d) $(\exists x\ (A(x) \Rightarrow\ B(x)) \Rightarrow (\exists x\ A(x) \Rightarrow \exists x\ B(x)))$

   (e) $(\exists x\ (A(x) \Rightarrow\ B(x)) \Rightarrow (\exists x\ A(x) \Rightarrow \exists x\ B(x)))$

4. Transform the following formulas with restricted quantifiers into a proper formulas of the predicate language $\mathcal{L}$.

   (a) $(\forall_{A(x)} \exists_{B(x)} C(x) \Rightarrow \neg \exists_{B(x)} \neg C(x))$

   (b) $(\exists_{A(x)} (\forall_{B(y)} C(y) \Rightarrow \neg C(x))$

   (c) $(\forall_{A(y)} \exists_{B(x)} D(x,y) \Rightarrow \neg \exists_{B(x)} C(x))$

   (d) $\forall_{A(x)} (\exists_{B(x)} C(x) \cup \neg \forall_{A(x)} C(x))$

5. Use proper Equational Laws for Quantifiers to prove that the following *Restricted Introduction and Elimination Laws* hold for any formulas A(x), B(x), C(x), and B, where B *does not contain any free occurrence* of x.

   (a) $\exists_{C(x)}\ (A(x) \cap B) \equiv (\exists_{C(x)}\ A(x) \cap B)$

   (b) $\forall_{C(x)} (A(x) \Rightarrow B) \equiv (\exists_{C(x)} A(x) \Rightarrow B)$

   (c) $\forall_{C(x)} (B \Rightarrow A(x)) \equiv (B \Rightarrow \forall_{C(x)} A(x))$

Chapter 3

Propositional Semantics: Classical and Many Valued

3.1 Formal Propositional Languages

We define here a general notion of a propositional language. We obtain, as specific cases, various languages for propositional classical logic as well as languages for many non-classical logics.

We assume that any propositional language contains a countably infinite set VAR of propositional variables. What distinguishes one propositional language from the other is the choice of its set CON of *propositional connectives.* We adopt a notation $\mathcal{L}_{CON}$ for a *propositional language* with the set CON of logical connectives. For example, the language $\mathcal{L}_{\{\neg\}}$ denotes a propositional language with only one connective $\neg$. The language $\mathcal{L}_{\{\neg,\Rightarrow\}}$ denotes that the language has only two connectives $\neg$ and $\Rightarrow$ adopted as propositional connectives. All propositional languages share the general way their sets of *formulas* are formed.

Theoretically one can use any symbols to denote propositional connectives. But there are some preferences, as connectives have a long history and intuitive meaning. The formal meaning, i.e. a *semantics* for a given language is discussed and defined in the next section.

Different semantics can share the same language. For example, the language $\mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow\}}$ is used as a propositional language for classical logic semantics, intuitionistic logic semantics, and many valued logics semantics. It is also possible for several languages to share the same semantics. The classical propositional logic is the best example of such situation. We will prove in the section 3.4 that the languages:

$$\mathcal{L}_{\{\neg\Rightarrow\}}, \mathcal{L}_{\{\neg\cap\}}, \mathcal{L}_{\{\neg\cup\}}, \mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow\}}, \mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow,\Leftrightarrow\}},$$

and even two languages with only one binary propositional connectives, denoted usually by $\uparrow$ and $\downarrow$, respectively, i.e languages $\mathcal{L}_{\{\uparrow\}}, \mathcal{L}_{\{\downarrow\}}$ all share the same semantics characteristic for the classical propositional logic.

The connectives have well established symbols and names, even if their semantics can differ. We use names *negation, conjunction, disjunction*, *implication* and *equivalence (biconditional)* for $\neg, \cap, \cup, \Rightarrow, \Leftrightarrow$, respectively. The connective $\uparrow$ is called *alternative negation* and $A \uparrow B$ reads: *not both A and B.* The connective $\downarrow$ is called *joint negation* and $A \downarrow B$ reads: *neither A nor B.*

Other most common propositional connectives are probably modal connectives of *possibility* and *necessity.* Standard modal symbols are $\Box$ for *necessity* and $\Diamond$ for *possibility.* We will also use symbols **C** and **I** for modal connectives of possibility and necessity, respectively.

A formula $\mathbf{C}A$, or $\Diamond A$ reads: *it is possible that A* , *A is possible*, and a formula $\mathbf{I}A$, or $\Box A$ reads: *it is necessary that A*, *A is necessary.*

A motivation for notation **C** and **I** arises from topological semantics for modal S4 and S5 logics. **C** becomes equivalent to a set closure operation, and **I** becomes equivalent to a set interior operation.

The symbols $\Diamond$, **C** and $\Box$, **I** are not the only symbols used for modal connectives. Other symbols include N for necessity and P for possibility. There is also a variety of modal logics created by computer scientists, all with their set of symbols and motivations for their use and their semantics. The modal logics *extend* the classical logic and hence their language is for example $\mathcal{L}_{\{\Box, \Diamond, \neg, \cap, \cup, \Rightarrow\}}$.

Knowledge logics also extend the classical logic. Their languages add to the classical connectives a new *knowledge* connective, often denoted denoted by K. The formula KA reads: *it is known that A* , *A is known.* The language of a knowledge logic is for example $\mathcal{L}_{\{K,\ \neg, \cap, \cup, \Rightarrow\}}$.

Autoepistemic logics use a *believe* connective, often denoted by B. The formula BA reads: *it is believed that A.* They also extend the classical logic and hence their language is $\mathcal{L}_{\{B,\ \neg, \cap, \cup, \Rightarrow\}}$.

Temporal logics add temporal connectives to the set of classical propositional connectives. For example some of them use connectives (operators, as they are often called) F, P, G, and H to denote the following intuitive readings. FA reads *A is true at some future time*, PA reads *A was true at some past time*, GA reads *A will be true at all future times*, and HA reads *A has always been true in the past.* In order to take account of this variation of truth-values over time, some formal semantics were created, and many more will be created.

It is possible to create connectives with more then one or two arguments, but we allow here only one and two argument connectives, as logics which will be discussed here use only those two kind of connectives.

We adopt the following definition, common to all propositional languages considered in our propositional logics investigations.

Definition 3.1 (Propositional Language)

By a propositional language with a set CON of **propositional connectives** *we understand a pair*

$$\mathcal{L}_{CON} = (\mathcal{A}, \mathcal{F}). \tag{3.1}$$

$\mathcal{A}$ *is a called* **propositional alphabet** *and* $\mathcal{F}$ *is called a set of* **propositional formulas** *of the language* $\mathcal{L}_{CON}$*. The alphabet* $\mathcal{A}$*, the set CON of propositional connectives, and the set* $\mathcal{F}$ *of propositional formulas are defined as follows.*

1. Alphabet $\mathcal{A}$

The alphabet $\mathcal{A} = VAR \cup CON \cup PAR$, where VAR, CON, PAR are all disjoint sets and VAR, CON are non-empty sets. VAR is countably infinite and is called a set of **propositional variables**; we denote elements of VAR by $a, b, c, ...$ etc, (with indices if necessary).

CON is a finite set of **propositional connectives**, PAR is a set of **auxiliary symbols**. We assume that $PAR \neq \emptyset$ and contains two elements $(,)$ called parentheses, i.e. $PAR = \{(,)\}$. The set PAR may be empty, for example of a case of Polish notation, but we assume that it contains two parenthesis as to make the reading of formulas more natural and uniform.

2. Propositional connectives CON

We assume that the set CON is non empty and finite. We specify it for specific cases (specific logics). It is possible to consider languages with connectives which have more then one or two arguments, nevertheless we restrict ourselves to languages with one or two argument connectives only.
We assume that

$$CON = C_1 \cup C_2$$

where C_1 is a finite set (possibly empty) of **unary connectives**, C_2 is a finite set (possibly empty) of **binary connectives** of the language $\mathcal{L}_{CON}$.

2. Set $\mathcal{F}$ of formulas

The set $\mathcal{F}$ is built recursively from the elements of the alphabet $\mathcal{A}$, i.e. $\mathcal{F} \subseteq \mathcal{A}^*$, where $\mathcal{A}^*$ is the set of all finite sequences (strings) form from elements of $\mathcal{A}$ and is defined as follows.

The set $\mathcal{F}$ of all **formulas** of a propositional language $\mathcal{L}_{CON}$ is **the smallest set**, such that the following conditions hold:

(1) $VAR \subseteq \mathcal{F}$;

(2) if $A \in \mathcal{F}$, $\bigtriangledown \in C_1$ i.e $\bigtriangledown$ is an one argument connective, then $\bigtriangledown A \in \mathcal{F}$;

(3) if $A, B \in \mathcal{F}$, $\circ \in C_2$ i.e $\circ$ is a two argument connective, then $(A \circ B) \in \mathcal{F}$.

The elements of the set $VAR \subseteq \mathcal{F}$ are called **atomic formulas**. The set $\mathcal{F}$ is also called a set of all **well formed formulas** (wff) of the language $\mathcal{L}_{CON}$.

The alphabet $\mathcal{A}$ is *countably infinite* and consequently the set $\mathcal{A}^*$ of all finite sequences of elements of $\mathcal{A}$ is also countably infinite. By definition, $\mathcal{F} \subseteq \mathcal{A}^*$, hence the set $\mathcal{F}$ is also countably infinite. We state as separate fact.

Fact 3.1 *For any propositional language $\mathcal{L}_{CON} = (\mathcal{A}, \mathcal{F})$, the set $\mathcal{F}$ of formulas is countably infinite. We hence consider here only* **infinitely countable languages**.

Observation 3.1

When defining a language $\mathcal{L}_{CON}$ we choose not only the propositional connectives but also the symbols denoting them.

For example, $\mathcal{L}_1 = \mathcal{L}_{\{\neg\}}$ and $\mathcal{L}_2 = \mathcal{L}_{\{\sim\}}$ are two different propositional languages both with negation as the only connective.

The choice of appropriate well established symbols for logical connectives depends on a personal preferences of books' authors and creators of different logics. One can find a variety of them in the literature. We presented some historical choices in the chapter 2.

Example 3.1

Let $\mathcal{L}_1 = \mathcal{L}_{\{\neg\}}$ and $\mathcal{L}_2 = \mathcal{L}_{\{\sim\}}$. The formulas of both languages $\mathcal{L}_1$, $\mathcal{L}_2$ are propositional variables or multiple negations of of a propositional variable.

The strings $a, \neg b, \neg\neg b, \neg\neg\neg a$ are well formed formulas of $\mathcal{L}_1$. The corresponding formulas of $\mathcal{L}_2$ are $a, \sim b, \sim\sim b, \sim\sim\sim a$.

Observe that the strings $(\neg a), \neg, \neg(\neg a), \neg(a), (\sim a), \neg, \sim (\sim a) \sim (a)$ are not well formed formulas of neither of the languages $\mathcal{L}_1, \mathcal{L}_2$.

We adopt the general definition of the set $\mathcal{F}$ of formulas of $\mathcal{L}_{CON}$ to for example the language $\mathcal{L}_{\{\sim\}}$ as follows.

Example 3.2

The set $\mathcal{F}$ of all **formulas** *of a propositional language $\mathcal{L}_{\{\sim\}}$ is the smallest set, such that the following conditions hold:*

(1) *$VAR \subseteq \mathcal{F}$ (atomic formulas);*

(2) *if $A \in \mathcal{F}$, then $\sim A \in \mathcal{F}$.*

Example 3.3

Consider now $\mathcal{L}_{CON}$ for the set of connectives $CON = \{\neg\} \cup \{\Rightarrow\}$, where $\neg \in C_1$ and $\Rightarrow \in C_2$. It means that we defined a language $\mathcal{L} = \mathcal{L}_{\{\neg, \Rightarrow\}}$.

By the initial recursive step we get for any $a \in VAR$, $a \in \mathcal{F}$. By the recursive step and its repetition we get for example that $\neg a \in \mathcal{F}$, $\neg\neg a \in \mathcal{F}$, $\neg\neg\neg a \in \mathcal{F}$, ... etc., i.e. get all formulas from the the example 5.10 language $\mathcal{L}_1$. But also we also get that $(a \Rightarrow a)$, $(a \Rightarrow b), \neg(a \Rightarrow b), (\neg a \Rightarrow b), \neg((a \Rightarrow a) \Rightarrow \neg(a \Rightarrow b))$.... etc. are all in $\mathcal{F}$ and infinitely many others.

Observe that $(\neg(a \Rightarrow b))), a \Rightarrow b, (a \Rightarrow)$ are not in $\mathcal{F}$.

Example 3.4

Consider $\mathcal{L} = \mathcal{L}_{CON}$ for $C_1 = \{\neg, P, N\}$, $C_2 = \{\Rightarrow\}$. If we understand P, N as a possibility and necessity connectives, the obtained language is called a modal language with only negation as non-modal connective.

The set of formulas $\mathcal{F}$ of $\mathcal{L}$ contains all formulas from example 5.11, but also formulas $Na, \neg Pa, P\neg a, (N\neg b \Rightarrow Pa), \neg P\neg a, ((N\neg b \Rightarrow Pa) \Rightarrow b)$, ... etc.

We adopt the general definition of the set $\mathcal{F}$ of formulas of $\mathcal{L}_{CON}$ to for example the *modal language* $\mathcal{L}_{\{\neg, P, N, \Rightarrow\}}$ as follows.

Example 3.5

The set $\mathcal{F}$ of all **formulas** *of a propositional language $\mathcal{L}_{\{\neg, P, N, \Rightarrow\}}$ is the smallest set, such that the following conditions hold:*

(1) *$VAR \subseteq \mathcal{F}$ (atomic formulas);*

(2) *if $A \in \mathcal{F}$, then $\neg A$, PA, NA $\in \mathcal{F}$;*

(3) *if $A, B \in \mathcal{F}$, then $(A \Rightarrow B) \in \mathcal{F}$.*

We introduce now formal definitions of basic syntactical notions of a main connective, a sub-formula of a given formula, and of a degree of a given formula.

Definition 3.2 (Main Connective)

Given a language $\mathcal{L}_{CON} = (\mathcal{A}, \mathcal{F})$.

For any connectives $\bigtriangledown \in C_1$ *and* $\circ \in C_2$,

$\bigtriangledown$ *is called* **a main connective** *of* $\bigtriangledown A \in \mathcal{F}$ *and*

$\circ$ *is* **a main connective** *of* $(B \circ C) \in \mathcal{F}$.

Observe that it follows directly from the definition of the set of formulas that for any formula $C \in \mathcal{F}$, exactly one of the following holds: C is atomic, or there is a unique formula A and a unique unary connective $\bigtriangledown \in C_1$, such that C is of the form $\bigtriangledown A$, or here are unique formulas A and B and a unique binary connective $\circ \in C_2$, such that C is $(A \circ B)$. We have hence proved the following.

Observation 3.2

For any formula $A \in \mathcal{F}$, *A is atomic or has a unique main connective.*

Example 3.6

The main connective of $(a \Rightarrow \neg Nb)$ *is* $\Rightarrow$. *The main connective of* $N(a \Rightarrow \neg b)$ *is* N. *The main connective of* $\neg(a \Rightarrow \neg b)$ *is* $\neg$ *The main connective of* $(\neg a \cup \neg(a \Rightarrow b))$ *is* $\cup$.

Definition 3.3

We define a notion of direct **a direct sub-formula** *as follows: 1. Atomic formulas have no direct sub-formulas. 2. A is a direct sub-formula of a formula* $\bigtriangledown A$, *where* $\bigtriangledown$ *is any unary connective. 3.* A, B *are direct sub-formulas of a formula* $(A \circ B)$ *where* $\circ$ *is any binary connective.*

Directly from the definition 3.3 we get the following.

Observation 3.3

For any formula A, A is atomic or has exactly one or two direct sub-formulas depending on its main connective being unary or binary, respectively.

Example 3.7

The formula $(\neg a \cup \neg(a \Rightarrow b))$ *has exactly* $\neg a$ *and* $\neg(a \Rightarrow b)$ *as direct sub-formulas.*

Definition 3.4

We define a notion of a **sub-formula** *of a given formula in two steps. 1. For any formulas A and B, A is a* **proper sub-formula** *of B if there is sequence of formulas, beginning with A, ending with B, and in which each term is a direct sub-formula of the next. 2. A* **sub-formula** *of a given formula A is any proper sub-formula of A, or A itself.*

The formula $(\neg a \cup \neg(a \Rightarrow b))$ has $\neg a$ and $\neg(a \Rightarrow b)$ as direct sub-formula. The formulas $\neg a$ and $\neg(a \Rightarrow b)$ have a and $(a \Rightarrow b)$ as their direct sub-formulas, respectively. The formulas $\neg a$, $\neg(a \Rightarrow b)$, a and $(a \Rightarrow b)$ are all proper sub-formulas of the formula $(\neg a \cup \neg(a \Rightarrow b))$ itself. Atomic formulas a and b are direct sub-formulas of $(a \Rightarrow b)$. Atomic formula b is a proper sub-formula of $\neg b$.

Example 3.8

The set of all sub-formulas of

$$(\neg a \cup \neg(a \Rightarrow b))$$

consists of $(\neg a \cup \neg(a \Rightarrow b))$, $\neg a$, $\neg(a \Rightarrow b)$, $(a \Rightarrow b)$, a *and* b.

Definition 3.5 (Degree of a formula)

By a degree of a formula we mean the number of occurrences of logical connectives in the formula.

The degree of $(\neg a \cup \neg(a \Rightarrow b))$ is 4. The degree of $\neg(a \Rightarrow b))$ is 2. The degree of $\neg a$ is 1. The degree of a is 0.

Note that the degree of any proper sub-formula of A must be one less than the degree of A. This is the central fact upon mathematical induction arguments are based. Proofs of properties formulas are usually carried by mathematical induction on their degrees.

Example 3.9

Given a formula $A : (\neg\mathbf{I}\neg a \Rightarrow (\neg\mathbf{C}a \cup (\mathbf{I}a \Rightarrow \neg\mathbf{I}b)))$.
1. The language to which A *belongs is a modal language* $\mathcal{L}_{\{\neg,\mathbf{C},\mathbf{C},\cup,\cap,\Rightarrow\}}$ *with the possibility connective* $\mathbf{C}$ *and necessity connective* $\mathbf{C}$*. Both of them are one argument connectives.*
2. The main connective of A *is* $\Rightarrow$*, the degree of* A *is 11.*
3. All sub-formulas of A *of the degree 0 are the atomic formulas* a, b*. All sub-formulas of* A *of the degree 1 are:* $\neg a$, $\mathbf{C}a$, $\mathbf{I}a$, $\mathbf{I}b$.

Languages with Propositional Constants

A propositional language $\mathcal{L}_{CON} = (\mathcal{A}, \mathcal{F})$ is called a language with propositional constants, when we distinguish certain constants, like symbol of truth T or falsehood F, or other symbols as elements of the alphabet. The propositional constants are zero-argument connectives. In this case the set CON of logical connectives contains a a finite, non empty set of *zero argument connectives* C_0, called **propositional constants**, i.e. we put

$$CON = C_0 \cup C_1 \cup C_2.$$

The definition of the set $\mathcal{F}$ of all **formulas** of the language $\mathcal{L}_{CON}$ contains now an additional recursive step and goes as follows.

The set $\mathcal{F}$ of all formulas of the language $\mathcal{L}_{CON}$ with propositional constants is the smallest set built from the signs of the alphabet $\mathcal{A}$, i.e. $\mathcal{F} \subseteq \mathcal{A}^*$, such that the following conditions hold:

(1) $VAR \subseteq \mathcal{F}$ (atomic formulas),

(2) $C_0 \subseteq \mathcal{F}$ (atomic formulas),

(3) if $A \in \mathcal{F}$, $\bigtriangledown \in C_1$ i.e $\bigtriangledown$ is an one argument connective, then $\bigtriangledown A \in \mathcal{F}$,

(4) if $A, B \in \mathcal{F}$, $\circ \in C_2$ i.e $\circ$ is a two argument connective, then $(A \circ B) \in \mathcal{F}$.

Example 3.10

Let $\mathcal{L} = \mathcal{L}_{\{T,\neg,\cap\}}$, i.e. $C_0 = \{V\}$. Atomic formulas of $\mathcal{L}$ are all variables and the symbol T.

The language admits formulas that involve the symbol T like $T, \neg T$, $(T \cap a)$, $(\neg a \cap \neg T)$, $\neg(b \cap T)$, etc... We might interpret the symbol T as a symbol of truth (statement that is always true).

Here are some exercises and examples dealing with the formal definition of propositional languages, syntactical correctness, and their expressiveness.

Exercise 3.1

Given a language $\mathcal{L} = \mathcal{L}_{\{\neg,\ C, I, \cup, \cap, \Rightarrow\}}$ and the following set S.

$$S = \{\mathbf{C}\neg a \Rightarrow (a \cup b),\ (\mathbf{C}(\neg a \Rightarrow (a \cup b))),\ C\neg(a \Rightarrow (a \cup b))\}$$

Determine which of the elements of $\mathcal{S}$ are, and which are not well formed formulas of $\mathcal{L}$. If $A \in \mathcal{S}$ is not a correct formula write its corrected version. For each correct or corrected formula determine its main connective, its degree and write what it says in the natural language.

Solution
1. $\mathbf{C}\neg a \Rightarrow (a \cup b)$ is not a well formed formula. The corrected formula is $(\mathbf{C}\neg a \Rightarrow (a \cup b))$. Its main connective is $\Rightarrow$ and the degree is 4. The corrected formula says: *If negation of a is possible, then we have a or b.*

Another corrected formula is $\mathbf{C}(\neg a \Rightarrow (a \cup b))$. Its main connective is $\mathbf{C}$, the degree is 4. The corrected formula says: *It is possible that not a implies a or b.*

2. $(\mathbf{C}(\neg a \Rightarrow (a \cup b)))$ is not a well formed formula. The correct formula is $\mathbf{C}(\neg a \Rightarrow (a \cup b))$. The main connective is $\mathbf{C}$, the degree is 4. The formula $\mathbf{C}(\neg a \Rightarrow (a \cup b))$ says: *It is possible that not a implies a or b .*

3. The formula $\mathbf{C}\neg(a \Rightarrow (a \cup b))$ is a correct formula. The main connective is $\mathbf{C}$, the degree is 4. The formula says: *the negation of the fact that a implies a or b is possible.*

Exercise 3.2

Given a set S of formulas:

$$S = \{((a \Rightarrow \neg b) \Rightarrow \neg a), \Box(\neg\Diamond a \Rightarrow \neg a), (a \cup \neg(a \Rightarrow b))\}.$$

Define a formal language $\mathcal{L}_{CON}$ to which to which all formulas in S belong, i.e. a language **determined** *by the set S.*

Solution
Any propositional language $\mathcal{L}_{\mathcal{CON}}$ is determined by its set of connectives. The connectives appearing in the formulas of the set S are: $\Rightarrow, \neg b, \Box, \Diamond$ and $\cup$. Hence the required language is $\mathcal{L}_{\{\neg, \Box, \Diamond, \cup, \Rightarrow\}}$.

Exercise 3.3

Write down a set S_1 all sub-formulas of the $\Diamond((a \cup \neg a) \cap b)$, a set S_2 all proper sub-formulas of $\neg(a \Rightarrow (b \Rightarrow))$.

Solution
The set S_1 of all sub-formulas of $\Diamond((a \cup \neg a) \cap b)$ is

$$S_1 = \{\Diamond((a \cup \neg a) \cap b),\ ((a \cup \neg a) \cap b),\ (a \cup \neg a),\ \neg a,\ b,\ a\}$$

a, b are atomic sub-formulas, and $\Diamond((a \cup \neg a) \cap b)$ is not a proper sub-formula.

The set S_2 of all proper sub-formulas of $\neg(a \Rightarrow (b \Rightarrow c))$ is

$$S_2 = \{(a \Rightarrow (b \Rightarrow c)),\ (b \Rightarrow c), a, b, c\}.$$

Exercise 3.4

Write the following natural language statement **S***:*

"From the fact that it is possible that Anne is not a boy we deduce that it is not possible that Anne is not a boy or, if it is possible that Anne is not a boy, then it is not necessary that Anne is pretty."

in the following two ways.

1. *As a formula $A_1 \in \mathcal{F}_1$ of a language $\mathcal{L}_{\{\neg,\ \Box,\ \Diamond,\ \cap,\ \cup,\ \Rightarrow\}}$.*

2. *As a formula $A_2 \in \mathcal{F}_2$ of a language $\mathcal{L}_{\{\neg,\ \cap,\ \cup,\ \Rightarrow\}}$.*

Solution
1. We translate the statement **S** into a formula A_1 of the modal language $\mathcal{L}_{\{\neg,\ \Box,\ \Diamond,\ \cap,\ \cup,\ \Rightarrow\}}$ as follows.

Propositional variables are: a, b. The variable a denotes statement *Anne is a boy* and b denotes a statement *Anne is pretty.*

Propositional modal connectives are: $\Box$, $\Diamond$. The connective $\Diamond$ reads *it is possible that*, and $\Box$ reads *it is necessary that.*

Translation: the formula A_1 is $(\Diamond\neg a \ \Rightarrow\ (\neg\Diamond\neg a \cup (\Diamond\neg a \Rightarrow \neg\Box b)))$.

2. We translate our statement into a formula A_2 of the language $\mathcal{L}_{\{\neg,\ \cap,\ \cup,\ \Rightarrow\}}$ as follows.

Propositional variables are: a, b. The variable a denotes statement *it is possible that Anne is not a boy* and b denotes a statement *it is necessary that Anne is pretty.* **Translation:** the formula A_2 is $(a \Rightarrow (\neg a \cup (a \Rightarrow \neg b)))$.

Exercise 3.5

Write the following natural language statement **S***:*

"For all natural numbers $n \in N$ the following implication holds: if $n < 0$, then there is a natural number m, such that it is possible that $n + m < 0$, or it is not possible that there is a natural number m, such that $m > 0$"

in the following two ways.

1. *As a formula A_1 of a language $\mathcal{L}_{\{\neg,\ \cap,\ \cup,\ \Rightarrow\}}$.*

2. *As a formula A_2 of a language $\mathcal{L}_{\{\neg,\ \Box,\ \Diamond,\ \cap,\ \cup,\ \Rightarrow\}}$.*

Solution
1. We translate the statement **S** into a formula A_1 of the language $\mathcal{L}_{\{\neg,\ \cap,\ \cup,\ \Rightarrow\}}$ as follows.

Propositional variables are: a, b. The variable a denotes statement *For all natural numbers $n \in N$ the following implication holds: if $n < 0$, then there is a natural number m, such that it is possible that $n + m < 0$.* The variable b denotes statement *it is not possible that there is a natural number m, such that $m > 0$.* **Translation:** the formula A_1 is $(a \cup \neg b)$.

2. We translate the statement **S** into a formula A_2 of a language $\mathcal{L}_{\{\neg,\ \Box,\ \Diamond,\ \cap,\ \cup,\ \Rightarrow\}}$ as follows. Propositional variables are: a, b. The variable a denotes statement *For all natural numbers $n \in N$ the following implication holds: if $n < 0$, then there is a natural number m, such that it is possible that $n + m < 0$.* The variable b denotes statement *there is a natural number m, such that $m > 0$.* **Translation:** the formula A_2 is $(a \cup \neg\Diamond b)$.

3.2 Extensional Semantics M

Given a propositional language $\mathcal{L}_{CON}$, the symbols for its connectives always have some intuitive meaning. A formal definition of the meaning of these symbols is called a semantics for the language $\mathcal{L}_{CON}$. A given language can have different semantics but we always define them in order to single out special formulas of the language, called tautologies, i.e. formulas of the language that is always true under the given semantics.

We introduced in Chapter 2 a notion of a classical propositional semantics, discussed its motivation and underlying assumptions. The assumption was that we consider only two logical values. The other one was that all classical propositional connectives are *extensional*. We have also observed that in everyday language there are expressions such as "I believe that", "it is possible that", " certainly", etc.... and they are represented by some propositional connectives which are not extensional. Non-extensional connectives do not play any role in mathematics and so are not discussed in classical logic and will be studied separately.

The extensional connectives are defined intuitively as such that the logical value of the formulas form by means of these connectives and certain given formulas depends only on the logical value(s) of the given formulas. We adopt a following formal definition of extensional connectives for a propositional language $\mathcal{L}$ and of an extensional semantic for $\mathcal{L}$.

Definition 3.6 (Extensional Connectives)

Let $\mathcal{L}_{CON}$ be such that $CON = C_1 \cup C_2$, where C_1, C_2 are the sets of unary and binary connectives, respectively. Let LV be a non-empty set of logical values. A connective $\bigtriangledown \in C_1$ or $\circ \in C_2$ is called **extensional** *if it is defined by a respective function*

$$\bigtriangledown : \ LV \longrightarrow LV \quad \text{or} \quad \circ : \ LV \times LV \longrightarrow LV.$$

A semantics **M** for a language $\mathcal{L}_{CON}$ is called **extensional** provided all connectives in CON are extensional and its notion of tautology is defined in terms of connectives and their logical values (see definition 3.7).

A semantics with a set of m-logical values is called a m-valued semantics. The classical semantics is a special case of a 2-valued extensional semantics. Given a language, its different semantics define corresponding different logics. Classical semantics defines classical propositional logic with its set of classical propositional tautologies. Many of m- valued logics are defined by various extensional semantics with sets of logical values LV with more then 2 elements. The languages of many important logics like modal, multi-modal, knowledge, believe, temporal contain connectives that are not extensional. Consequently they are defined by the non-extensional semantics. The intuitionistic logic is based on the same language as the classical one, its Kripke Models semantics is not extensional.

Defining a semantics for a given propositional language means more then defining propositional connectives. The *ultimate goal* of any semantics is to define the notion of its own *tautology.* In order to define which formulas of $\mathcal{L}_{CON}$ we want to to be tautologies under a given semantics **M** we assume that the set LV of logical values of **M** always has a distinguished logical value, often denoted by T for "absolute" truth. We also can distinguish, and often we do, another special value F representing "absolute" falsehood. We will use these symbols T, F. We may also use other symbols like 1, 0 or others. The value T serves to define a notion of a tautology (as a formula always "true").

Extensional semantics share not only the similar pattern of defining their connectives (definition 3.6), but also the method of defining the notion of a *tautology.* We hence define a general notion of an extensional semantics (definition 3.7) as sequence of steps leading to the definition of a tautology. Here are the steps.

Step1: we define all connectives of **M** as specified by definition 3.6.

Step 2: we define the main component of the definition of a tautology, namely a function v that assigns to any formula $A \in \mathcal{F}$ its logical value from VL. It is often called a truth assignment and we will use this name.

Step 3: given a truth assignment v and a formula $A \in \mathcal{F}$, we define what does it mean that v *satisfies* A, i.e. that v is a *model* for A under semantics **M**.

Step 4: we define a notion of *tautology* as follows: A is a tautology under semantics **M** if and only if all truth assignments v *satisfy* A, i.e. all truth assignments v are *models* for A.

We use a notion of a model because it is an important, if not the most important notion of modern logic. It is usually defined in terms of the notion of satisfaction. In classical propositional logic these two notions are the same. The use of expressions *" v satisfies A"* and *"v is a model for A"* is interchangeable. This is also a case for the extensional semantics; in particular for some non-classical semantics, like m-valued semantics discussed in this chapter.

The notions of satisfaction and model are not interchangeable for predicate languages semantics. We already discussed these notions in chapter 2 and will define them in full formality in chapter 8 on predicate logic. The use of the notion of a model also allows us to adopt and discuss the standard predicate logic definitions of consistency and independence for propositional case.

Given a language $\mathcal{L}_{CON}$ and non-empty set LV of logical values. We assume that the set LV has a special, distinguished logical value which serves to define a notion of tautology under the semantics **M**. We denote this distinguished value as T. We define formally a general notion of an extensional semantics **M** for $\mathcal{L}_{CON}$ as follows.

Definition 3.7 (Extensional Semantics)

A formal definition of an extensional semantics $\mathbf{M}$ *for a given language* $\mathcal{L}_{CON}$ *consists of specifying the following steps defining its main components.*

Step 1*: we define a set LV of logical values and its distinguished value T, and define all connectives of* $\mathcal{L}_{CON}$ *to be extensional;*
Step 2*: we define notion of a truth assignment and its extension;*
Step 3*: we define notions of satisfaction, model, counter model;*
Step 4*: we define notion of a tautology under the semantics* $\mathbf{M}$*.*

What differs one semantics from the other is the choice of the set LV of logical values and definition of the the connectives of $\mathcal{L}_{CON}$, i.e. the components defined in the Step1. The definitions for the Steps 2 and 3, 4 are modification of the definitions established for the classical case and they are as follows.

Step 1: we follow the definition 3.6 to define the connectives of $\mathbf{M}$.

Step 2 : we define a function called truth assignment and its extension in terms of the propositional connectives as defined in the Step 1. We use the term $\mathbf{M}$ truth assignment and $\mathbf{M}$ truth extension to stress that it is defined relatively to a given semantics $\mathbf{M}$.

Definition 3.8 (M Truth Assignment)

Let LV be the set of logical values of $\mathbf{M}$ *and VAR the set of propositional variables of the language* $\mathcal{L}_{CON}$*. Any function* $v : VAR \longrightarrow LV$*, is called a truth assignment under semantics* $\mathbf{M}$*, for short* $\mathbf{M}$ *truth assignment.*

Definition 3.9 (M Truth Extension)

Given $\mathbf{M}$ *truth assignment*
$v : VAR \longrightarrow LV$*. We define its extension* v^* *to the set* $\mathcal{F}$ *of all formulas of* $\mathcal{L}_{CON}$ *as any function*

$$v^* : \mathcal{F} \longrightarrow LV,$$

such that the following conditions are satisfied.

(i) *for any* $a \in VAR$*,*

$$v^*(a) = v(a);$$

(ii) *For any connectives* $\bigtriangledown \in C_1$*,* $\circ \in C_2$*, and for any formulas* A*,* $B \in \mathcal{F}$*,*

$$v^*(\bigtriangledown A) = \bigtriangledown v^*(A), \qquad v^*((A \circ B)) = \circ(v^*(A), v^*(B)).$$

We call the v^* *the* **M truth extension***.*

The symbols on the left-hand side of the equations represent connectives in their *natural language* meaning and the symbols on the right-hand side represent connectives in their *semantical meaning* as defined in the **Step1**.

We use names "**M** truth assignment", "**M** truth extension" to stress that we define them for the set of logical values of **M** and moreover, that the extension of v connects the formulas of the language with the connectives as defined by the semantics **M**.

Notation Remark For a given function f, we use a symbol f^* to denote its *extension* to a larger domain. Mathematician often use the same symbol f for both a function and its extension f^*.

Step 3: the notions of satisfaction and model are interchangeable in extensional semantics. They are not interchangeable in other propositional semantics and in semantics for predicate languages.We define them as follows.

Definition 3.10 (M Satisfaction, Model)

Given an **M** *truth assignment*

$v : VAR \longrightarrow LV$ *and its* **M** *truth extension* v^*. *Let* $T \in LV$ *be the distinguished logical value. We say that*
the truth assignment v **M satisfies** *a formula* A *if and only if* $v^*(A) = T$.
We write symbolically

$$v \models_{\mathbf{M}} A.$$

Any truth assignment v*, such that* $v \models_{\mathbf{M}} A$ *is called* **M model** *for* A.

Definition 3.11 (M Counter Model)

Given an **M** *truth assignment*

$v : VAR \longrightarrow LV$. *Let* $T \in LV$ *be the distinguished logical value. We say that* v **does not satisfy** *a formula* $A \in \mathcal{F}$ *if and only if* $v^*(A) \neq T$.

We denote it by

$$v \not\models_{\mathbf{M}} A.$$

A any v*, such that* $v \not\models_{\mathbf{M}} A$ *is called* **M counter model** *for* A.

Step 4: we define the notion of a tautology under semantics **M**, called **M tautology** as follows.

Definition 3.12 (M Tautology)

For any formula $A \in \mathcal{F}$,

A *is* **M tautology** *if and only if* $v \models_{\mathbf{M}} A$*, for all truth assignments* v*,* $v : VAR \longrightarrow LV$. *We denote it as*

$$\models_{\mathbf{M}} A.$$

We also say that A is **M tautology** *if and only if all truth assignments v are* **M models** *for A.*

Observe that directly from definition 3.11 we get the following equivalent form of the definition 3.12.

Definition 3.13

For any formula $A \in \mathcal{F}$,

A is a **M tautology** *if and only if $v^*(A) = T$, for all truth assignments v, $v : VAR \longrightarrow LV$.*

We denote by **MT** the set of all tautologies under the semantic **M**, i.e.

$$\mathbf{MT} = \{A \in \mathcal{F} : \ \models_{\mathbf{M}} A.\} \tag{3.2}$$

Obviously, when we develop a logic by defining its semantics we want the semantics to be such that the logic has a non empty set of its tautologies. We stress that fact by putting it in a form of the following definition.

Definition 3.14

Given a language $\mathcal{L}_{CON}$ and its extensional semantics **M** *(definition 3.7), we say that the semantics* **M** *is* **well defined** *if and only if its set* **MT** *of all tautologies (3.2) is non empty, i.e. when*

$$\mathbf{MT} \neq \emptyset \tag{3.3}$$

We follow the definitions and pattens established here first in section 3.3. We use them to define and discuss in details the classical propositional semantics. Definitions and short discussions of some of the many-valued semantics follow next in section 3.5. Many valued logics had their beginning in the work of Łukasiewicz (1920). He was the first to define a 3- valued extensional semantics for a language $\mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow\}}$ of classical logic, and called it *a three valued logic* for short. The other logics, now of historical value followed and we will discuss some of them. In particular we present a Heyting 3-valued semantics as an introduction to the definition and discussion of first ever semantics for the intuitionistic logic and some modal logics. It was proposed by J.C.C McKinsey and A. Tarski in 1946-48 in a form of cylindrical algebras, now called pseudo-boolean algebras, or Heyting algebras. The semantics in a form of abstract algebras are called algebraic models for logics. It became a separate field of modern logic. The algebraic models are generalization of the extensional semantics, hence the importance of this section. It can me treated as an introduction to *algebraic models for logics.* It will be discussed again in chapter7.

3.3 Classical Semantics

We adopt **Steps 1- 4** of the general definition 3.7 of extensional semantics to the case of the classical propositional logic as follows.

Step 1: we define the language, set of logical values, and define all connectives of the language to be **extensional**

The language is $\mathcal{L}_{\{\neg,\ \cup,\ \cap,\ \Rightarrow,\ \Leftrightarrow\}}$. The set LV of logical values is $\{T,\ F\}$. The letters T, F stand as symbols of truth and falsehood, respectively. We adopt T as the distinguished value. There are other notations for logical values, for example 0,1, but we will use T, F.

Definition of connectives

Negation is a function $\neg : \{T, F\} \longrightarrow \{T, F\}$, such that $\neg(T) = F,\ \neg(F) = T$. We write it as $\neg T = F,\ \ \neg F = T$.

Notation: we write the name of a two argument function (our connective) *between the arguments*, not in front as in function notation, i.e. we write for example, $T \circ T = T$ instead of $\circ(T, T) = T$.

Conjunction is a function $\cap :\ \{T, F\} \times \{T, F\} \longrightarrow \{T, F\}$, such that $\cap(T, T) = T,\quad \cap(T, F) = F,\quad \cap(F, T) = F,\quad \cap(F, F) = F$. We write it as $T \cap T = T,\quad T \cap F = F,\quad F \cap T = F,\quad F \cap F = F$.

Disjunction is a function $\cup :\ \{T, F\} \times \{T, F\} \longrightarrow \{T, F\}$, such that $\cup(T, T) = T,\ \ \cup(T, F) = T,\ \ \cup(F, T) = T,\ \ \cup(F, F) = F$. We write it as $T \cup T = T,\quad T \cup F = T,\quad F \cup T = T,\quad F \cup F = F$.

Implication is a function $\Rightarrow:\ \{T, F\} \times \{T, F\} \longrightarrow \{T, F\}$, such that $\Rightarrow (T, T) = T,\ \ \Rightarrow (T, F) = F,\ \ \Rightarrow (F, T) = T,\ \ \Rightarrow (F, F) = T$. We write it as $T \Rightarrow T = T,\ \ T \Rightarrow F = F,\ \ F \Rightarrow T = T,\ \ F \Rightarrow F = T$.

Equivalence is a function $\Leftrightarrow:\ \{T, F\} \times \{T, F\} \longrightarrow \{T, F\}$, such that $\Leftrightarrow (T, T) = T,\ \ \Leftrightarrow (T, F) = F,\ \ \Leftrightarrow (F, T) = F,\ \ \Leftrightarrow (T, T) = T$. We write it as $T \Leftrightarrow T = T,\quad T \Leftrightarrow F = F,\quad F \Leftrightarrow T = F,\quad T \Leftrightarrow T = T$.

We write function defining the connectives in a standard form of tables defining operations in finite sets. We call these tables (3.4) *truth tables definition* of propositional connectives, or *classical connectives truth tables* for short.

Classical Connectives Truth Tables (3.4)

¬	T	F
	F	T

∩	T	F
T	T	F
F	F	F

∪	T	F
T	T	T
F	T	F

$\Rightarrow$	T	F
T	T	F
F	T	T

$\Leftrightarrow$	T	F
T	T	F
F	F	T

As ultimate goal of our semantics is to define the notion of tautology, a formula that is always true, we assume that the set $\{T, F\}$ of our logical values is ordered and $F < T$, This makes the symbol T (for truth) the "greatest" logical value, what truth supposed to be. We now can write simple formulas defining the connectives (respective function) as follows.

Classical Connectives Formulas (3.5)

$\neg$: $\{F,T\} \longrightarrow \{F,T\}$, such that $\neg F = T$, $\neg T = F$.

$\cap$: $\{F,T\} \times \{F,T\} \longrightarrow \{F,\ T\}$, such that for any $x, y \in \{F,T\}$, $\cap(x,y) = min\{x,\ y\}$. We write it as $x \cup y = min\{x,\ y\}$.

$\cup$: $\{F,T\} \times \{F,T\} \longrightarrow \{F,\ T\}$, such that for any $x, y \in \{F,T\}$, $\cup(x,y) = max\{x,\ y\}$. We write it as $x \cup y = max\{x,\ y\}$.

$\Rightarrow$: $\{F,T\} \times \{F,T\} \longrightarrow \{F,\ T\}$, such that for any $x, y \in \{F,T\}$, $\Rightarrow (x,y) = \cup(\neg x, y)$. We write it as $x \Rightarrow y = \neg x \cup y$.

$\Leftrightarrow$: $\{F,T\} \times \{F,T\} \longrightarrow \{F,\ T\}$, such that for any $x, y \in \{F,T\}$, $\Leftrightarrow (x,y) = \cup(\Rightarrow (x,y), \Rightarrow (y,x))$.
We write it as $x \Leftrightarrow y = (x \Rightarrow y) \cap (y \Rightarrow x)$.

Exercise 3.6

Prove that the above connectives formulas are correct, i.e. that they define the same classical connectives as defined in **Step 1**.

Solution
This is a problem of proving equality of functions that are given the same names. We have to show that the use of the same names: $\neg$, $\cup$, $\cap$, $\Rightarrow$, $\Leftrightarrow$ for them is justified. The equality of functions is defined as follows.

Definition 3.15

Given two sets A, B and functions f, g, such that $f: A \longrightarrow B$ *and* $g: A \longrightarrow B$. *We say that the functions f, g are* **equal** *and write it* $f = g$ *if and only if f(x) = g(x) for all elements* $x \in A$.

The negation definition is the same in both cases. We prove that the two conjunctions and two disjunctions functions are the equal by comparing both

definitions (3.4) and (3.5). We verify now the correctness of the implication function formula. Consider two functions $\Rightarrow: \{T,F\} \times \{T,F\} \longrightarrow \{T,F\}$ and $h : \{T,F\} \times \{T,F\} \longrightarrow \{T,F\}$, where $\Rightarrow$ is the classical implication defined by definition (3.4) and h is defined by the definition (3.5), i.e. by the formula $h(x,y) = \cup(\neg x, y)$. Observe that we have already proved that functions $\cup$ and $\neg$ are equal in both cases. We prove that $\Rightarrow = h$ by evaluating that $\Rightarrow (x,y) = h(x,y) = \cup(\neg x, y)$, for all $(x,y) \in \{T,F\} \times \{T,F\}$ as as follows.
$T \Rightarrow T = T$ and $h(T,T) = \neg T \cup T = F \cup T = T$ yes.
$T \Rightarrow F = F$ and $h(T,F) = \neg T \cup F = F \cup F = F$ yes.
$F \Rightarrow F = T$ and $h(F,F) = \neg F \cup F = T \cup F = T$ yes.
$F \Rightarrow T = T$ and $h(F,T) = \neg F \cup T = T \cup T = T$ yes.
This proves the correctness of the implication formula $\Rightarrow (x,y) = \cup(\neg x, y)$. We write it as $x \Rightarrow y = \neg x \cup y$ and call it a formula defining implication in terms of disjunction and negation. We verify the correctness of the equivalence formula $\Leftrightarrow (x,y) = \cup(\Rightarrow (x,y), \Rightarrow (y,x))$ in a similar way.

Special Properties of Connectives

Observe that the formulas defining connectives of implication and equivalence are certain compositions of previously defined connectives. Classical semantics is a special one, its connectives have strong properties that often do not hold under other semantics, extensional or not. One of them is a property of *definability of connectives*, the other one is a *functional dependency.* These are basic properties one asks about any new semantics, and hence a logic, being created. We generalize these the notion of *functional dependency* of connectives under a given extensional semantics **M**.

Definition 3.16 (Definability of Connectives)

Given a propositional language $\mathcal{L}_{CON}$ *and its extensional semantics* **M**. *A connective* $\circ \in CON$ *is* **definable** *in terms of some connectives* $\circ_1, \circ_2, ...\circ_n \in CON$ *if and only if* $\circ$ *is a certain function composition of functions* $\circ_1, \circ_2, ...\circ_n$, *as they are defined by the semantics* **M**.

We have just proved in Exercise 3.6 that the implication $\Rightarrow$ is definable in terms of $\cup$ and $\neg$ under classical semantics as it is a composition of $\cup$ and $\neg$ defined by the formula $\Rightarrow (x,y) = \cup(\neg x, y)$. The classical equivalence is definable in terms of $\Rightarrow$ and $\cap$ by the formula $\Leftrightarrow (x,y) = \cup(\Rightarrow (x,y), \Rightarrow (y,x))$.

Definition 3.17 (Functional Dependency)

Given a propositional language $\mathcal{L}_{CON}$ *and its extensional semantics* **M**. *A property of defining the set of connectives CON in terms of its proper subset is called* **a functional dependency** *of connectives under the semantics* **M**.

Proving the property of functional dependency under a given semantics **M** consists of identifying a proper subset CON_0 of the set CON of connectives, such that each connective $\circ \in CON - CON_0$ is definable (definition 3.16) in terms of connectives from CON_0. This is usually a difficult, and often impossible task for many semantic. We prove now that it holds in the classical case.

Theorem 3.1

The set of connectives of the language $\mathcal{L}_{\{\neg,\ \cup,\ \cap,\ \Rightarrow,\ \Leftrightarrow\}}$ *is functionally dependent under the classical semantics.*

Proof
Let's take a set $\{\neg,\ \cup\}$. We have already proved in Exercise 3.6 that the implication $\Rightarrow$ and is definable in terms of $\cup$ and $\neg$ by the formula $x \Rightarrow y = \neg x \cup y$. The conjunction is defined by easy verification, similar to the one in Exercise 3.6, by a formula $x \cap y = \neg(\neg x \cup \neg y)$. By Exercise 3.6, the equivalence formula is definable in terms of $\Rightarrow$ and $\cap$ by the formula $x \Leftrightarrow y = (x \Rightarrow y) \cap (y \Rightarrow x)$. The final formula for for the equivalence is $x \Leftrightarrow y = (\neg x \cup y) \cap (\neg y \cup x)$.

There are many ways to prove this theorem, it means there are many ways to choose a proper subset CON_0 of the set $\{\neg,\ \cup,\ \cap,\ \Rightarrow,\ \Leftrightarrow\}$ that defines all other connectives. Here are the choices.

Theorem 3.2 (Definability of Connectives)

All connectives of the language $\mathcal{L}_{\{\neg,\ \cup,\ \cap,\ \Rightarrow,\ \Leftrightarrow\}}$ *are definable in terms of* $\neg$ *and* $\circ$*, for any* $\circ \in \{\cup,\ \cap,\ \Rightarrow\}$.

Proof
We list all required definability formulas, including the formulas developed in the proof of Theorem 3.1. An easy verification of their correctness is left as an exercise.
1. Definability in terms of $\Rightarrow$ and $\neg$.
$x \cap y = \neg(x \Rightarrow \neg y)$, $x \cup y = \neg x \Rightarrow y$, $x \Leftrightarrow y = \neg((x \Rightarrow y) \Rightarrow \neg(y \Rightarrow x))$.
2. Definability in terms of $\cap$ and $\neg$.
$x \cup y = \neg(\neg x \cap \neg y)$, $x \Rightarrow y = \neg(x \cap \neg y)$, $x \Leftrightarrow y = \neg(x \cap \neg y) \cap \neg(y \cap \neg x)$.
3. Definability in terms of $\cup$ and $\neg$.
$x \Rightarrow y = \neg x \cup y$, $x \cap y = \neg(\neg x \cup \neg y)$ $x \Leftrightarrow y = (\neg x \cup y) \cap (\neg y \cup x)$.

There are two other important classical binary connectives denoted by $\uparrow$ and $\downarrow$. The connective $\uparrow$ was discovered in 1913 by H.M. Sheffer, who called it *alternative negation.* Now it is often called a *Sheffer's connective.* A formula $(A \uparrow B)$ reads: *not both A and B.* The connective $\downarrow$ was discovered in 1920 by J. Łukasiewicz and named *joint negation.* The formula $(A \downarrow B)$ reads: *neither A nor B.* They are defined as follows.

Alternative Negation is a function $\uparrow$: $\{T, F\} \times \{T, F\} \longrightarrow \{T, F\}$ such that $T \uparrow T = F, \quad T \uparrow F = T, \quad F \uparrow T = T, \quad F \uparrow F = T.$

Joint Negation is a function $\downarrow$: $\{T, F\} \times \{T, F\} \longrightarrow \{T, F\}$ such that $T \downarrow T = F, \quad T \downarrow F = F, \quad F \downarrow T = F, \quad F \downarrow F = T.$

Truth Tables for $\uparrow$ and $\downarrow$

$\uparrow$	T	F
T	F	T
F	T	T

$\downarrow$	T	F
T	F	F
F	F	T

We extend our language $\mathcal{L}_{\{\neg,\ \cup,\ \cap,\ \Rightarrow,\ \Leftrightarrow\}}$ by adding Sheffer and Łukasiewicz connectives to it. We obtain the language $\mathcal{L}_{\{\neg,\ \cup,\ \cap,\ \Rightarrow,\ \Leftrightarrow,\ \uparrow,\ \downarrow\}}$ that contains now all possible classical connectives.

Theorem 3.3

All connectives of a language $\mathcal{L}_{\{\neg,\ \cup,\ \cap,\ \Rightarrow,\ \Leftrightarrow,\ \uparrow,\ \downarrow\}}$ are definable in terms of $\uparrow$, and also separately in terms of $\downarrow$.

Proof
Definability formulas of $\neg$ and $\cap$ in terms of $\uparrow$ are the following.

$$\neg x = x \uparrow x, \quad x \cap y = (x \uparrow y) \uparrow (x \uparrow y) \tag{3.6}$$

Definability formulas for of the connectives $\{\cup,\ \Rightarrow,\ \Leftrightarrow\}$ in terms of $\uparrow$ follow directly from the formulas in the proof of Theorem 3.2 and the formulas (3.6). Observe that the $x \uparrow y = \neg(x \cup y)$. The definability of $x \downarrow y$ in terms of $x \uparrow y$ follows from (3.6) and definability of $\cup$ in terms $\uparrow$.
Definability formulas of $\neg$ and $\cup$ in terms of $\downarrow$ are, by simple verification, the following.

$$\neg x = x \downarrow x, \quad x \cup y = (x \downarrow y) \downarrow (x \downarrow y) \tag{3.7}$$

Definability formulas for of the connectives $\{\cap,\ \Rightarrow,\ \Leftrightarrow, \uparrow\}$ in terms of $\downarrow$ follow directly, as in the previous case, from the Theorem 3.2 and the formulas (3.7).

Functional dependency and definability of connectives as expressed in Theorems 3.2, 3.3 are very strong and characteristic properties of the classical semantics. They hold, for some connectives for some non-classical logics, never in others. For example, the necessity connective $\Box$ is definable in terms of the possibility connectives $\Diamond$ and negation $\neg$ in Modal S4 and S5 logics, but not in majority of others. The classical implication is definable in terms of negation and disjunction, but the intuitionistic implication is not. We defined and discussed these classical properties here as they have to be addressed and examined when one is building semantics for any of a non-classical logic.

Step 2: Truth Assignment, Truth Extension

We define now and examine the components in the Step 2 of the definition 3.7. We start with the basic notion of the *truth assignment*. We adopt the extensional semantics $\mathbf{M}$ definition 3.8 to the classical case as follows.

Definition 3.18 (Truth Assignment)

Let VAR be the set of all propositional variables of the language $\mathcal{L}_{\{\neg,\ \cup,\ \cap,\ \Rightarrow,\ \Leftrightarrow\}}$. *A* **truth assignment** *is any function* $v : VAR \longrightarrow \{T, F\}$.

The function v defined above is called the truth assignment because it can be thought as an assignment to each variable (which represents a logical sentence) its logical value of T(ruth) of F(alse). Observe that the domain of the truth assignment is the set of propositional variables, i.e. the truth assignment is defined only for *atomic formulas.*

We now extend the truth assignment v from the set of *atomic formulas* to the set of all formulas $\mathcal{F}$ in order define formally the assignment of a logical value to *any formula* $A \in \mathcal{F}$.

The definition of the *truth extension* of the truth assignment v to the set $\mathcal{F}$ follows the definition 3.8 for the extensional semantics $\mathbf{M}$.

Definition 3.19 (Truth Extension) *Given the truth assignment* $v : VAR \longrightarrow \{T, F\}$. *We define its extension* v^* *to the set* $\mathcal{F}$ *of all formulas as any function* $v^* : \mathcal{F} \longrightarrow \{T, F\}$, *such that the following conditions are satisfied.*

(1) *for any* $a \in VAR$, $\quad v^*(a) = v(a)$;

(2) *for any* $A, B \in \mathcal{F}$,

$$v^*(\neg A) = \neg v^*(A);$$

$$v^*((A \cap B)) = \cap(v^*(A), v^*(B));$$

$$v^*((A \cup B)) = \cup(v^*(A), v^*(B));$$

$$v^*((A \Rightarrow B)) = \Rightarrow (v^*(A), v^*(B));$$

$$v^*((A \Leftrightarrow B)) = \Leftrightarrow (v^*(A), v^*(B)).$$

The symbols on the *left-hand side* of the equations represent the connectives in their *natural language meaning.* The symbols on the *right-hand side* represent connectives in their *classical semantics* meaning defined by the classical connectives defined by the classical Truth Tables.

For binary connectives (two argument functions) we adopt a convention to write the symbol of the connective (name of the 2 argument function) between its arguments as we do in a case arithmetic operations. We use this standard notation and re-write the definition 3.19 as follows.

Definition 3.20 (Standard Notation) *Given the truth assignment* $v : VAR \longrightarrow \{T, F\}$. *We define its extension* v^* *to the set* $\mathcal{F}$ *of all formulas as any function* $v^* : \mathcal{F} \longrightarrow \{T, F\}$, *such that the following conditions are satisfied.*

> **(1)** *for any* $a \in VAR$, $\quad v^*(a) = v(a)$;
>
> **(2)** *for any* $A, B \in \mathcal{F}$,

$$v^*(\neg A) = \neg v^*(A);$$
$$v^*((A \cap B)) = v^*(A) \cap v^*(B);$$
$$v^*((A \cup B)) = v^*(A) \cup v^*(B);$$
$$v^*((A \Rightarrow B)) = v^*(A) \Rightarrow v^*(B);$$
$$v^*((A \Leftrightarrow B)) = v^*(A) \Leftrightarrow v^*(B).$$

Given a formula A: $((a \Rightarrow b) \cup \neg a))$ and a truth assignment v, such that $v(a) = T$, $v(b) = F$. We evaluate the logical value of the formula A using the standard notation definition 3.20 as follows.
$v^*(A) = v^*(((a \Rightarrow b) \cup \neg a))) = \cup(v^*((a \Rightarrow b), v^*(\neg a)) = \cup(\Rightarrow (v^*(a), v^*(b)), \neg v^*(a)))$ $= \cup(\Rightarrow (v(a), v(b)), \neg v(a))) = \cup(\Rightarrow (T, F), \neg T)) = \cup(F, F) = F$.

Observe that we did not specify $v(x)$ of any $x \in VAR - \{a, b\}$, as these values do not influence the computation of the logical value of the formula A. We say: "v such that" as we consider its values for the variables a and b only. Nevertheless, the domain of the truth assignment v is *always* is the set of all variables VAR and we have to *remember* that.

Given a formula A: $((a \Rightarrow b) \cup \neg a))$ and a truth assignment v, such that $v(a) = F$, $v(b) = F$. We use now the standard notation definition 3.20 to evaluate the logical value of the formula A. We write is as follows. $v^*(A) = v^*(((a \Rightarrow b) \cup \neg a))) = v^*((a \Rightarrow b)) \cup v^*(\neg a) = (v(a) \Rightarrow v(b)) \cup \neg v(a) = (F \Rightarrow F) \cup \neg F = T \cup T = T$.

Step 3: Satisfaction, Model, Counter-Model

We define now and examine the components in Step 3 of the definition 3.7. We adopt the extensional semantics **M** definitions 3.10, 3.11, and 3.12 to the classical case as follows.

Definition 3.21 (Satisfaction)

Let $v : VAR \longrightarrow \{T, F\}$.
We say that v **satisfies** *a formula $A \in \mathcal{F}$ if and only if $v^*(A) = T$. We denote it by $v \models A$.*

v **does not satisfy** *a formula $A \in \mathcal{F}$ if and only if $v^*(A) \neq T$. We denote it by $v \not\models A$.*

The relation $\models$ is often called a **satisfaction relation**. Observe, that in the classical semantics we have that $v^*(A) \neq T$ if and only if $v^*(A) = F$. In this case we say that v **falsifies** a formula A.

Exercise 3.7

Let A be a formula $((a \Rightarrow b) \cup \neg a))$ and v be a truth assignment $v : VAR \longrightarrow \{T, F\}$, such that $v(a) = T$, $v(b) = F$, and $v(x) = F$ for all $x \in VAR - \{a, b\}$. Show that $v \not\models ((a \Rightarrow b) \cup \neg a))$.

Proof We evaluate the logical value of the formula A as follows: $v^*(A) = v^*((a \Rightarrow b) \cup \neg a)) = (v^*(a \Rightarrow b) \cup v^*(\neg a)) = ((v(a) \Rightarrow v(b)) \cup \neg v(a)) = ((T \Rightarrow F) \cup \neg T) = (F \cup F) = F$. It proves tha $v \not\models ((a \Rightarrow b) \cup \neg a))$ and we say that v falsifies the formula A.

As we remarked before, in practical cases we use a short-hand notation for while evaluating the logical value of a given formula. Here is a short procedure for any v and A. We use show it how it works for v and A from the exercise 3.7.

Short-hand Evaluation
Given any formula $A \in \mathcal{F}$ and any truth assignment $v : VAR \longrightarrow \{T, F\}$.
1. We write the value of v only for variables appearing in the formula in A.
In our case we write: $a = T$, $b = F$ for $v(a) = T, v(b) = F$.
2. Replace all variabes in A by their respective logical values.
In our case we replace a by T and b by F in the formula A $((a \Rightarrow b) \cup \neg a))$. We get an equation $((T \Rightarrow F) \cup \neg T)$.
3. Use use the connectives definition, in this case the definitionTTables to evaluate the logical value of the equation obtained in the step **2**.
In our case we evaluate $((T \Rightarrow F) \cup \neg T) = (F \cup F) = F$.
4. Write your answer in one of the forms: $v \models A$, $v \not\models A$ or "v satisfies A", " v falsifies A"

In our case v *falsifies* A and write $v \not\models ((a \Rightarrow b) \cup \neg a))$.

Example 3.11

Let A be a formula $((a \cap \neg b) \cup \neg c)$ and v be a truth assignment $v : VAR \longrightarrow \{T, F\}$, such that $v(a) = T$, $v(b) = F$, $v(c) = T$, and $v(x) = T$ for all $x \in VAR - \{a, b, c\}$. Using the **the short-hand** *notation we get $((T \cap \neg F) \cup \neg T) = ((T \cap T) \cup F) = (T \cup F) = T$. It proves that v satisfies the formula A and we write $v \models ((a \cap \neg b) \cup \neg c)$.*

Definition 3.22 (Model, Counter Model)

Given a formula $A \in \mathcal{F}$.

Any $v : VAR \longrightarrow \{T, F\}$, such that $v \models A$ is called a **model** *for A .*

Any v, such that $v \not\models A$ is called a **counter model** *for the formula A .*

The truth assignment from the Example 3.11 is a model for the formula $((a \cap \neg b) \cup \neg c)$ and the truth assignment from the Exercise 3.7 is a counter-model for the formula $((a \Rightarrow b) \cup \neg a))$.

Step 4: Classical Tautology Definition

There are two equivalent ways to define the notion of classical tautology. We will use them interchangeably. The first uses the notion of truth assignment and states the following.

Definition 3.23 (Tautology 1) *For any formula $A \in \mathcal{F}$,*
A is **tautology** *if and only if $v^*(A) = T$ for all $v : VAR \longrightarrow \{T, F\}$.*

The second uses the notion of satisfaction and model and the fact that in any extensional semantic the notions " *v satisfies A*" and "*v is a model for A*" are interchangeable. It is stated as follows.

Definition 3.24 (Tautology 2)

For any formula $A \in \mathcal{F}$,
A is **tautology** *if and only if $v \models A$ for all $v : VAR \longrightarrow \{T, F\}$, i.e. when all truth assignments are* **models** *for A.*

We write symbolically

$$\models A$$

for the statement "A is a **tautology**".

Remark 3.1

We use the symbol $\models A$ *only for classical tautology. For all other extensional semantics* $\mathbf{M}$ *we must use the symbol* $\models_{\mathbf{M}} A$ *and say " A is a tautology under a semantics* $\mathbf{M}$*, or to say in short "A is a* $\mathbf{M}$ *semantics tautology".*

We usually use the definition 3.24 to express that a formula in not a tautology, i.e. we say that a formula is not a tautology if it has a counter model. To stress it we put it in a form of a following definition.

Definition 3.25

For any formula $A \in \mathcal{F}$*,*
A is **not a tautology** *if and only if A has a* **counter model***;*
i.e. when there is a truth assignment $v : VAR \longrightarrow \{T, F\}$*, such that* $v \not\models A$*.*

We denote the statement "A **is not a tautology**" symbolically by

$$\not\models\ A.$$

A formula $A:\ ((a \Rightarrow b) \cup \neg a))$ is not a tautology ($\not\models\ ((a \Rightarrow b) \cup \neg a)))$. A truth assignment $v : VAR \longrightarrow \{T, F\}$, such that $v(a) = T,\ v(b) = F$, and $v(x) = F$ for all $x \in VAR - \{a, b\}$ is a counter model for A, as we proved Exercise 3.7.

This ends the formal definition of classical semantics that follows the pattern for extensional semantics established in the definition 3.7.

3.3.1 Tautologies: Decidability and Verification Methods

There is a large number of basic and important tautologies listed and discussed in Chapter 2. We assume that the reader is familiar, or will familiarize with them if needed. We will refer to them and use them within our book. Chapter 2 also provides the motivation for classical approach to definition of tautologies as ways of describing correct rules of our mathematical reasoning. It also contains an informal definition of classical semantics and discusses a tautology verification method. We have just defined formally the classical semantics. Our goal now is to prove formally that the notion of classical tautology is decidable (Theorem 3.9) and to prove correctness of the tautology verification method presented in Chapter 2. Moreover we present here other basic tautology verification methods and prove their correctness.

We start now a natural question. How do we verify whether a given formula $A \in \mathcal{F}$ is or is not a tautology? The answer seems to be very simple. By definition 3.23 we have to examine *all* truth assignments $v : VAR \longrightarrow \{T, F\}$. If they all evaluate to T, we proved that $\models A$. If at least one evaluates to F, we found a counter model and proved $\not\models A$. The verification process is decidable, if the we have only a finite number of v to consider. So now all we have to do is to *count* how many truth assignments there are, i.e. how many there are functions that map the set VAR of propositional variables into the set $\{T, F\}$ of logical values. In order to do so we need to introduce some standard notations and some known facts. For a given set X, we denote by $|X|$ the *cardinality* of X. In a case of a finite set, it is called a number of elements of the set. We write $|X| = n$ to denote that *X has n elements*, for $n \in N$. We have a special names and notations to denote the *cardinalities* of infinite set. In particular we write $|X| = \aleph_0$ and say " cardinality of X is aleph zero," for any countably infinite set X. We write $|X| = \mathcal{C}$ and say " cardinality of X is continuum" for any uncountable set X that has the same cardinality as Real numbers.

Theorem 3.4 (Counting Functions)

For any sets X, Y, there are $|Y|^{|X|}$ functions that map the set X into Y.

In particular, when the set X is countably infinite and the set Y is finite, then there are $n^{\aleph_0} = \mathcal{C}$ functions that map the set X into Y.

In our case of counting the truth assignment $v : VAR \longrightarrow \{T, F\}$ we have that $|VAR| = \aleph_0$ and $|\{T, F\}| = 2$. We know that $2^{\aleph_0} = \mathcal{C}$ and hence we get directly from Theorem 3.4 the following.

Theorem 3.5 (Counting Truth Assignments)

There are uncountably many (exactly as many as real numbers) of all possible truth assignments v, where $v : VAR \longrightarrow \{T, F\}$.

Definition 3.26

For any $A \in \mathcal{F}$, let VAR_A be a set of all propositional variables appearing in A. Any function $v_A : VAR_A \longrightarrow \{T, F\}$, is called a **truth assignment restricted** *to A.*

Example 3.12

Let $A = ((a \Rightarrow \neg b) \cup \neg c)$. The set of variables appearing in A is $VAR_A = \{a, b, c\}$. The truth assignment **restricted** *to A is any function $v_A : \{a, b, c\} \longrightarrow \{T, F\}$.*

Definition 3.27

Given a formula $A \in \mathcal{F}$ and a set VAR_A of all propositional variables appearing in A. Any function $v_A : VAR_A \longrightarrow \{T, F\}$, such that $v \models A$ ($v \not\models A$) is called a **restricted model (counter model)** *for A.*

We use the following particular case of Theorem 3.4 to count, for any formula A, possible truth assignment restricted to A, i.e. all possible restricted models and counter models for A.

Theorem 3.6 (Counting Functions 1)

For any finite sets X and Y, if X has n elements and Y has m elements, then there are m^n possible functions that map X into Y.

We also can prove it independently by Mathematical Induction over m.

Given a formula $A \in \mathcal{F}$, the set VAR_A is always finite, and $|\{T, F\}| = 2$, so directly from Theorem 3.6 we get the following.

Theorem 3.7 (Counting Restricted Truth)

For any $A \in \mathcal{F}$, there are $2^{|VAR_A|}$ of possible truth assignments **restricted** *to A*

So there are $2^3 = 8$ possible truth assignment **restricted** to the formula $A = ((a \Rightarrow \neg b) \cup \neg c)$. We usually list them, and their value on the formula A in a form of an extended truth table below.

v_A	a	b	c	$v^*(A)$ *computation*	$v^*(A)$
v_1	T	T	T	$(T \Rightarrow T) \cup \neg T = T \cup F = T$	T
v_2	T	T	F	$(T \Rightarrow T) \cup \neg F = T \cup T = T$	T
v_3	T	F	F	$(T \Rightarrow F) \cup \neg F = F \cup T = T$	T
v_4	F	F	T	$(F \Rightarrow F) \cup \neg T = T \cup F = T$	T
v_5	F	T	T	$(F \Rightarrow T) \cup \neg T = T \cup F = T$	T
v_6	F	T	F	$(F \Rightarrow T) \cup \neg F = T \cup T = T$	T
v_7	T	F	T	$(T \Rightarrow F) \cup \neg T = F \cup F = F$	F
v_8	F	F	F	$(F \Rightarrow F) \cup \neg F = T \cup T = T$	T

(3.8)

v_1, $_2$, v_3, v_4, v_5, v_6, v_8 are **restricted models** for A and v_7 is a **restricted counter model** for A.

Now we are ready to prove the *correctness* of the well known truth tables tautology verification method. We formulate it as the follows.

Theorem 3.8 (Truth Tables)

For any formula $A \in \mathcal{F}$,
$\models A$ *if and only if* $v_A \models A$ *for all* $v_A : VAR_A \longrightarrow \{T, F\}$, *i.e.*
$\models A$ *if and only if all* $v_A : VAR_A \longrightarrow \{T, F\}$ *are restricted models for A.*

Proof Assume $\models A$. By definition 3.24 we have that $v \models A$ for all $v : VAR \longrightarrow \{T, F\}$, hence $v_A \models A$ for all $v_A : VAR_A \longrightarrow \{T, F\}$ as $VAR_A \subseteq VAR$.

Assume $v_A \models A$ for all $v_A : VAR_A \longrightarrow \{T, F\}$. Take any $v : VAR \longrightarrow \{T, F\}$, as $VAR_A \subseteq VAR$, any $v : VAR \longrightarrow \{T, F\}$ is an *extersion* of some v_A, i.e. $v(a) = v_A(a)$ for all $a \in VAR_A$. By Truth Extension Definition 3.19 we get that $v^*(A) = {v_A}^*(A) = T$ and $v \models A$. This ends the proof.

Directly from Theorem 3.7 and the above Theorem 3.8 we get the proof of the correctness and decidability of the Truth Tables Method, and hence the decidability of the notion of classical propositional tautology.

Theorem 3.9 (Tautology Decidability)

For any formula $A \in \mathcal{F}$, one has to examine at most 2^{VAR_A} restricted truth assignments $v_A : VAR_A \longrightarrow \{F, T\}$ in order to decide whether

$$\models A \quad \textit{or} \quad \not\models A,$$

i.e. the notion of classical tautology is decidable.

We present now and prove correctness of some basic tautologies verification methods. We just proved (Theorem 3.9) the correctness of the truth table tautology verification method, so we we start with it.

Truth Table Method

The verification method, called a **truth table method** consists of examination, for any formula A, all possible truth assignments restricted to A. By theorem 3.7 we have to perform at most $2^{|VAR_A|}$ steps. If we find a truth assignment which evaluates A to F, we stop the process and give answer: $\not\models A$. Otherwise we continue. If all truth assignments evaluate A to T, we give answer: $\models A$.

We usually list all restricted truth assignments v_A in a form of a truth table similar to the table3.8, hence the name of the method.

Consider, for example, a formula A: $(a \Rightarrow (a \cup b))$. There are $2^2 = 4$ possible truth assignment restricted to A. We usually list them, and evaluate their value on the formula A in a form of an extended truth table as follows.

w	a	b	$w^*(A)$ *computation*	$w^*(A)$
w_1	T	T	$T \Rightarrow (T \cup T) = T \Rightarrow T = T$	T
w_2	T	F	$T \Rightarrow (T \cup F) = T \Rightarrow T = T$	T
w_3	F	T	$F \Rightarrow (F \cup T) = F \Rightarrow T = T$	T
w_4	F	F	$F \Rightarrow (F \cup F) = F \Rightarrow F = T$	T

(3.9)

The table (3.9) shows that all $w : VAR_A \longrightarrow \{T, F\}$ are restricted models for A and hence by Theorem 3.9 we proved that $\models (a \Rightarrow (a \cup b))$ and $\mathbf{T} \neq \emptyset$.

Observe that the table (3.9) proves that the formula $\not\models\ ((a \Rightarrow \neg b) \cup \neg c)$.

Moreover we have proved that the condition (3.3) of the definition 3.14 is fulfilled and the classical semantics is well defined. We put it as a separate statement.

Fact 3.2

The classical semantics is **well defined**.

The complexity of the truth table methods grows exponentially. Impossible for humans to handle formulas with more then few variables, and cumbersome for computers for formulas with a great number of variables, In practice, if we need, we use often much shorter and more elegant tautology verification methods presented below.

Proof by Contradiction Method

In this method, in order to prove that $\models A$ we proceed as follows.

We assume that $\not\models A$. We work with this assumption. If we get a **contradiction**, we have proved that $\not\models A$ is impossible. We hence proved $\models A$. If we do not get a contradiction, it means that the assumption $\not\models A$ is true, i.e. we have proved that A is not a tautology.

Exercise 3.8

Follow the Proof by Contradiction Method and examine whether
$\models (a \Rightarrow (a \cup b))$.

Solution
We use a short-hand notation.
Assume that $\not\models\ (a \Rightarrow (a \cup b))$. It means that $(a \Rightarrow (a \cup b)) = F$ for some truth assignment v. By definition of implication $\Rightarrow$ we have that
$(a \Rightarrow (a \cup b)) = F$ if and only if $a = T$ and $(a \cup b) = F$.

From $a = T$ and $(a \cup b) = F$ we get $(T \cup b) = F$. This is a **contradiction** with the definition of disjunction $\cup$. Hence we proved $\models\ (a \Rightarrow (a \cup b))$.

Exercise 3.9

Use the Proof by Contradiction Method to decide whether
$\models\ ((a \cup b) \Rightarrow a)$.

Solution We do not use short-hand notation.

Assume that $\not\models\ ((a \cup b) \Rightarrow a)$. It means that there is $v : VAR \longrightarrow \{T, F\}$, such that $v^*(((a \cup b) \Rightarrow a)) = F$. We evaluate, $v^*(((a \cup b) \Rightarrow a))) = v^*((a \cup b)) \Rightarrow v(a)$

and we get that the truth assignment v is such that $v^*((a \cup b)) \Rightarrow v(a) = F$. By definition implication $\Rightarrow$ we have that $v^*((a \cup b)) \Rightarrow v(a) = F$ if and only if $v(a) \cup v(b) = T$ and $(a) = F$. From $(a) = F$ and $v(a) \cup v(b) = T$ we get that $F \cup v(b) = T$. This is **possible** for any $v : VAR \longrightarrow \{T, F\}$, such that $v(b) = T$. This proves that any truth assignment $v : VAR \longrightarrow \{T, F\}$, such that $(a) = F, v(b) = T$ is a counter model for $((a \cup b) \Rightarrow a)$, i.e. that $\not\models ((a \cup b) \Rightarrow a)$.

Substitution Method

We define and prove the correctness of a method, called Substitution Method that allows us to obtain new tautologies from formulas already proven to be tautologies.

We can use the same reasoning as we used in the solution to the Exercise 3.8 that proved $\models (a \Rightarrow (a \cup b))$ to prove that, for example the formulas

$$((((a \Rightarrow b) \cap \neg c) \Rightarrow ((((a \Rightarrow b) \cap \neg c) \cup \neg d)) \tag{3.10}$$

$$((a \Rightarrow b) \cap \neg c) \cup d) \cap \neg e) \Rightarrow (((a \Rightarrow b) \cap \neg c) \cup d) \cap \neg e) \cup ((a \Rightarrow \neg e))) \tag{3.11}$$

are also a tautologies.

Instead of repeating the same argument from Exercise 3.8 for a much more complicated formulas we make a simple observation that we can obtain (3.10), (3.11) from the formula $(a \Rightarrow (a \cup b))$ by a proper *substitutions* (replacements) of more complicated formulas for the variables a and b in $(a \Rightarrow (a \cup b))$. We use a notation $A(a, b) = (a \Rightarrow (a \cup b))$ to denote that $(a \Rightarrow (a \cup b))$ is a formula A with two variables a, b and we denote by

$$A(a/A_1,\ b/A_2)$$

a result of a substitution (replacement) of formulas $A_1,\ A_2$ on a place of the variables a, b, respectively, everywhere where they appear in $A(a, b)$.
Theorem 3.10 we are going to prove states that *substitutions* lead always from a tautology to a tautology. In particular, making the following substitutions s1 and s2 in $A(a, b) = (a \Rightarrow (a \cup b))$ we get, that the respective formulas (3.10), (3.11) are tautologies.

By substitution s1: $A(a/((a \Rightarrow b) \cap \neg c),\ b/\neg d)$ we get that

$$\models ((((a \Rightarrow b) \cap \neg c) \Rightarrow ((((a \Rightarrow b) \cap \neg c) \cup \neg d)).$$

By substitution s2: $A(a/((a \Rightarrow b) \cap \neg c),\ b/((a \Rightarrow \neg e))$ we get that
$\models (((a \Rightarrow b) \cap \neg c) \cup d) \cap \neg e) \Rightarrow (((a \Rightarrow b) \cap \neg c) \cup d) \cap \neg e) \cup ((a \Rightarrow \neg e)))$.

The theorem 3.10 describes validity of a method of constructing new tautologies from given tautologies. In order to formulate and prove it we first introduce needed notations.

Let $A \in \mathcal{F}$ be a formula and $VAR_A = \{a_1, a_2, ...a_n\}$ be the set of all propositional variables appearing in A. We will denote it by $A(a_1, a_2, ...a_n)$.

Given a formula $A(a_1, a_2, ...a_n)$, and $A_1, ...A_n$ be any formulas. We denote by

$$A(a_1/A_1, ..., a_n/A_n)$$

the result of simultaneous replacement (substitution) in $A(a_1, a_2, ...a_n)$ the variables $a_1, a_2, ...a_n$ by formulas $A_1, ...A_n$, respectively.

Theorem 3.10

For any formulas $A(a_1, a_2, ...a_n),\ A_1,\ \dots\ ,\ A_n \in \mathcal{F}$,

If $\models A(a_1, a_2, ...a_n)$ *and* $B = A(a_1/A_1, ..., a_n/A_n)$, *then* $\models B$.

Proof. Let $B = A(a_1/A_1, ..., a_n/A_n)$. Let $b_1, b_2, ...b_m$ be all those propositional variables which occur in $A_1, ...A_n$. Given a truth assignment $v : VAR \longrightarrow \{T, F\}$, any values $v(b_1), v(b_2), ...v(b_m)$ defines the logical value of $A_1, ...A_n$, i.e. $v^*(A_1), ...v^*(A_n)$ and, in turn, $v^*(B)$.

Let $w : VAR \longrightarrow \{T, F\}$ be a truth assignment such that $w(a_1) = v^*(A_1), w(a_2) = v^*(A_2), ...w(a_n) = v^*(A_n)$. Obviously, $v^*(B) = w^*(A)$. Since A is a propositional tautology, $w^*(A) = T$, for all possible w, hence $v^*(B) = w^*(A) = T$ for all truth assignments w and B is also a tautology.

We have proved (Exercise 3.8) that the formula $D(a, b) = (a \Rightarrow (a \cup b))$ is a tautology. By the above Theorem 3.10 we get that $D(a/A,\ b/B) = ((A \cup B) \Rightarrow A)$ is a tautology. We hence get the following.

Fact 3.3

For any $A, B \in \mathcal{F}$, $\models ((A \cup B) \Rightarrow A)$.

Generalization Method

Now let's look at the task of finding whether the formulas (3.10), (3.11) are tautologies from yet another perspective. This time we observe that both of them are build in a similar way as a formula $(A \Rightarrow (A \cup B))$, for $A = ((a \Rightarrow b) \cap \neg c),\ B = \neg d$ in (3.10) and for $A = ((a \Rightarrow b) \cap \neg c),\ B = ((a \Rightarrow \neg e))$ in (3.11).

It means we represent, if it is possible, a given formula as a particular case of some much simpler *general formula.* Hence the name Generalization Method.

We then use Proof by Contradiction Method or Substitution Method to examine whether the representation of the given formula is /is not a tautology.

In this case, we prove, for example Proof by Contradiction Method by that $\models (A \Rightarrow (A \cup B))$, for any formulas $A, B \in \mathcal{F}$ and get, as a particular cases for A, B that that both formulas (3.10), (3.11) are tautologies.

Let's assume that there are formulas $A, B \in \mathcal{F} \not\models (A \Rightarrow (A \cup B))$. This means that $(A \Rightarrow (A \cup B)) = F$ for some truth assignment v. This holds only when $A = T$ and $(A \cup B) = F$, i.e. $(T \cup B) = F$. This is a contradiction with the definition of $\cup$. So $\models (A \Rightarrow (A \cup B))$ for all $A, B \in \mathcal{F}$.

Exercise 3.10

Show that $v \models (\neg((a \cap \neg b) \Rightarrow ((c \Rightarrow (\neg f \cup d)) \cup e)) \Rightarrow ((a \cap \neg b) \cap (\neg(c \Rightarrow (\neg f \cup d)) \cap \neg e)))$, *for all* $v : VAR \longrightarrow \{T, F\}$.

Solution
Observe that we really have to prove that $\models (\neg((a \cap \neg b) \Rightarrow ((c \Rightarrow (\neg f \cup d)) \cup e)) \Rightarrow ((a \cap \neg b) \cap (\neg(c \Rightarrow (\neg f \cup d)) \cap \neg e)))$. We can hence use any of our tautology verification methods. In this case $VAR_A = \{a, b, c, d, e, f\}$, so there are $2^6 = 64$ restricted truth assignments to consider. Much too many to apply the Truth Table Method. Our formula is also far too complicated to guess a simple tautology from which we could obtain it by the Substitution Method.

The Proof by Contradiction Method is less complicated, but before we apply it let's look closer at the sub-formulas of our formula and patterns they form inside the formula it, i.e. we try to apply the Generalization Method first.
Let's put $B = (a \cap \neg b)$, $C = (c \Rightarrow (\neg f \cup d))$, $D = e$. We re-write our formula in a general form as $(\neg(B \Rightarrow (C \cup D)) \Rightarrow (B \cap (\neg C \cap \neg D)))$ and prove that for all $B, C, D \in \mathcal{F}$,

$$\models (\neg(B \Rightarrow (C \cup D)) \Rightarrow (B \cap (\neg C \cap \neg D)).$$

We use Proof by Contradiction Method, i.e. we assume that there are formulas $B, C, D \in \mathcal{F}$, such that

$$\not\models (\neg(B \Rightarrow (C \cup D)) \Rightarrow (B \cap (\neg C \cap \neg D)).$$

This means that there is a truth assignment v, such that (we use short-hand notation) $(\neg(B \Rightarrow (C \cup D)) \Rightarrow (B \cap (\neg C \cap \neg D))) = F$. By definition of implication it is possible if and only if $\neg(B \Rightarrow (C \cup D)) = T$ and $(B \cap (\neg C \cap \neg D)) = F$, i.e. if and only if
$(B \Rightarrow (C \cup D)) = F$ and $(B \cap (\neg C \cap \neg D)) = F$ Observe that $(B \Rightarrow (C \cup D)) = F$ if and only if $B = T$, $C = F$, $D = F$. We now evaluate the logical value of $(B \cap (\neg C \cap \neg D))$ for $B = T, C = F, D = F$, i.e. we compute $(B \cap (\neg C \cap \neg D)) = (T \cap (\neg F \cap \neg F)) = (T \cap (T \cap T)) = T$. This **contradicts** that we must have $(B \cap (\neg C \cap \neg D)) = F$. This proves that for all $B, C, D \in \mathcal{F}$

$$\models (\neg(B \Rightarrow (C \cup D)) \Rightarrow (B \cap (\neg C \cap \neg D))),$$

and hence is holds for our particular case, i..e.

$$\models (\neg((a \cup b) \Rightarrow ((c \Rightarrow d) \cup e)) \Rightarrow ((a \cup b) \cap (\neg(c \Rightarrow d) \cap \neg e)))$$

and that all truth assignments are models for $(\neg((a \cup b) \Rightarrow ((c \Rightarrow d) \cup e)) \Rightarrow ((a \cup b) \cap (\neg(c \Rightarrow d) \cap \neg e)))$.

Sets of Formulas; Tautologies and Contradictions

We distinguish now special sets of formulas and examine their properties. We define sets of all tautologies, contradictions, consistent sets, inconsistent sets and discuss a notion of independence of formulas from sets of formulas.

Definition 3.28 (Set of Tautologies)

We denote by $\mathbf{T}$ *the set of all tautologies, i.e. we put*

$$\mathbf{T} = \{A \in \mathcal{F} : \quad \models A.\}$$

We distinguish now another type of formulas, called *contradictions.*

Definition 3.29 (Contradiction)

A formula $A \in \mathcal{F}$ *is called a contradiction if it does not have a model.*

We write symbolically $=\mid A$ for the statement "A is a **contradiction**."

Directly from the Definition 8.21 we have that

$$=\mid A \quad \text{if and only if} \quad v \not\models A \ \text{ for all } \ v : \ VAR \longrightarrow \{T, F\}.$$

Example 3.13

The following formulas are contradictions

$$(a \cap \neg a), \quad (a \cap \neg(a \cup b)), \quad \neg(a \Rightarrow a), \quad \neg(\neg(a \cap b) \cup b)).$$

Definition 3.30 (Set of Contradictions)

We denote by $\mathbf{C}$ *the set of all tautologies, i.e. we put*

$$\mathbf{C} = \{A \in \mathcal{F} : \quad =\mid A.\}$$

Following the proof of Theorem 3.10 we get similar theorem for contradictions, and hence a a proof of correctness of the Substitution Method of constructing new contradictions.

Theorem 3.11

For any formulas $A(a_1, a_2, ...a_n), \ A_1, \ \dots, \ A_n \in \mathcal{F}$,

If $A(a_1, a_2, ...a_n) \in \mathbf{C}$ *and* $B = A(a_1/A_1, ..., a_n/A_n)$, *then* $B \in \mathbf{C}$.

Directly from the Theorem 3.11 we get the following.

Example 3.14 *For any formulas $A, B \in \mathcal{F}$, the following formulas are contradictions*

$$(A \cap \neg A), \quad (A \cap \neg(A \cup B)), \quad \neg(A \Rightarrow A), \quad \neg(\neg(A \cap B) \cup B)).$$

Observe, that there are formulas which neither in $\mathbf{T}$ nor in $\mathbf{C}$, for example $(a \cup b)$. Any truth assignment v, such that $v(a) = F, v(b) = F$ falsifies $(a \cup b)$ and it proves that it is not a tautology. Any truth assignment v, such that $v(a) = T, v(b) = T$ satisfies $(a \cup b)$, what proves that it is not a contradiction.

3.3.2 Sets of Formulas: Consistency and Independence

Next important notions for any logic are notions of consistency, inconsistency of the sets of formulas and the independence of a formula from the set of formulas. We adopt the following definitions.

Definition 3.31

A truth truth assignment $v : VAR \longrightarrow \{T, F\}$ is **model** *for the set $\mathcal{G} \subseteq \mathcal{F}$ of formulas if and only if $v \models A$ for all formulas $A \in \mathcal{G}$. We denote it by*

$$v \models \mathcal{G}.$$

The restriction $v_{\mathcal{G}}$ of the **model** *v to the domain $VAR_{\mathcal{G}} = \bigcup_{A \in \mathcal{G}} VAR_A$ is called a* **restricted model** *for $\mathcal{G}$.*

Exercise 3.11

Find a model and a restricted model for a set

$$\mathcal{G} = \{((a \cap b) \Rightarrow b), \ (a \cup b), \neg a\}.$$

Solution
Let v be a truth assignment $v : VAR \longrightarrow \{T, F\}$. By the defininition 3.31, $v \models \{((a \cap b) \Rightarrow b), \ (a \cup b), \neg a\}$ if and only if $v^*(((a \cap b) \Rightarrow b)) = T$, $v^*((a \cup b) = T)$, and $v^*(\neg a) = T$. Observe that $\models ((a \cap b) \Rightarrow b)$, so we have to find v, such that $v^*((a \cup b)) = T$, $v^*(\neg a) = T$. This holds if and only if $v(a) = F$ and $F \cup v(b) = T$, i.e. if and only if $v(a) = F$ and $v(b) = T$. This proves that any v such that $v(a) = F$ and $v(b) = T$ is a model for $\mathcal{G}$, and $\mathcal{G}$ has only one restricted model. We put it as a separate fact.

Fact 3.4

Given $\mathcal{G} = \{((a \cap b) \Rightarrow b),\ (a \cup b), \neg a\}$, we have that $VAR_{\mathcal{G}} = \bigcup_{A \in \mathcal{G}} VAR_A = \{a, b\}$ and $v_{\mathcal{G}} : \{a, b\} \longrightarrow \{T, F\}$, such that $v_{\mathcal{G}}(a) = F$ and $v_{\mathcal{G}}(b) = T$ is a unique **restricted model** *for $\mathcal{G}$.*

Observation 3.4

For some sets $\mathcal{G} \subseteq \mathcal{F}$, $VAR_{\mathcal{G}}$ can be infinite. For example, for $\mathcal{G} = VAR$ we have that $VAR_{\mathcal{G}} = VAR$ and the notions of model and restricted model are the same.

Definition 3.32

A set $\mathcal{G} \subseteq \mathcal{F}$ is called ***consistent*** *if and only if there is $v : VAR \longrightarrow \{T, F\}$, such that $v \models \mathcal{G}$.*

Otherwise the set $\mathcal{G}$ is called ***inconsistent****.*

Plainly speaking, a set $\mathcal{G}$ is consistent if it has a model, and is inconsistent if it does not have a model.

Example 3.15

The set $\mathcal{G}_1 = \{((a \cap b) \Rightarrow b),\ (a \cup b), \neg a\}$ is **consistent** *as $v : VAR \longrightarrow \{T, F\}$, such that $v(a) = F$ and $v(b) = T$ is the model for $\mathcal{G}_1$.*

The set $\mathcal{G}_2 = VAR$ is also **consistent***, as $v : VAR \longrightarrow \{T, F\}$, such that $v(a) = T$, for all $a \in VAR$ is a model for $\mathcal{G}_2$.*

Observe that $\mathcal{G}_1$ is a finite consistent set. $\mathcal{G}_2$ is an infinite consistent set. This and other examples justify the need of truth assignment domain being the set VAR of all propositional variables.

Example 3.16

The set $\mathcal{G}_1 = \{((a \cap b) \Rightarrow b),\ (a \cap \neg a), \neg a\}$ is a finite **inconsistent** *set as it contains a formula $(a \cap \neg a) \in \mathbf{C}$.*

The set $\mathcal{G}_2 = VAR \cup \{\neg a\}$ for some $a \in VAR$, is an infinite **inconsistent** *set as it contains a certain variable a and its negation $\neg a$.*

Of course the most obvious example of an infinite consistent set is the set $\mathbf{T}$ of all tautologies, and of an infinite inconsistent consistent set is the set $\mathbf{C}$ of all contradictions.

Definition 3.33

A formula $A \in \mathcal{F}$ is called ***independent*** *from a set $\mathcal{G} \subseteq \mathcal{F}$ if and only if the sets $\mathcal{G} \cup \{A\}$ and $\mathcal{G} \cup \{\neg A\}$ are both* **consistent**. *I.e. when there are truth assignments v_1, v_2 such that*

$$v_1 \models \mathcal{G} \cup \{A\} \quad \textit{and} \quad v_2 \models \mathcal{G} \cup \{\neg A\}.$$

Exercise 3.12

Show that a formula $A = ((a \Rightarrow b) \cap c)$ is ***independent*** *from the set $\mathcal{G} = \{((a \cap b) \Rightarrow b),\ (a \cup b), \neg a\}$.*

Solution
We define two truth assignments $v_1,\ v_2 : VAR \longrightarrow \{T, F\}$ such that $v_1 \models \mathcal{G} \cup \{(a \Rightarrow b) \cap c)\}$ and $v_2 \models \mathcal{G} \cup \{\neg(a \Rightarrow b) \cap c)\}$ as follows. We have just proved (Exercise 3.11) that any $v : VAR \longrightarrow \{T, F\}$, such that $v(a) = F$, $v(b) = T$ is a model for $\mathcal{G}$. Take as v_1 any truth assignment such that $v_1(a) = v(a) = F$, $v_1(b) = v(b) = T$, $v_1(c) = T$. We evaluate $v_1{}^*(A) = v_1{}^*(((a \Rightarrow b) \cap c)) = (F \Rightarrow T) \cap T = T$. This proves that $v_1 \models \mathcal{G} \cup \{A\}$. Take as v_2 any truth assignment such that, $v_2(a) = v(a) = F$, $v_2(b) = v(b) = T$, $v_2(c) = F$. We evaluate $v_2{}^*(\neg A) = v_2{}^*(\neg(((a \Rightarrow b) \cap c)) = T \cap T = T$. This proves that $v_2 \models \mathcal{G} \cup \{\neg A\}$. It ends the proof that formula A is **independent** from $\mathcal{G}$.

Exercise 3.13

Show that a formula $A = (\neg a \cap b)$ is ***not independent*** *from $\mathcal{G} = \{((a \cap b) \Rightarrow b),\ (a \cup b), \neg a\}$.*

Solution We have to show that it is impossible to construct v_1, v_2 such that $v_1 \models \mathcal{G} \cup \{A\}$ and $v_2 \models \mathcal{G} \cup \{\neg A\}$. From Fact 6.1 $\mathcal{G}$ has a unique restricted model $v : \{a, b\} \longrightarrow \{T, F\}$, such that $v(a) = F$, and $v(b) = T$. and $\{a, b\} = VAR_A$. So we have to check now if it is possible $v \models A$ and $v \models \neg A$. We evaluate $v^*(A) = v^*((\neg a \cap b) = \neg v(a) \cap v(b) = \neg F \cap T = T \cap T = T$ and get $v \models A$. By definition $v^*(\neg A) = \neg v^*(A) = \neg T = F$ and $v \not\models \neg A$. This end the proof that the formula $A = (\neg a \cap b)$ is **not independent** from $\mathcal{G}$.

Exercise 3.14

Given a set $\mathcal{G} = \{a, (a \Rightarrow b)\}$.
Find a formula A that is **independent** *from $\mathcal{G}$.*

Solution
Observe that truth assignment v such that $v(a) = T, v(b) = T$ is the only restricted model for $\mathcal{G}$. So we have to come up with a formula A such that there are two different truth assignments, v_1, v_2 such that $v_1 \models \mathcal{G} \cup \{A\}$ and $v_2 \models \mathcal{G} \cup \{\neg A\}$. Let's think about as simple a formula as it could be, namely

let's consider $A = c$, where c any propositional variable (atomic formula) different from a and b. $\mathcal{G} \cup \{A\} = \{a, (a \Rightarrow b), c\}$ and any truth assignment v_1, such that $v_1(a) = T,\ v_1(b) = T, v_1(c) = T$ is a model for $\mathcal{G} \cup \{c\}$. Likewise for $\mathcal{G} \cup \{\neg c\} = \{a, (a \Rightarrow b), \neg c\}$. Any v_2 such that $v_2(a) = T,\ v_2(b) = T, v_2(c) = F$ is a model for $\mathcal{G} \cup \{\neg c\}$. This proves that we have found the formula $A = c$ that is **independent** from $\mathcal{G}$.

Here is a simple generalization of the Exercise 3.14.

Exercise 3.15

Find an infinite number of formulas that are **independent** *from* $\mathcal{G} = \{((a \cap b) \Rightarrow b),\ (a \cup b), \neg a\}$.

Solution
First we have to find all $v : VAR \longrightarrow \{T, F\}$ such that $v \models \{((a \cap b) \Rightarrow b),\ (a \cup b), \neg a\}$, i.e such that (shorthand notation) $((a \cap b) \Rightarrow b) = T,\ (a \cup b) = T,\ \neg a = T$. Observe that $\models ((a \cap b) \Rightarrow b)$, so we have to consider only $(a \cup b) = T,\ \neg a = T$. This holds if and only if $a = F$ and $(F \cup b) = T$, i.e. if and only if $a = F$ and $b = T$. This proves that that $v_{\mathcal{G}}$ such that $v_{\mathcal{G}}(a) = F$ and $v_{\mathcal{G}}(b) = T$ is the only one restricted model for $\mathcal{G}$. All possible models for $\mathcal{G}$ must be extensions of $v_{\mathcal{G}}$. We define a countably infinite set of formulas (and their negations) and corresponding extensions v of $v_{\mathcal{G}}$ (restricted to to the set of variables $\{a, b\}$) such that $v \models \mathcal{G}$ as follows.

Observe that all extensions of v of $v_{\mathcal{G}}$ have as domain the infinitely countable set $VAR = \{a_1,\ a_2,\ \dots, a_n \dots\}$. We take as the infinite set of formulas in which every formula is to be proved independent of $\mathcal{G}$ the set of *atomic formulas*

$$\mathcal{F}_0 = VAR - \{a,\ b\} = \{a_1,\ a_2,\ \dots, a_n \dots\} - \{a,\ b\}.$$

Let $c \in \mathcal{F}_0$. We define truth assignments $v_1,\ v_2 : VAR \longrightarrow \{T, F\}$ as follows

$v_1(a) = v(a) = F,\ v_1(b) = v(b) = T$, and $v_1(c) = T$ for all $c \in \mathcal{F}_0$.

$v_2(a) = v(a) = F,\ v_2(b) = v(b) = T$, and $v_2(c) = F$ for all $c \in \mathcal{F}_0$.

Obviously, $v_1 \models \mathcal{G} \cup \{c\}$ and $v_2 \models \mathcal{G} \cup \{\neg c\}$ for all $c \in \mathcal{F}_0$. What proves that the set $\mathcal{F}_0$ is a countably infinite set of formulas **independent** from $\mathcal{G} = \{((a \cap b) \Rightarrow b),\ (a \cup b), \neg a\}$.

3.4 Classical Tautologies and Equivalence of Languages

We first present here a set of most widely used classical propositional tautologies which we will use, in one form or other, in our investigations in future chapters.

Another extended list of tautologies and their discussion is presented in Chapter 2.

As the next step we define notions of a *logical equivalence* and an *equivalence of languages*. We prove that all of the languages

$$\mathcal{L}_{\{\neg \Rightarrow\}},\ \mathcal{L}_{\{\neg \cap\}},\ \mathcal{L}_{\{\neg \cup\}},\ \mathcal{L}_{\{\neg, \cap, \cup, \Rightarrow\}},\ \mathcal{L}_{\{\neg, \cap, \cup, \Rightarrow, \Leftrightarrow\}},\ \mathcal{L}_{\{\uparrow\}},\ \mathcal{L}_{\{\downarrow\}}$$

are equivalent under classical semantics and hence can be used (and are) as different languages for classical propositional logic.

We generalize these notions to the case of any extensional semantics $\mathbf{M}$ in section 3.6. We also discuss and examine some particular many valued extensional semantics and properties of their languages in section 3.5.

Some Tautologies

For any $A, B \in \mathcal{F}$, the following formulas are tautologies.

Implication and Negation

$$(A \Rightarrow (B \Rightarrow A)), \quad ((A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C))),$$
$$((\neg B \Rightarrow \neg A) \Rightarrow ((\neg B \Rightarrow A) \Rightarrow B)) \tag{3.12}$$
$$(A \Rightarrow A), \quad (B \Rightarrow \neg\neg B), \quad (\neg A \Rightarrow (A \Rightarrow B)), \quad (A \Rightarrow (\neg B \Rightarrow \neg(A \Rightarrow B))),$$
$$(\neg\neg B \Rightarrow B), \quad ((A \Rightarrow B) \Rightarrow ((\neg A \Rightarrow B) \Rightarrow B)), \quad ((\neg A \Rightarrow A) \Rightarrow A).$$

Disjunction, Conjunction

$$(A \Rightarrow (A \cup B)), \quad (B \Rightarrow (A \cup B)), \quad ((A \cap B) \Rightarrow A), \quad ((A \cap B) \Rightarrow A),$$
$$((A \Rightarrow C) \Rightarrow ((B \Rightarrow C) \Rightarrow ((A \cup B) \Rightarrow C))),$$
$$(((A \cap B) \Rightarrow C) \Rightarrow (A \Rightarrow (B \Rightarrow C)),$$
$$(\neg(A \cap B) \Rightarrow (\neg A \cup \neg B)), \quad ((\neg A \cup \neg B) \Rightarrow \neg(A \cap B)), \tag{3.13}$$
$$((\neg A \cup B) \Rightarrow (A \Rightarrow B)), \quad ((A \Rightarrow B) \Rightarrow (\neg A \cup B)),$$
$$(A \cup \neg A).$$

Contraposition (1)

$$((A \Rightarrow B) \Leftrightarrow (\neg B \Rightarrow \neg A)), \quad ((B \Rightarrow A) \Leftrightarrow (\neg A \Rightarrow \neg B)). \tag{3.14}$$

Contraposition (2)

$$((\neg A \Rightarrow B) \Leftrightarrow (\neg B \Rightarrow A)), \quad ((A \Rightarrow \neg B) \Leftrightarrow (B \Rightarrow \neg A)). \tag{3.15}$$

Double Negation

$$(\neg\neg A \Leftrightarrow A), \tag{3.16}$$

Logical Equivalences

Logical equivalence is a very useful notion to use when we want to obtain new formulas or new tautologies, if needed, on a base of some already known in a way that guarantee preservation of the logical value of the initial formula. For any formulas A, B, we say that are *logically equivalent* if they always have the same logical value. We write it symbolically as $A \equiv B$. We have to remember that the symbol " $\equiv$" not a logical connective. It is a *metalanguage symbol* for saying "A, B are logically equivalent". This is a very useful symbol. It says that two formulas always have the same logical value, hence it can be used in the same way we use the equality symbol " =." Formally we define it as follows.

Definition 3.34 (Logical Equivalence)

For any $A,\ B \in \mathcal{F}$, we say that the formulas A and B are logically equivalent and denote it as $A \equiv B$

if and only if $v^*(A) = v^*(B)$, for all $v : VAR \ \rightarrow \ \{T, F\}$.

Observe that the following property follows directly from the definition 3.34.

Property 3.1

For any formulas $A, B \in \mathcal{F}$,

$$A \equiv B \quad \textit{if and only if} \quad \models (A \Leftrightarrow B)$$

For example we write the laws of contraposition (3.17), (3.18), and the law of double negation (3.19) as **logical equivalences** as follows.

E - Contraposition (1), (2)

$$(A \Rightarrow B) \equiv (\neg B \Rightarrow \neg A), \quad (B \Rightarrow A) \equiv (\neg A \Rightarrow \neg B). \tag{3.17}$$

E - Contraposition (2)

$$(\neg A \Rightarrow B) \equiv (\neg B \Rightarrow A), \quad (A \Rightarrow \neg B) \equiv (B \Rightarrow \neg A). \tag{3.18}$$

E - Double Negation

$$\neg\neg A \equiv A. \tag{3.19}$$

Logical equivalence is a very useful notion when we want to obtain new formulas, or tautologies, if needed, on a base of some already known in a way that guarantee preservation of the logical value of the initial formula.

For example, we easily obtain equivalences for laws of E -Contraposition (3.18) from equivalences for laws of E- Contraposition (3.17) and the E - Double Negation equivalence (3.19) as follows. $(\neg A \Rightarrow B) \equiv^{(3.17)} (\neg B \Rightarrow \neg\neg A) \equiv^{(3.19)} (\neg B \Rightarrow A)$. We also have that $(A \Rightarrow \neg B) \equiv^{(3.17)} (\neg\neg B \Rightarrow \neg A) \equiv^{(3.19)} (B \Rightarrow \neg A)$. This end the proof of E- Contraposition (3.18).

The correctness of the above procedure of proving new equivalences from the known ones is established by the following theorem.

Theorem 3.12 (Equivalence Substitution)

Let a formula B_1 be obtained from a formula A_1 by a substitution of a formula B for one or more occurrences of a sub-formula A of A_1, what we denote as

$$B_1 = A_1(A/B).$$

Then the following holds for any formulas $A,\ A_1,\ B,\ B_1 \in \mathcal{F}$.

$$\textit{If} \quad A \equiv B, \quad \textit{then} \;\; A_1 \equiv B_1. \tag{3.20}$$

Proof
By the logical equivalence Definition 3.34 proving our theorem statement 8.44 is equivalent to proving that the implication

$$\text{If} \quad v^*(A) = v^*(B), \quad \text{then} \;\; v^*(A_1) = v^*(B_1) \tag{3.21}$$

holds for all $v : VAR \;\rightarrow\; \{T, F\}$.

Consider a truth assignment v. If $v^*(A) \neq v^*(B)$, then the implication (3.21) is vacuously true. If $v^*(A) = v^*(B)$, then so $v^*(A_1) = v^*(B_1)$, since B_1 differs from A_1 only in containing B in some places where A_1 contains A and the implication (3.21) holds.

Example 3.17

Let $A_1 = (C \cup D)$ and $B = \neg\neg C$. By E - Double Negation equivalence (3.19) we have that $\neg\neg C \equiv C$. Let $B_1 = A_1(C/B) = A_1(C/\neg\neg C) = (\neg\neg C \cup D)$. By the Equivalence Substitution Theorem 3.12

$$(C \cup D) \equiv (\neg\neg C \cup D).$$

Equivalence of Languages

The next set of equivalences, or corresponding tautologies, correspond the notion of *definability of connectives* discussed in section 3.3. For example, a tautology

$$\models ((A \Rightarrow B) \Leftrightarrow (\neg A \cup B))$$

makes it possible, via Property 3.1, to define implication in terms of disjunction and negation. We state it in a form of logical equivalence and call it as follows.

Definability of Implication in terms of negation and disjunction:

$$(A \Rightarrow B) \equiv (\neg A \cup B) \tag{3.22}$$

Observation 3.5 *The direct proof of this and other Definability of Connectives Equivalences presented here follow from the definability formulas developed in the the proof of the Definability of Connectives Theorem 3.2, hence the names.*

We are using the logical equivalence notion, instead of the tautology notion, as it makes the manipulation of formulas much easier.

The equivalence 3.22 allows us, by the force of Theorem 3.12 to replace any formula of the form $(A \Rightarrow B)$ placed anywhere in another formula by a formula $(\neg A \cup B)$ while preserving their logical equivalence. Hence we can use the equivalence (3.22) to transform a given formula containing implication into an logically equivalent formula that does contain implication (but contains negation and disjunction).

We usually use the equation 3.22 to transform any formula A of language containing implication into a formula B of language containing disjunction and negation and not containing implication at all, such that $A \equiv B$.

Example 3.18

Let $A = ((C \Rightarrow \neg B) \Rightarrow (B \cup C))$.

We use equality (3.22) to transform A into a logically equivalent formula not containing $\Rightarrow$ as follows.

$((C \Rightarrow \neg B) \Rightarrow (B \cup C)) \equiv (\neg(C \Rightarrow \neg B) \cup (B \cup C))) \equiv (\neg(\neg C \cup \neg B) \cup (B \cup C)))$.

It means that for example that we can, by the Theorem 3.12 transform any formula A of the language $\mathcal{L}_1 = \mathcal{L}_{\{\neg, \cap, \Rightarrow\}}$ into a logically formula B of the language $\mathcal{L}_2 = \mathcal{L}_{\{\neg, \cap, \cup\}}$. In general, we say that we can transform a language $\mathcal{L}_1$ into a logically equivalent language $\mathcal{L}_2$ if the following condition **C1** holds.

C1: for any formula A of $\mathcal{L}_1$, there is a formula B of $\mathcal{L}_2$, such that $A \equiv B$.

Example 3.19

Let $A = (\neg A \cup (\neg A \cup \neg B))$. *We also can use, in this case, the equivalence 3.22 as follows.*

$(\neg A \cup (\neg A \cup \neg B)) \equiv (\neg A \cup (A \Rightarrow \neg B)) \equiv (A \Rightarrow (A \Rightarrow \neg B))$.

It means we eliminated disjunction from A by replacing it by logically equivalent formula containing implication only.

Observe, that we can't always use the equivalence (3.22) to eliminate any disjunction. For example, we can't use it for a formula $A = ((a \cup b) \cap \neg a)$.

In order to be able to transform any formula of a language containing disjunction (and some other connectives) into a language with negation and implication (and some other connectives), but without disjunction we need the following logical equivalence.

Definability of Disjunction in terms of negation and implication:

$$(A \cup B) \equiv (\neg A \Rightarrow B) \tag{3.23}$$

Example 3.20

Consider a formula $A = (a \cup b) \cap \neg a)$.

We use equality (3.23) to transform A into its logically equivalent formula not containing $\cup$ *as follows:* $((a \cup b) \cap \neg a) \equiv ((\neg a \Rightarrow b) \cap \neg a)$.

In general, we use the equality 3.23 and Theorem 3.12 to transform any formula C of the language $\mathcal{L}_2 = \mathcal{L}_{\{\neg,\cap,\cup\}}$ into a logically equivalent formula D of the language $\mathcal{L}_1 = \mathcal{L}_{\{\neg,\cap,\Rightarrow\}}$. In general, the following condition hols.

C2: for any formula C of $\mathcal{L}_2$, there is a formula D of $\mathcal{L}_1$, such that $C \equiv D$.

The languages $\mathcal{L}_1$ and $\mathcal{L}_2$ for which we the conditions **C1, C2** hold are logically equivalent and denote it by $\mathcal{L}_1 \equiv \mathcal{L}_2$.

We put it in a general, formal definition as follows.

Definition 3.35 (Equivalence of Languages)

Given two languages:
$\mathcal{L}_1 = \mathcal{L}_{CON_1}$ *and* $\mathcal{L}_2 = \mathcal{L}_{CON_2}$, *for* $CON_1 \neq CON_2$.
We say that they are **logically equivalent** *and denote it as* $\mathcal{L}_1 \equiv \mathcal{L}_2$ *if and only if the following conditions* **C1, C2** *hold.*

C1 *For every formula* A *of* $\mathcal{L}_1$, *there is a formula* B *of* $\mathcal{L}_2$, *such that* $A \equiv B$,

C2 *For every formula* C *of* $\mathcal{L}_2$, *there is a formula* D *of* $\mathcal{L}_1$, *such that* $C \equiv D$.

Example 3.21

To prove the logical equivalence $\mathcal{L}_{\{\neg,\cup\}} \equiv \mathcal{L}_{\{\neg,\Rightarrow\}}$ *we need two definability equivalences (3.22) and (3.23), and the Theorem 3.12.*

Exercise 3.16

To prove the logical equivalence $\mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow\}} \equiv \mathcal{L}_{\{\neg,\cap,\cup\}}$ we needed only the definability equivalence (3.22).

Solution
The equivalence (3.22) proves, by Theorem 3.12 that for any formula A of $\mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow\}}$ there is B of $\mathcal{L}_{\{\neg,\cap,\cup\}}$ that equivalent to A, i.e. condition **C1** holds. Any formula A of language $\mathcal{L}_{\{\neg,\cap,\cup\}}$ is also a formula of $\mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow\}}$ and of course $A \equiv A$, so both conditions **C1** and **C2** of definition 3.35 are satisfied.

Exercise 3.17

Show that $\mathcal{L}_{\{\neg,\cap\}} \equiv \mathcal{L}_{\{\neg,\Rightarrow\}}$.

Solution
The equivalence of languages holds by Theorem 3.12, Observation 3.5, and the following two logical equalities. **Definability of Conjunction** in terms of implication and negation and **Definability of Implication** in terms of conjunction and negation:

$$(A \cap B) \equiv \neg(A \Rightarrow \neg B) \tag{3.24}$$

$$(A \Rightarrow B) \equiv \neg(A \cap \neg B). \tag{3.25}$$

Exercise 3.18

Show that $\mathcal{L}_{\{\neg,\cap\}} \equiv \mathcal{L}_{\{\neg,\cup\}}$.

Solution
Similarly, it is true by Theorem 3.12, Observation 3.5, and the following two logical equalities. Definability of disjunction in terms of negation and conjunction and definability of conjunction in terms of negation and disjunction:

$$(A \cup B) \equiv \neg(\neg A \cap \neg B) \tag{3.26}$$

$$(A \cap B) \equiv \neg(\neg A \cup \neg B). \tag{3.27}$$

Theorem 3.12, Observation 3.5, and definability of equivalence in terms of implication and conjunction equality

$$(A \Leftrightarrow B) \equiv ((A \Rightarrow B) \cap (B \Rightarrow A)). \tag{3.28}$$

prove that, for example, $\mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow\}} \equiv \mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow,\Leftrightarrow\}}$.

Exercise 3.19

Show that $\mathcal{L}_{\{\neg,\cap\}} \equiv \mathcal{L}_{\{\uparrow\}}$ and $\mathcal{L}_{\{\neg,\cap\}} \equiv \mathcal{L}_{\{\uparrow\}}$

Proof
We use the proof of Theorem 3.3 to prove the following definability equivalences of $\neg$ and $\cap$ in terms of $\uparrow$:

$$\neg A \equiv (A \uparrow A), \quad (A \cap B) \equiv (A \uparrow B) \uparrow (A \uparrow B) \tag{3.29}$$

and definability equivalences of $\neg$ and $\cup$ in terms of $\downarrow$:

$$\neg A \equiv (A \downarrow A), \; (A \cup B) \equiv (A \downarrow B) \downarrow (A \downarrow B). \tag{3.30}$$

This proves the condition **C1** of definition 3.35.

The definability equivalences for fulfillment of the condition **C2** are:

$$(A \uparrow B) = \neg(A \cup B) \quad \text{and} \quad (A \uparrow B) = \neg(A \cup B) \tag{3.31}$$

Here are some more frequently used, important logical equivalences.

Idempotent

$$(A \cap A) \equiv A, \qquad (A \cup A) \equiv A,$$

Associativity

$$((A \cap B) \cap C) \equiv (A \cap (B \cap C)), \;\; ((A \cup B) \cup C) \equiv (A \cup (B \cup C)),$$

Commutativity

$$(A \cap B) \equiv (B \cap A), \qquad (A \cup B) \equiv (B \cup A),$$

Distributivity

$$(A \cap (B \cup C)) \equiv ((A \cap B) \cup (A \cap C)), \;\; (A \cup (B \cap C)) \equiv ((A \cup B) \cap (A \cup C)),$$

De Morgan Laws

$$\neg(A \cup B) \equiv (\neg A \cap \neg B), \;\; \neg(A \cap B) \equiv (\neg A \cup \neg B).$$

Negation of Implication

$$\neg(A \Rightarrow B) \equiv (A \cap \neg B), \tag{3.32}$$

De Morgan laws are named after A. De Morgan (1806 - 1871), an English logician, who discovered analogous laws for the algebra of sets. They stated that for any sets A,B the complement of their union is the same as the intersection of their complements, and vice versa, the complement of the intersection of two sets is equal to the union of their complements. The laws of the propositional calculus were formulated later, but they are usually also called De Morgan Laws.

Observe that De Morgan Laws tell us how to negate disjunction and conjunction, so the laws stating how to negate other connectives follows them.

Consider a tautology A: $\models ((\neg(A \Rightarrow B) \Rightarrow \neg A) \Rightarrow (A \Rightarrow B))$.

We know by (3.22) that $(A \Rightarrow B) \equiv (\neg A \cup B)$. By Theorem 3.12, if we replace $(A \Rightarrow B)$ by $(\neg A \cup B)$ in A, the logical value of A will remain the same and $((\neg(A \Rightarrow B) \Rightarrow \neg A) \Rightarrow (A \Rightarrow B)) \equiv ((\neg(\neg A \cup B) \Rightarrow \neg A) \Rightarrow (\neg A \cup B))$. Now we use de Morgan Laws and Double Negation Laws and by Theorem 3.12 we get $((\neg(A \Rightarrow B) \Rightarrow \neg A) \Rightarrow (A \Rightarrow B)) \equiv ((\neg(\neg A \cup B) \Rightarrow \neg A) \Rightarrow (\neg A \cup B)) \equiv (((\neg\neg A \cap \neg B) \Rightarrow \neg A) \Rightarrow (\neg A \cup B)) \equiv (((A \cap \neg B) \Rightarrow \neg A) \Rightarrow (\neg A \cup B))$.

This proves that

$$\models (((A \cap \neg B) \Rightarrow \neg A) \Rightarrow (\neg A \cup B)).$$

Exercise 3.20

Prove using proper logical equivalences that

(i) $\neg(A \Leftrightarrow B) \equiv ((A \cap \neg B) \cup (\neg A \cap B))$,

(ii) $((B \cap \neg C) \Rightarrow (\neg A \cup B)) \equiv ((B \Rightarrow C) \cup (A \Rightarrow B))$.

Solution (i)
$\neg(A \Leftrightarrow B) \equiv^{(3.28)} \neg((A \Rightarrow B) \cap (B \Rightarrow A)) \equiv^{de\ Morgan} (\neg(A \Rightarrow B) \cup \neg(B \Rightarrow A)) \equiv^{(3.32)} ((A \cap \neg B) \cup (B \cap \neg A)) \equiv^{commut} ((A \cap \neg B) \cup (\neg A \cap B))$.

Solution (ii)
$((B \cap \neg C) \Rightarrow (\neg A \cup B)) \equiv^{(3.23)} (\neg(B \cap \neg C) \cup (\neg A \cup B)) \equiv^{de\ Morgan} ((\neg B \cup \neg\neg C) \cup (\neg A \cup B)) \equiv^{(3.19)} ((\neg B \cup C) \cup (\neg A \cup B)) \equiv^{(3.23)} ((B \Rightarrow C) \cup (A \Rightarrow B))$.

3.5 Many Valued Semantics: Łukasiewicz, Heyting, Kleene, Bohvar

Many valued logics in general and 3-valued logics in particular is an old object of study which has its beginning in the work of a Polish mathematician Jan Leopold Łukasiewicz in 1920. He was the first to define a 3 - valued semantics for the language $\mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow\}}$ of classical logic, and called it a *three valued logic* for short. He left the problem of finding a proper axiomatic proof system for it (i.e. the one that is complete with respect to his semantics) open. The same happened to all other 3 - valued semantics presented here. They were also first called *3 valued logics* and this terminology is still widely used. Nevertheless, as these logics were defined only semantically, i.e. defined by providing a semantics for their languages we call them just semantics (for logics to be developed), not logics. Existence of a proper axiomatic proof system for a given semantics and proving its completeness is always a next open question to be answered (when it is possible). A process of creating a logic (based on a given language) always is three fold: we define semantics, create axiomatic proof system and prove completeness theorem that established a relationship between semantics and proof system.

The first of many valued logics invented were first presented in a semantical form only for other components to be developed later. We can think about the process of their creation as inverse to the creation of Classical Logic, Modal Logics, the Intuitionistic Logic which existed as axiomatic systems longtime before invention of their formal semantics.

Formal definition of many valued extensional semantics $\mathbf{M}$ for the language $\mathcal{L}$ we present and discuss here follows the extensional semantics Definition 3.7 in general and the pattern of presented in detail for the classical case (Section 3.3) in particular. It consists of giving definitions of the following main components:

Step 1: given the language $\mathcal{L}$ we define a set of logical values and its distinguish value T, and define all logical connectives of $\mathcal{L}$
Step 2: we define notions of a truth assignment and its extension;
Step 3: we define notions of satisfaction, model, counter model;
Step 4: we define notions tautology under the semantics $\mathbf{M}$.

We present here some of the historically first 3-valued extensional semantics, called also 3-valued logics. They are named after their authors: *Łukasiewicz, Heyting, Kleene*, and *Bochvar*.

The 3-valued semantics we define here enlist a third logical value, besides classical T, F. We denote this third value by $\perp$, or m in case of Bochvar semantics. We also assume that the third value is intermediate between truth and falsity, i.e. that $F < \perp < T$ and $F < m < T$.

There has been many of proposals relating both to the intuitive interpretation of this third value $\perp$. If T is the only designated value, the third value $\perp$ corresponds to some notion of *incomplete information*, like *undefined* or *unknown* and is often denoted by the symbol U or I. If, on the other hand, $\perp$ corresponds to *inconsistent information*, i.e. its meaning is something like *known to be both true and false* then corresponding semantics takes both T and the third logical value $\perp$ as designated. In general, the third logical value denotes a notion of "unknown", "uncertain", "undefined", or even can express that "we don't have a complete information", depending on the context and motivation for the logic we plan to develop. In all of presented here semantics we take T as *designated value*, i.e. T is the value that defines the notion of *satisfiability* and *tautology.*

Łukasiewicz Semantics L

Motivation

Łukasiewicz developed his semantics (called logic) to deal with future contingent statements. According to him, such statements are not just neither true nor false but are indeterminate in some metaphysical sense. It is not only that we do not know their truth value but rather that they do not possess one. Intuitively, $\perp$ signifies that the statement cannot be assigned the value true of false; it is not simply that we do not have sufficient information to decide the truth value but rather the statement does not have one.

We define all the steps of the Definition3.7 in case of Łukasiewicz' s semantics to establish a pattern and proper notation. We leave the detailed steps of other semantics as an exercise for the reader.

Step 1: L Connectives

The language of the semantics **L** is $\mathcal{L}_{\{\neg,\ \cup,\ \cap,\ \Rightarrow\}}$. The set LV of logical values is $\{T,\ \perp, F\}$. T is the distinguished value. We assume that the set of log0cal values is ordered, i.e. that

$$F <\perp< T.$$

L Negation is a function $\neg : \{T, \perp, F\} \longrightarrow \{T, \perp, F\}$ such that

$$\neg \perp=\perp, \quad \neg T = F, \quad \neg F = T.$$

L Conjunction is a function $\cap : \{T, \perp, F\} \times \{T, \perp, F\} \longrightarrow \{T, \perp, F\}$ such that for any $(x, y) \in \{T, \perp, F\} \times \{T, \perp, F\}$, we put

$$x \cap y = min\{x,\ x\}.$$

L Disjunction is a function $\cup : \{T, \perp, F\} \times \{T, \perp, F\} \longrightarrow \{T, \perp, F\}$, such that for any $(a, b) \in \{T, \perp, F\} \times \{T, \perp, F\}$, we put

$$x \cup y = max\{x,\ y\}$$

L Implication is a function $\Rightarrow$: $\{T, \perp, F\} \times \{T, \perp, F\} \longrightarrow \{T, \perp, F\}$ such that for any $(x, y) \in \{T, \perp, F\} \times \{T, \perp, F\}$, we put

$$x \Rightarrow y = \begin{cases} \neg x \cup y & \text{if } x > y \\ T & \text{otherwise} \end{cases} \tag{3.33}$$

We write function defining the connectives in a standard form of tables defining operations in finite sets. We call these tables *truth tables definition* of propositional connectives, or *L connectives truth tables* for short.

L Connectives Truth Tables

$\neg$	F	$\perp$	T
	T	$\perp$	F

$\cap$	F	$\perp$	T
F	F	F	F
$\perp$	F	$\perp$	$\perp$
T	F	$\perp$	T

$\cup$	F	$\perp$	T
F	F	$\perp$	T
$\perp$	$\perp$	$\perp$	T
T	T	T	T

$\Rightarrow$	F	$\perp$	T
F	T	T	T
$\perp$	$\perp$	T	T
T	F	$\perp$	T

Step 2: Truth Assignment, Truth Extension

A **truth assignment** is now any function $v : VAR \longrightarrow \{F,\ \perp,\ T\}$. We define its extension to the set $\mathcal{F}$ of all formulas as any function $v^* : \mathcal{F} \longrightarrow \{T, F\}$, such that the following conditions are satisfied.

(1) for any $a \in VAR$, $v^*(a) = v(a)$;

(2) for any $A, B \in \mathcal{F}$,

$$v^*(\neg A) = \neg v^*(A);$$

$$v^*((A \cap B)) = v^*(A) \cap v^*(B);$$

$$v^*((A \cup B)) = v^*(A) \cup v^*(B);$$

$$v^*((A \Rightarrow B)) = v^*(A) \Rightarrow v^*(B).$$

Step 3: Satisfaction, Model, Counter-Model

We say that a truth assignment $v : VAR \longrightarrow \{F,\ \perp,\ T\}$ **L** satisfies a formula $A \in \mathcal{F}$ if and only if $v^*(A) = T$. We denote it by $v \models_{\mathbf{L}} A$.

Any truth assignment v, $v : VAR \longrightarrow \{F,\ \perp,\ T\}$ such that $v \models_{\mathbf{L}} A$ is called a **L** model for A.

We say that a truth assignment v does not **L** satisfy a formula $A \in \mathcal{F}$ and denote it by $v \not\models_{\mathbf{L}} A$, if and only if $v^*(A) \neq T$.

Any truth assignment v, $v : VAR \longrightarrow \{F,\ \perp,\ T\}$ such that $v \not\models_{\mathbf{L}} A$ is called a **L** counter- model for A.

Step 4: L Tautology

We define, for any $A \in \mathcal{F}$, A is a **L** tautology if and only if $v^*(A) = T$ for all $v : VAR \longrightarrow \{F, \perp, T\}$. We also say that A is a **L** tautology if and only if all truth assignments $v : VAR \longrightarrow \{F, \perp, T\}$ are **L** models for A. We write the statement " A is a **L** tautology" symbolically as

$$\models_{\mathbf{L}} A.$$

As a next step we define, as we did in the case of classical semantics the notions of restricted truth assignment and restricted models, (Definitions 3.26, 3.27) i.e. we have the following.
Any function $v_A : VAR_A \longrightarrow \{F, \perp, T\}$, such that $v_A \models_{\mathbf{L}} A$ ($v_A \not\models_{\mathbf{L}} A$) is called a restricted **L** model (**L** counter model) for A, where VAR_A is the set of all propositional variables appearing in A. We call the function v_A, a truth assignment restricted to A, or restricted truth assignment for short.

We prove, in the same way we proved Theorem 3.8 in Section 3.3, the following theorem that justifies the correctness of the truth tables **L** tautologies verification method.

Theorem 3.13 (L Truth Tables)

For any formula $A \in \mathcal{F}$,
$\models_{\mathbf{L}} A$ *if and only if* $v_A \models_{\mathbf{L}} A$ *for all* $v_A : VAR_A \longrightarrow \{T, \perp, F\}$, *i.e.*
$\models_{\mathbf{L}} A$ *if and only if all* v_A *are restricted models for* A.

Directly from Theorem 3.13 we get that the notion of **L** propositional tautology is decidable, i.e. that the following holds.

Theorem 3.14 (Decidability)

For any formula $A \in \mathcal{F}$, one has examine at most 3^{VAR_A} truth assignments $v_A : VAR_A \longrightarrow \{F, \perp, T\}$ in order to decide whether $\models_{\mathbf{L}} A$, or $\not\models_{\mathbf{L}} A$, i.e. the notion of **L** *tautology is decidable.*

We denote by **LT** the set of all **L** tautologies, i.e. we have that

$$\mathbf{LT} = \{A \in \mathcal{F} : \models_{\mathbf{L}} A\}. \tag{3.34}$$

We just proved (Theorem 3.14) the correctness of the truth table tautology verification method for **L** semantics stated as follows.

L Truth Table Method

The verification method, called a **truth table method** consists of examination, for any formula A, all possible truth assignments **restricted** to A. By Theorem 3.13 we have to perform at most $3^{|VAR_A|}$ steps. If we find a truth assignment which **does not** evaluate A to T, i.e. evaluates A to F, or to $\perp$, we stop the process and give answer: $\not\models_{\mathbf{L}} A$. Otherwise we continue. If all truth assignments evaluate A to T, we give answer: $\models_{\mathbf{L}} A$.

Consider, for example, a formula A: $(a \Rightarrow a)$. There are $3^1 = 3$ possible restricted truth assignment $v: \{a\} \longrightarrow \{F, \perp, T\}$. We list them, and evaluate their value on the formula A in a form of an extended truth table as follows.

v	a	$v^*(A)$ *computation*	$v^*(A)$
v_1	T	$T \Rightarrow T = T$	T
v_2	$\perp$	$\perp \Rightarrow \perp = T$	T
v_3	F	$F \Rightarrow F = T$	T

This proves that the classical tautology $(a \Rightarrow a)$ is also a **L** tautology, i.e.

$$\models \ (a \Rightarrow a) \ \text{ and } \ \models_{\mathbf{L}} \ (a \Rightarrow a). \tag{3.35}$$

Moreover (3.35) proves that the condition (3.3) of the definition 3.14 is fulfilled and the **L** semantics is well defined. We put it as a separate fact.

Fact 3.5

The Łukasiewicz semantics **L** *is* **well defined**.

As a next step we can adopt all other classical tautology verification methods from Section 3.3. It is a quite straightforward adaptation and we leave it a san exercise. Moreover it works for all of many valued semantics presented here, as does the Decidability Theorem 3.14.

When defining and developing a new logic the first question one asks is how it relates and compares with the classical case, it means with the classical logic. In case of new semantics (logics defined semantically) we describe this relationship in terms of respective sets of tautologies.

Let **LT**, **T** denote the sets of all **L** and classical tautologies, respectively.

Theorem 3.15

The following relationship holds between classical and **L** *tautologies:*

$$\mathbf{LT} \neq \mathbf{T} \ \textit{and} \ \ \mathbf{LT} \subset \mathbf{T}. \tag{3.36}$$

Proof
Consider a formula $(\neg a \cup a)$. It is obviously a classical tautology. Take any truth assignment $v: VAR \longrightarrow \{F, \perp, T\}$ such that $v(a) = \perp$. By definition we have that $v^*(\neg a \cup a) = v^*(\neg a) \cup v^*(a) = \neg v(a) \cup v(a) = \neg \perp \cup \perp = \perp \cup \perp = \perp$.

This proves that v is a **L** counter-model for $(\neg a \cup a)$ and hence $\not\models_{\mathbf{L}} \ (\neg a \cup a)$. This proves $\mathbf{LT} \neq \mathbf{T}$.

Observe now that if we restrict the values of functions defining **L** connectives to the values T and F only, we get the functions defining the classical connectives. It is directly visible when we compare the **L** and classical connectives truth tables. This means that if $v^*(A) = T$ for all $v : VAR \longrightarrow \{F, \bot, T\}$, then $v^*(A) = T$ for any $v : VAR \longrightarrow \{F, T\}$ and for any $A \in \mathcal{F}$, i.e. $\mathbf{LT} \subset \mathbf{T}$.

Exercise 3.21

Use the fact that $v : VAR \longrightarrow \{F, \bot, T\}$ *is such that* $v^*((a \cap b) \Rightarrow \neg b) =\bot$ *under* **L** *semantics to evaluate* $v^*(((b \Rightarrow \neg a) \Rightarrow (a \Rightarrow \neg b)) \cup (a \Rightarrow b))$. *Use shorthand notation.*

Solution
Observe that $((a \cap b) \Rightarrow \neg b) =\bot$ in two cases.

c1: $(a \cap b) =\bot$ and $\neg b = F$.
c12: $(a \cap b) = T$ and $\neg b =\bot$.

Consider **c1**. We have $\neg b = F$, i.e. $b = T$, and hence $(a \cap T) =\bot$ if and only if $a =\bot$. We get that v is such that $v(a) =\bot$ and $v(b) = T$. We evaluate (in short hand notation) $v^*(((b \Rightarrow \neg a) \Rightarrow (a \Rightarrow \neg b)) \cup (a \Rightarrow b)) = (((T \Rightarrow \neg \bot) \Rightarrow (\bot \Rightarrow \neg T)) \cup (\bot \Rightarrow T)) = ((\bot \Rightarrow \bot) \cup T) = T$.

Consider **c2**. We have $\neg b =\bot$, i.e. $b =\bot$, and hence $(a \cap \bot) = T$ what is impossible, hence v from case **c1** is the only one, and $v^*(((b \Rightarrow \neg a) \Rightarrow (a \Rightarrow \neg b)) \cup (a \Rightarrow b)) = T$.

$\mathbf{L}_4$ Semantics

We define the semantics $\mathbf{L}_4$ as follows. The language is $\mathcal{L} = \mathcal{L}_{\{\neg, \Rightarrow, \cup, \cap\}}$. The logical connectives $\neg, \Rightarrow, \cup, \cap$ of $\mathbf{L}_4$ as the following operations in the set $\{F, \bot_1, \bot_2, T\}$, where $\{F < \bot_1 < \bot_2 < T\}$.

$\mathbf{L}_4$ **Negation** is a function such that $\neg\bot_1 = \bot_1$, $\neg\bot_2 = \bot_2$, $\neg F = T$, $\neg T = F$.

$\mathbf{L}_4$ **Conjunction** is a function such that for any $x, y \in \{F, \bot_1, \bot_2, T\}$, $x \cap x = min\{x, y\}$.

$\mathbf{L}_4$ **Disjunction** is a function such that for any $x, y \in \{F, \bot_1, \bot_2, T\}$, $x \cup y = min\{x, y\}$.

$\mathbf{L}_4$ **Implication** is a function such that for any $x, y \in \{F, \bot_1, \bot_2, T\}$,

$$x \Rightarrow y = \begin{cases} \neg x \cup y & \text{if } x > y \\ T & \text{otherwise} \end{cases} \tag{3.37}$$

Exercise 3.22

Here are 3 simple problems.

1. Write down $\mathbf{L}_4$ *Connectives Truth Tables.*

2. Give an example of a $\mathbf{L}_4$ *tautology.*

3. We know that the formula $((a \Rightarrow b) \Rightarrow (\neg a \cup b))$ *is a classical tautology, i.e.* $\models ((a \Rightarrow b) \Rightarrow (\neg a \cup b))$. *Verify whether* $\models_{L_4} ((a \Rightarrow b) \Rightarrow (\neg a \cup b))$.

Solution 1.
Here are $\mathbf{L}_4$ Connectives Truth Tables.

$\neg$	F	$\bot_1$	$\bot_2$	T
	T	$\bot_1$	$\bot_2$	F

$\cap$	F	$\bot_1$	$\bot_2$	T
F	F	F	F	F
$\bot_1$	F	$\bot_1$	$\bot_1$	$\bot_1$
$\bot_2$	F	$\bot_1$	$\bot_2$	$\bot_2$
T	F	$\bot_1$	$\bot_2$	T

$\cup$	F	$\bot_1$	$\bot_2$	T
F	F	$\bot_1$	$\bot_2$	T
$\bot_1$	$\bot_1$	$\bot_1$	$\bot_2$	T
$\bot_2$	$\bot_2$	$\bot_2$	$\bot_2$	T
T	T	T	T	T

$\Rightarrow$	F	$\bot_1$	$\bot_2$	T
F	T	T	T	T
$\bot_1$	$\bot_1$	T	T	T
$\bot_2$	$\bot_2$	$\bot_2$	T	T
T	F	$\bot_1$	$\bot_2$	T

Solution 2.
Observe that by definition of $\mathbf{L}_4$ implication we get $x \Rightarrow x = T$ for all $x \in \{F, \bot_1, \bot_2, T\}$. Hence $v^*((a \Rightarrow a)) = v(a) \Rightarrow v(a) = T$ for all v, what proves $\models_{\mathbf{L}_4} (a \Rightarrow a)$.

Solution 3.
We use the Proof by Contradiction Method (section 3.3) to verify whether $\models_{\mathbf{L}_4}((a \Rightarrow b) \Rightarrow (\neg a \cup b))$. Observe that it applied to any situation, as its correctness is based on our classical reasoning. Assume that $\not\models_{\mathbf{L}_4} ((a \Rightarrow b) \Rightarrow (\neg a \cup b))$. Let $v : VAR \longrightarrow \{F, \bot_1, \bot_2, T\}$, such that $v^*(((a \Rightarrow b) \Rightarrow (\neg a \cup b))) \neq T$. Observe that in $\mathbf{L}_4$ semantics, for any formula $A \in \mathcal{F}$, $v^*(A) \neq T$ gives us three possibilities $v^*(A) = F$, $v^*(A) = \bot_1$, or $v^*(A) = \bot_2$ to consider (as opposed to one case in classical case). It is a lot of work, but still less then listing and evaluating $4^2 = 16$ possibilities of all restricted truth assignment. Moreover, our formula is a classical tautology, hence we know that it evaluates in T for all combinations of T and F. A good strategy is to examine first some possibilities

of evaluating variables a, b for combination of $\bot_1, \bot_2$ with hope of finding a counter model. So let's v be a truth assignment such that $v(a) = v(b) = \bot_1$. We evaluate $v^*((a \Rightarrow b) \Rightarrow (\neg a \cup b)) = ((\bot_1 \Rightarrow \bot_1) \Rightarrow (\neg\bot_1 \cup \bot_1)) = (T \Rightarrow (\bot_1 \cup \bot_1)) = (T \Rightarrow \bot_1) = \bot_1$. This proves that v is a counter-model for our formula. Observe that the v serves also as a **L** counter model for A when we put $\bot_1$=$\bot$ and so we get

$$\models ((a \Rightarrow b) \Rightarrow (\neg a \cup b)),\ \not\models_{\mathbf{L_4}} ((a \Rightarrow b) \Rightarrow (\neg a \cup b)), \not\models_{\mathbf{L}}\ ((a \Rightarrow b) \Rightarrow (\neg a \cup b))$$

Obviously, any v such that $v(a) = v(b) = \bot_2$ is also a counter model for A, as $v^*((a \Rightarrow b) \Rightarrow (\neg a \cup b)) = ((\bot_2 \Rightarrow \bot_2) \Rightarrow (\neg\bot_2 \cup \bot_2)) = (T \Rightarrow (\bot_2 \cup \bot_2)) = (T \Rightarrow \bot_2) = \bot_2$. We leave it as an exercise to find all possible counter models for A.

Heyting Semantics H

Motivation

We discuss here the semantics **H** because of its connection with intuitionistic logic. The **H** connectives are such that they represent operations in a certain 3 element algebra, historically called a 3 element *pseudo-boolean* algebra. Pseudo-boolean algebras were created by McKinsey and Tarski in 1948 to provide semantics for the intuitionistic logic. The intuitionistic logic, the most important rival to the classical logic was defined and developed by its inventor Brouwer and his school in 1900s as a proof system only. Heyting provided its first axiomatization which everybody accepted. McKinsey and Tarski proved the completeness of the Heyting axiomatization with respect to their pseudo boolean algebras semantics. The pseudo boolean algebras are also called Heyting algebras in his honor and so is our semantics **H**.

We say, that formula A is an intutionistic tautology if and only if it is valid in *all pseudo boolean (Heying) algebras*. The pseudo boolean algebras are defined in a very general and mathematically sophisticated way. Their universe (it means the set of logical values) can be any non empty set. Their operations that correspond to $\mathcal{L} = \mathcal{L}_{\{\neg,\Rightarrow,\cup,\cap\}}$ connectives must fulfill a set of special properties (axioms). But we can prove that the operations defined by **H** connectives form a 3-element pseudo boolean algebra with the universe $U = \{F, \bot, T\}$. Hence, if A is an intuitionistic tautology, then in it is also valid (tautologically true) for the **H** semantics, i.e. all intuitionistic propositional tautologies are also the **H** semantics tautologies. It means that our **H** is a good candidate for finding counter models for the formulas that might not be intuitionistic tautologies.

The other type of models, called Kripke models were defined by Kripke in 1964 and were proved later to be equivalent to the pseudo-boolean models. They are very general and serve as a method of defining *not extensional* semantics for various classes of logics. That includes semantics for a great number of modal, knowledge, belief logics, and many new logics developed and being developed by computer scientists.

H Connectives

We adopt the same language as in case of classical and Łukasiewicz's **L** semantics, i.e. $\mathcal{L} = \mathcal{L}_{\{\neg, \Rightarrow, \cup, \cap\}}$. We assume, as before, that $\{F < \perp < T\}$.

The connectives $\neg$, $\cup$, $\cap$ of **H** are defined as in **L** semantics. They are functions defined by formulas $x \cup y = max\{x, y\}$, $x \cap y = min\{x, y\}$, for any $x, y \in \{F, \perp, T\}$.

The definition of implication and negation for **H** semantics differs **L** semantics and we define them as follows.

H Implication is a function $\Rightarrow$: $\{T, \perp, F\} \times \{T, \perp, F\} \longrightarrow \{T, \perp, F\}$ such that for any $(x, y) \in \{T, \perp, F\} \times \{T, \perp, F\}$, we put

$$x \Rightarrow y = \begin{cases} T & \text{if } x \leq y \\ y & \text{otherwise} \end{cases} \tag{3.38}$$

H negation is a function $\neg$: $\{F, \perp, T\} \longrightarrow \{F, \perp, T\}$, such that

$$\neg a = a \Rightarrow F.$$

The truth tables for **H** disjunction and conjunction are hence the same as corresponding **L** tables and the truth tables for **H** implication and negation are as follows.

$\Rightarrow$	F	$\perp$	T
F	T	T	T
$\perp$	F	T	T
T	F	$\perp$	T

$\neg$	F	$\perp$	T
	T	*F*	F

For **Steps 2 - 4** of the definition 3.7 we adopt definitions established for **L** semantics. For example, we define the notion of **H** tautology as follows.

Definition 3.36 (H Tautology)

For any formula $A \in \mathcal{F}$,
A is a **H** *tautology if and only if* $v^*(A) = T$, *for all* $v : VAR \longrightarrow \{F, \perp, T\}$, *i.e.* $v \models_{\mathbf{H}} A$ *for all v. We write*

$$\models_{\mathbf{H}} A$$

to denote that a formula A is an **H** *tautology.*

We leave it as an exercise to the reader to prove, in the same way as in case of classical semantics (section3.3) the following theorems that justify the truth table method of verification and the decidability theorem for **H**.

Theorem 3.16 (H Truth Tables)

For any formula $A \in \mathcal{F}$,
$\models_{\mathbf{H}} A$ *if and only if* $v_A \models_{\mathbf{H}} A$ *for all* $v_A : VAR_A \longrightarrow \{T, \perp, F\}$, *i.e.*
$\models_{\mathbf{H}} A$ *if and only if all v_A are restricted models for A.*

Theorem 3.17 (H Decidability)

For any formula $A \in \mathcal{F}$, one has examine at most 3^{VAR_A} truth assignments $v_A : VAR_A \longrightarrow \{F, \perp, T\}$ in order to decide whether $\models_{\mathbf{H}} A$, or $\not\models_{\mathbf{H}} A$, i.e. the notion of **H** *tautology is decidable.*

We denote by **HT** the set of all **H** tautologies, i.e.

$$\mathbf{HT} = \{A \in \mathcal{F} : \models_{\mathbf{H}} A\}.$$

The following fact establishes relationship between classical and **H** tautologies.

Theorem 3.18

Let **HT**, **LT**, **T** *denote the sets of all* **H**, **L**, *and classical tautologies, respectively. Then the following relationship holds.*

$$\mathbf{HT} \neq \mathbf{LT}, \quad \mathbf{HT} \neq \mathbf{T}, \quad \textit{and} \quad \mathbf{HT} \subset \mathbf{T}. \tag{3.39}$$

Proof
A formula $(\neg a \cup a)$ a classical tautology and not an **H** tautology. Take any truth assignment $v : VAR \longrightarrow \{F, \perp, T\}$ such that $v(a) = \perp$. We evaluate is $v^*((\neg a \cup a) = \neg \perp \cup \perp = F \cup \perp = \perp$ This proves that $(\neg a \cup a) \notin$ **HT** and hence **HT** $\neq$ **T**. Directly from the definition of **H** connectives we get that if we restrict the values of the functions defining them T and F only, we get the functions defining the classical connectives. Hence for any formula $A \in$ **TH** we have that $A \in$ **TH** and **LT** $\subset$ **T**. A formula $(\neg\neg a \Rightarrow a)$ is a **L** tautology and not an **H** tautology by easy evaluation as presented in example 3.23 and (3.40). This proves **HT** $\neq$ **LT**.

Exercise 3.23

We know that $v : VAR \longrightarrow \{F, \perp, T\}$ is such that $v^*((a \cap b) \Rightarrow (a \Rightarrow c)) = \perp$ under **H** *semantics.*

Evaluate $v^*(((b \Rightarrow a) \Rightarrow (a \Rightarrow \neg c)) \cup (a \Rightarrow b))$. You can use a short hand notation.

Solution
By definition of **H** connectives we have that for any v, $v^*((a\cap b) \Rightarrow (a \Rightarrow c)) =\perp$ if and only if $a \cap b) = T$ and $(a \Rightarrow c) =\perp$ if and only if $a = T, b = T$ and $(T \Rightarrow c) =\perp$ if and only if $c =\perp$. Hence $v^*((a \cap b) \Rightarrow (a \Rightarrow c)) =\perp$ if and only if $a = T,\ b = T, c =\perp$. We evaluate $v^*(((b \Rightarrow a) \Rightarrow (a \Rightarrow \neg c)) \cup (a \Rightarrow b)) = (((T \Rightarrow T) \Rightarrow (T \Rightarrow \neg \perp)) \cup (T \Rightarrow T)) = ((T \Rightarrow (T \Rightarrow F)) \cup T) = T$.

Exercise 3.24

We know that the following formulas are basic classical tautologies

$$\models (a \cup \neg a), \qquad \models (\neg\neg a \Rightarrow a), \qquad \models ((a \Rightarrow b) \Rightarrow (\neg a \cup b)). \tag{3.40}$$

Use the **H** *semantics to prove that none of them is intuitionostic tautology.*

Solution Any $v : VAR \longrightarrow \{F,\ \perp,\ T\}$ such that $v(a) = v(b) =\perp$ is an **H** counter model for all of the formulas. We evaluate (in shorthand notation) it as follows. $\perp \cup \neg \perp = \perp \cup F = \perp \neq T$, $\neg\neg \perp \Rightarrow \perp = \neg F \Rightarrow \perp = T \Rightarrow \perp = \perp \neq T$,

$(\perp \Rightarrow \perp) \Rightarrow (\neg \perp \cup \perp) = T \Rightarrow (\neg \perp \cup \perp) = T \Rightarrow (F \cup \perp) = T \Rightarrow \perp = \perp \neq T$. We hence proved by the fact *"if a given formula A is not the* **H** *semantics tautology, it is not intuitionistic tautology"* that none of classical tautologies (3.40) is neither intuitionostic nor **H** tautology.

The **H** semantics can serve as a tool of proving that some formulas are not intutionistic tautologies, but it is not a universal one

Example 3.22

We know that the classical tautology $(\neg(a \cap b) \Rightarrow (\neg a \cup \neg b))$ *is not intuitionistic tautology, but nevertheless* $\models_{\mathbf{H}}(\neg(a \cap b) \Rightarrow (\neg a \cup \neg b))$.

Proof
We use the Proof by Contradiction Method (section 3.3.1) and shorthand notation. Assume that $\not\models_{\mathbf{H}} (\neg(a \cap b) \Rightarrow (\neg a \cup \neg b))$. Let $v : VAR \longrightarrow \{F,\ \perp,\ T\}$ such that $v^*((\neg(a \cap b) \Rightarrow (\neg a \cup \neg b))) \neq T$. We have to consider two cases: **c1** $v^*((\neg(a \cap b) \Rightarrow (\neg a \cup \neg b))) =\perp$ and **c2** $v^*((\neg(a \cap b) \Rightarrow (\neg a \cup \neg b))) = F$. If we get a contradiction in *both cases* we have proved $\models_{\mathbf{H}}(\neg(a\cap b) \Rightarrow (\neg a\cup\neg b))$.

Consider case **c1**. By definition of $\Rightarrow$ we have that $v^*((\neg(a\cap b) \Rightarrow (\neg a\cup\neg b))) =\perp$ if and only if $\neg(a\cap b) = T$ and $\neg a\cup\neg b =\perp$ if and only if $a\cap b = F$ and $\neg a\cup\neg b =\perp$. Let's look $\neg a \cup \neg b =\perp$. This is possible in 3 cases. 1. $\neg a =\perp$ and $\neg b =\perp$. Contradiction with the definition of $\perp$ as $\neg x \neq\perp$ for all $x \in \{F, \perp, T\}$. 2. $\neg a =\perp$ and $\neg b = F$. Contradiction with the definition of $\perp$. 3. $\neg a = F$ and $\neg b =\perp$.

Contradiction with the definition of $\perp$. This proves that case **c1** always leads to contradiction.

Consider case **c2**. By definition of $\Rightarrow$ we have that $v^*((\neg(a \cap b) \Rightarrow (\neg a \cup \neg b))) = F$ if and only if 1. $\neg(a \cap b) = \perp$, $\neg a \cup \neg b = F$. Contradiction. 2. $\neg(a \cap b) = T$ and $\neg a \cup \neg b = F$ if and only if $a \cap b = F$ and $\neg a \cup \neg b = F$. Observe that $a \cap b = F$ in 3 cases. Two involve only T, F and we get a contradiction as in classical case (our formula is classical tautology). We have hence to consider only the cases when $a = \perp, b = F$ and $a = F, b = \perp$. They both lead to the contradiction with $\neg a \cup \neg b = F$. This proves that case **c2** always leads to contradiction and it ends the proof.

We can of course also use the Truth Tables Method that involves listing and evaluating all of $2^3 = B$ restricted truth assignments.

Kleene Semantics K

Kleene's logic semantics was originally conceived to accommodate undecided mathematical statements.

Motivation

In Kleene's semantics the third logical value $\perp$, intuitively, represents *undecided.* Its purpose is to signal a state of partial ignorance. A sentence a is assigned a value $\perp$ just in case it is not *known* to be either true of false.

For example, imagine a detective trying to solve a murder. He may conjecture that Jones killed the victim. He cannot, at present, assign a truth value T or F to his conjecture, so we assign the value $\perp$, but it is certainly either true of false and $\perp$ represents our ignorance rather then total unknown.

K Connectives

We adopt the same language as in a case of classical, Łukasiewicz's **L**, and Heyting **H** semantics, i.e. $\mathcal{L} = \mathcal{L}_{\{\neg, \Rightarrow, \cup, \cap\}}$.

We assume, as before, that $\{F < \perp < T\}$. The connectives $\neg$, $\cup$, $\cap$ of **K** are defined as in **L, H** semantics. They are functions defined by formulas $x \cup y = max\{x, y\}$, $x \cap y = min\{x, y\}$, for any $x, y \in \{F, \perp, T\}$, and

$$\neg \perp = \perp, \quad \neg F = T, \quad \neg T = F.$$

The **K** implication is defined by the same formula as the classical, i.e.

$$x \Rightarrow y = \neg x \cup y. \tag{3.41}$$

for any $x,\ y \in \{F, \perp, T\}$.

The connectives truth tables for the **K** negation, disjunction and conjunction are the same as the corresponding tables for **L, H** and **K** implication table is as follows.

$\Rightarrow$	F	$\perp$	T
F	T	T	T
$\perp$	$\perp$	$\perp$	T
T	F	$\perp$	T

For **Steps 2 - 4** of the definition of **K** semantics we follow the general **M** semantics definition 3.7, or adopt its particular case of **L** semantics definition. For example, we define the notion of **K** tautology as follows.

Definition 3.37 (K Tautology)

For any formula $A \in \mathcal{F}$,
*A is a **K** tautology if and only if $v^*(A) = T$, for all truth assignments $v : VAR \longrightarrow \{F, \perp, T\}$, i.e. $v \models_{\mathbf{K}} A$ for all v.*

We write

$$\models_{\mathbf{K}} A$$

to denote that A is a **K** tautology. We prove, in the same way as in case of **L** semantics the following theorems that justify truth table method of verification and decidability theorem for **K**.

Theorem 3.19 (K Truth Tables)

For any formula $A \in \mathcal{F}$,
$\models_{\mathbf{K}} A$ if and only if $v_A \models_{\mathbf{K}} A$ for all $v_A : VAR_A \longrightarrow \{T, \perp, F\}$, i.e.
$\models_{\mathbf{K}} A$ if and only if all v_A are restricted models for A.

Directly from Theorem 3.19 we get that the notion of **K** propositional tautology is decidable, i.e. that the following holds.

Theorem 3.20 (K Decidability)

*For any formula $A \in \mathcal{F}$, one has examine at most 3^{VAR_A} truth assignments $v_A : VAR_A \longrightarrow \{F, \perp, T\}$ in order to decide whether $\models_{\mathbf{L}} A$, or $\not\models_{\mathbf{K}} A$, i.e. the notion of **K** tautology is decidable.*

We write

$$\mathbf{KT} = \{A \in \mathcal{F} : \ \models_K A\}$$

to denote the set of all **K** tautologies. The following establishes relationship between **L**, **K**, and classical tautologies.

Theorem 3.21

Let **LT**, **T**, **KT** *denote the sets of all* **L**, *classical, and* **K** *tautologies, respectively. Then the following relationship holds.*

$$\mathbf{LT} \neq \mathbf{KT}, \quad \mathbf{KT} \neq \mathbf{T}, \ \textit{and} \ \ \mathbf{KT} \subset \mathbf{T}. \tag{3.42}$$

Proof
Obviously $\models \ (a \Rightarrow a)$ and also by (3.35) $\models_{\mathbf{L}} (a \Rightarrow a)$. Consider now any v such that $v(a) = \perp$. We evaluate in **K** semantics $v^*(a \Rightarrow a) = v(a) \Rightarrow v(a) = \perp \Rightarrow \perp = \perp$. This proves that $\not\models_{\mathbf{K}} (a \Rightarrow a)$ and hence the first two relationships in (3.42) hold. The third one follows directly from the the fact that, as in the **L** case, if we restrict the functions defining **K** connectives to the values T and F only, we get the functions defining the classical connectives.

Exercise 3.25

We know that formulas $((a \cap b) \Rightarrow a), (a \Rightarrow (a \cup b)), (a \Rightarrow (b \Rightarrow a))$ *are classical tautologies. Show that none of them is* **K** *tautology.*

Solution Consider any v such that $v(a) = v(b) = \perp$. We evaluate (in short hand notation) $v^*(((a \cap b) \Rightarrow a) = (\perp \cap \perp) \Rightarrow \perp = \perp \Rightarrow \perp = \perp \Rightarrow \perp \perp$, $v^*((a \Rightarrow (a \cup b))) = \perp \Rightarrow (\perp \cup \perp) = \perp \Rightarrow \perp = \perp$, and $v^*((a \Rightarrow (b \Rightarrow a))) = (\perp \Rightarrow (\perp \Rightarrow \perp) = \perp \Rightarrow \perp = \perp$. This proves that v such that $v(a) = v(b) = \perp$ is a counter model for all of them. We generalize this example and prove that in fact a similar truth assignment can serve as a counter model for not only any classical tautology, but also for any formula A of $\mathcal{L}_{\{\neg, \Rightarrow, \cup, \cap\}}$.

Theorem 3.22

For any formula $A \in \mathcal{F}$, $\not\models_{\mathbf{K}} A$, *i.e. the set of all* **K** *tautologies is empty. We write it as*

$$\mathbf{KT} = \emptyset.$$

Proof
We show that a truth assignment $v : VAR \longrightarrow \{F, \perp, T\}$, such that $v(a) = \perp$ for all $a \in VAR$ is a counter model for any $A \in \mathcal{F}$. We carry the proof the by mathematical induction over the degree $d(A)$ of the formula A.

Base Case: n=1 i.e. $d(A) = 1$. In this case we have that $A = \neg a$ for any $a \in VAR$, or $A = (a \circ b)$ for $a, b \in VAR, \circ \in \{\cup, \cap, \Rightarrow\}$.

We evaluate: $v^*(A) = v^*(\neg a) = \neg v^*(a) = \neg \perp = \perp$, $v^*(a \circ b) = v^*(a) \circ v^*(b) = \perp \circ \perp = \perp$. This proves that the *Base Case* holds.

Inductive assumption: $v^*(B) = \perp$ for all B such that $d(B) = k$ and $1 \leq k < n$.
Inductive thesis: $v^*(A) = \perp$ for any A such that $d(A) = n$.

Let A be such that $d(A) = n$. We have two cases to consider.

Case 1. $A = \neg B$, so $d(B) = n - 1 < n$. By inductive assumption $v^*(B) = \perp$. Hence $v^*(A) = v^*(\neg B) = \neg v^*(B) = \neg \perp = \perp$ and inductive thesis holds.

Case 2. $A = (B \circ C)$ for $B, C \in \mathcal{F}, \circ \in \{\cup, \cap, \Rightarrow\}$ (and $d(A) = n$). Let $d(B) = k_1, d(C) = k_2$. Hence $d(A) = d(B \circ C) = k_1 + k_2 + 1 = n$. We get that $k_1 + k_2 = n - 1 < n$. From $k_1 + k_2 < n$ we get that $k_1 < n$ and $k_2 < n$. Hence by inductive assumption $v^*(B) = \perp$ and $v^*(C) = \perp$. We evaluate: $v^*(A) = v^*(B \circ C) = v^*(B) \circ v^*(C) = \perp \circ \perp = \perp$. This ends the proof.

Observe that the theorem 3.22 does not invalidate relationships (3.42). They become now perfectly true statements

$$\mathbf{LT} \neq \emptyset, \quad \mathbf{T} \neq \emptyset, \text{ and } \emptyset \subset \mathbf{T}.$$

But when we develop a logic by defining its semantics we must make sure for semantics to be such that it has a non empty set of its tautologies. The semantics **K** is an example of a correctly and carefully defined semantics that is not well defined in terms of the definition 3.14. We write is as separate fact.

Fact 3.6

The Kleene semantics **K** *is* **not well defined**.

K semantics also provides a justification for a need of introducing the definition 3.14 as a distinction between correctly and well defined semantics. This is the main reason why it is included here.

Bochvar semantics B

Bochvar's 3-valued logic was directly inspired by considerations relating to semantic paradoxes. Here is the motivation for definition of its semantics.

Motivation

Consider a semantic paradox given by a sentence: *this sentence is false.* If it is true it must be false, if it is false it must be true. There have been many proposals relating to how one may deal with semantic paradoxes. Bohvar's proposal adopts a strategy of a change of logic. According to Bochvar, such sentences are neither true of false but rather *paradoxical* or *meaningless*. The semantics follows the principle that the third logical value, denoted now by m is in some sense "infectious"; if one one component of the formula is assigned the value m then the formula is also assigned the value m.

Bohvar also adds an one argument *assertion operator* S that asserts the logical value of T and F, i.e. $SF = F$, $ST = T$ and it asserts that meaningfulness is false, i.e $Sm = F$.

Language $\mathcal{L}_{\mathbf{B}}$

The language of **B** semantics differs from all previous languages in that it contains an extra one argument assertion connective S added to the usual set $\{\neg, \Rightarrow, \cup, \cap\}$ of the language $\mathcal{L} = \mathcal{L}_{\{\neg, S, \Rightarrow, \cup, \cap\}}$ of all previous semantics.

$$\mathcal{L}_{\mathbf{B}} = \mathcal{L}_{\{\neg, S, \Rightarrow, \cup, \cap\}}. \tag{3.43}$$

The set LV of logical values is $\{T,\ m,\ F\}$. T is the distinguished value.

B Connectives

We define the connectives of $\mathcal{L}_{\mathbf{B}}$ the functions defined in the set $\{F, mT\}$ by the following truth tables.

B Connectives Truth Tables

$\neg$	F	m	T
	T	m	F

$\cap$	F	m	T
F	F	m	F
m	m	m	m
T	F	m	T

$\cup$	F	m	T
F	F	m	T
m	m	m	m
T	T	m	T

$\Rightarrow$	F	m	T
F	T	m	T
m	m	m	m
T	F	m	T

S	F	m	T
	F	F	T

For all other steps of definition of **B** semantics we follow the standard way established for extensional **M** semantics, we did in all previous cases. In particular we define the notion of **B** tautology as follows.

Definition 3.38

A formula A of $\mathcal{L}_{\mathbf{B}}$ is a **B** *tautology if and only if* $v^*(A) = T$, *for all*

$v : VAR \longrightarrow \{F, m, T\}$, i.e. if all variable assignments v are **B** *models for A.*

We write

$$\models_{\mathbf{B}} A$$

to denote that A is an **B** tautology.

We, prove, in the same way as for all previous logics semantics, the following theorems that justify the truth table method of verification and decidability for **B** tautologies.

Theorem 3.23 (B Truth Tables)

For any formula A of $\mathcal{L}_{\mathbf{B}}$,
$\models_{\mathbf{B}} A$ if and only if $v_A \models_{\mathbf{B}} A$ for all $v_A : VAR_A \longrightarrow \{F, m, T\}$.

Theorem 3.24 (Decidability)

For any formula A of $\mathcal{L}_{\mathbf{B}}$, one has examine at most 3^{VAR_A} truth assignments $v : VAR_A \longrightarrow \{F, m, T\}$ in order to decide whether $\models_{\mathbf{B}} A$, or $\not\models_{\mathbf{B}} A$, i.e. the notion of **B** *tautology is decidable.*

Let denote by $\mathcal{F}_{\mathbf{B}}$ the set of formulas of the language $\mathcal{L}_{\mathbf{B}}$ and by **BT** the set of all **B** tautologies:

$$\mathbf{BT} = \{A \in \mathcal{F}_{\mathbf{B}} : \ \models_{\mathbf{B}} \ A\}.$$

Which formulas (if any) are the **B** tautologies is more complicated to determine then in the case previous semantics because we have the following Fact 3.7.

Fact 3.7

For any formula $A \in \mathcal{F}_{\mathbf{B}}$ which do not contain a connective S, i.e. for any formula A of the language $\mathcal{L}_{\{\neg, \Rightarrow, \cup, \cap\}}$, $\not\models_{\mathbf{B}} A$.

Proof We show that a truth assignment $v : VAR \longrightarrow \{F, m, T\}$, such that $v(a) = m$ for all $a \in VAR$ is a counter model for any $A \in \mathcal{F}$. The proof the by mathematical induction over the degree $d(A)$ of the formula A is similar to the proof of Theorem 3.22 and is left to the reader as an exercise.

By the Fact 3.7 for a formula to be considered to be a **B** tautology, it must contain the connective S. We get by easy evaluation that $\models_{\mathbf{B}} (Sa \cup \neg Sa)$. This proves that $\mathbf{BT} \neq \emptyset$ and the **B** semantics is well defined by definition 3.14. Of course not all formulas containing the connective S are **B** tautologies, for example

$$\not\models_{\mathbf{B}} (a \cup \neg Sa), \ \not\models_{\mathbf{B}} (Sa \cup \neg a), \ \not\models_{\mathbf{B}} (Sa \cup S\neg a),$$

as any truth assignment v, such that $v(a) = m$ is a counter model for all of them, because $m \cup x = m$ for all $x \in \{F, m, T\}$ and $Sm \cup S\neg m = F \cup Sm = F \cup F = F$.

3.6 M Tautologies, M Consistency, and M Equivalence of Languages

The classical truth tables verification method a and classical decidability theorem hold in a proper form in all of **L. H, K** and **B** semantics as it was discussed separately for each of them. We didn't discuss other classical tautologies verification methods of substitution and generalization. We do it now in a general and unifying way for a special case of an extensional **M** semantics, namely for any semantics **M** with a **finite** set of logical values.

3.6.1 M Tautologies Verification Methods

Given an extensional semantics **M** defined for a propositional language $\mathcal{L}_{CON}$ with the set $\mathcal{F}$ of formulas and a **finite**, non empty set LV of logical values. We introduce, as we did in classical, and other cases a notion of a restricted model (definition 3.26) and prove, in a similar way as we proved theorem 3.8 the following theorem that justifies the correctness of the **M** truth tables tautologies verification method.

Theorem 3.25 (M Truth Tables)

For any formula $A \in \mathcal{F}$,
$\models_{\mathbf{M}} A$ *if and only if* $v_A \models_{\mathbf{M}} A$ *for all* $v_A : VAR_A \longrightarrow LV$, *i.e.*
$\models_{\mathbf{M}} A$ *if and only if all* v_A *are restricted models for* A.

M Truth Table Method

A verification method, called a **M** truth table method consists of examination, as in the classical case, for any formula A, all possible **M** truth assignments restricted to A. By theorem 3.25 we have to perform at most $|LV|^{|VAR_A|}$ steps. If we find a restricted truth assignment which evaluates A to a value different then T, we stop the process and give answer: $\not\models_{\mathbf{M}} A$. Otherwise we continue. If all **M** truth assignments restricted to A evaluate A to T, we give answer: $\models_{\mathbf{M}} A$.

Example 3.23

Consider a formula $(\neg\neg a \Rightarrow a)$ *and* **H** *semantics. We evaluate*

v	a	$v^*(A)$ *computation*	$v^*(A)$
v_1	T	$\neg\neg T \Rightarrow T = \neg F \Rightarrow T = F \Rightarrow T = T$	T
v_2	$\perp$	$\neg\neg \perp \Rightarrow \perp = \neg F \Rightarrow \perp = T \Rightarrow \perp = \perp$	$\perp$

It proves that $\not\models_{\mathbf{H}} (\neg\neg a \Rightarrow a)$.

Example 3.24

Consider a formula $(\neg\neg a \Rightarrow a)$ *and* **L** *semantics. We evaluate*

v	a	$v^*(A)$ *computation*	$v^*(A)$
v_1	T	$\neg\neg T \Rightarrow T = \neg F \Rightarrow T = F \Rightarrow T = T$	T
v_2	$\perp$	$\neg\neg \perp \Rightarrow \perp = \neg \perp \Rightarrow \perp = \perp \Rightarrow \perp = T$	T
v_3	F	$\neg\neg F \Rightarrow F = \neg T \Rightarrow F = F \Rightarrow F = T$	T

It proves that $\models_{\mathbf{L}} (\neg\neg a \Rightarrow a)$.

We also proved that the set **HT** of all **H** tautologies is different from the set set **LT** of all **L** tautologies, i.e.

$$\mathbf{LT} \neq \mathbf{HT} \tag{3.44}$$

Directly from Theorem 3.25 and the above we get that the notion of **M** propositional tautology is decidable, i.e. that the following holds.

Theorem 3.26 (M Decidability)

For any formula $A \in \mathcal{F}$, *one has examine at most* $|LV|^{VAR_A}$ *truth assignments* $v_A : VAR_A \longrightarrow LV$ *in order to decide whether* $\models_{\mathbf{M}} A$, *or* $\not\models_{\mathbf{M}} A$, *i.e. the notion of* **M** *tautology is decidable.*

M Proof by Contradiction Method

In this method, in order to prove that $\models_{\mathbf{M}} A$ we proceed as follows. We assume that $\not\models_{\mathbf{M}} A$. We work with this assumption. If we get a **contradiction**, we have proved that $\not\models_{\mathbf{M}} A$ is impossible. We hence proved $\models_{\mathbf{M}} A$. If we do not get a contradiction, it means that the assumption $\not\models_{\mathbf{M}} A$ is true, i.e. we have proved that A is not **M** tautology.
Observe that correctness of his method is based on a correctness of classical reasoning. Its correctness is based on the *Reductio ad Absurdum* classical tautology $\models ((\neg A \Rightarrow (B \cap \neg B)) \Rightarrow A)$. The contradiction to be obtained follows from the properties of the **M** semantics under consideration.

Substitution Method
The Substitution Method allows us to obtain, as in a case of classical semantics new **M** tautologies from formulas already proven to be **M** tautologies. The theorem 3.27 and its proof is a straightforward modification of the classical proof (theorem 3.27) and we leave it as an exercise to the reader. It assesses the validity of the substitution method. In order to formulate and prove it we first remind of the reader of needed notations.

Let $A \in \mathcal{F}$ be a formula and $VAR_A = \{a_1, a_2, ...a_n\}$ be the set of all propositional variables appearing in A. We will denote it by $A(a_1, a_2, ...a_n)$. Given a formula $A(a_1, a_2, ...a_n)$, and $A_1, ...A_n$ be any formulas. We denote by

$$A(a_1/A_1, ..., a_n/A_n)$$

the result of simultaneous replacement (substitution) in $A(a_1, a_2, ...a_n)$ variables $a_1, a_2, ...a_n$ by formulas $A_1, ...A_n$, respectively.

Theorem 3.27

For any formulas $A(a_1, a_2, ...a_n),\ A_1,\ \dots\ ,\ A_n \in \mathcal{F}$,

If $\models_{\mathbf{M}} A(a_1, a_2, ...a_n)$ *and* $B = A(a_1/A_1, ..., a_n/A_n)$, *then* $\models_{\mathbf{M}} B$.

We have proved (exercise 3.24) that the formula $D(a) = (\neg\neg a \Rightarrow a)$ is **L** tautology. By the above theorem 3.27 we get that $D(a/A) = (\neg\neg A \Rightarrow A)$ is also **L** tautology for any formula $A \in \mathcal{F}$. We hence get the following.

Fact 3.8

For any $A \in \mathcal{F}$, $\models_{\mathbf{L}} (\neg\neg A \Rightarrow A)$.

M Generalization Method

In this method we represent, if it is possible, a given formula as a particular case of some simpler *general formula*. Hence the name Generalization Method. We then use other methods to examine the simpler formula thus obtained.

Exercise 3.26

Prove that

$\models_{\mathbf{L}} (\neg\neg(\neg((a \cap \neg b) \Rightarrow ((c \Rightarrow (\neg f \cup d)) \cup e)) \Rightarrow ((a \cap \neg b) \cap (\neg(c \Rightarrow (\neg f \cup d)) \cap \neg e))) \Rightarrow (\neg((a \cap \neg b) \Rightarrow ((c \Rightarrow (\neg f \cup d)) \cup e)) \Rightarrow ((a \cap \neg b) \cap (\neg(c \Rightarrow (\neg f \cup d)) \cap \neg e))))$.

Solution
Observe that our formula is a particular case of a more general formula $(\neg\neg A \Rightarrow A)$ for $A = (\neg((a \cap \neg b) \Rightarrow ((c \Rightarrow (\neg f \cup d)) \cup e)) \Rightarrow ((a \cap \neg b) \cap (\neg(c \Rightarrow (\neg f \cup d)) \cap \neg e)))$ and by fact 3.8 our formula is proved to be **L** tautology.

3.6.2 M Consistency

One of the most important notions for any logic are notions of consistency and inconsistency. We introduced and discussed them in case of classical semantics in section 3.3. We formulate them now for any **M** extensional semantics and examine them in cases of **L** and **H** semantics.

Consider $\mathcal{L}_{CON}$ and let $\mathcal{S} \neq \emptyset$ be any non empty set of formulas of $\mathcal{L}_{CON}$. Let $\mathbf{M}$ be an extensional semantics for $\mathcal{L}_{CON}$. We adopt the following definitions.

Definition 3.39 *A truth truth assignment* $v : VAR \longrightarrow LV$ *is a* **M model** *for the set* $\mathcal{G}$ *of formulas if and only if* $v \models_{\mathbf{M}} A$ *for all formulas* $A \in \mathcal{G}$.

We denote it by $v \models_{\mathbf{M}} \mathcal{G}$.

Definition 3.40 *A set* $\mathcal{G} \subseteq \mathcal{F}$ *is called* ***M consistent*** *if and only if there is* $v : VAR \longrightarrow LV$, *such that* $v \models_{\mathbf{M}} \mathcal{G}$. *Otherwise the set* $\mathcal{G}$ *is called* ***M inconsistent***.

Observe that the definition of inconsistency can be stated as follows.

Definition 3.41 *A set* $\mathcal{G} \subseteq \mathcal{F}$ *is called* ***M inconsistent*** *if and only if for all* $v : VAR \longrightarrow LV$ *there is a formula* $A \in \mathcal{G}$, *such that* $v^*(A) \neq T$.

Plainly speaking, a set $\mathcal{G}$ is consistent if it has a model, and is inconsistent if it does not have a model under a semantic $\mathbf{M}$.

Exercise 3.27

Prove that the set

$$\mathcal{G} = \{((a \cap b) \Rightarrow b),\ (a \cup b), \neg a\}$$

is **L**, **H**, *and* **K** *consistent.*

Solution Consider a truth assignment $v : VAR \longrightarrow \{T, \perp, F\}$. By the definition 3.40, v must be such that $v^*(((a \cap b) \Rightarrow b)) = T$, $v^*((a \cup b) = T)$, and $v^*(\neg a) = T$. We want to prove that such v exists. Observe that $((a \cap b) \Rightarrow b)$ is classical tautology, so let's try to find $v : VAR \longrightarrow \{T, F\}$ such that $v^*((a \cup b)) = T$, $v^*(\neg a) = T$. This holds when $v(a) = F$ and hence $F \cup v(b) = T$. This gives us v such that $v(a) = F$ and $v(b) = T$. We proved that the connectives of **L**, **H**, and **K** semantics when restricted to the values T and F become classical connectives. Hence any v such that $v(a) = F$ and $v(b) = T$ is a **L**, **H**, and **K** model for $\mathcal{G}$.

The same argument prove the following general fact.

Fact 3.9

For any non empty set $\mathcal{G}$ *of formulas of a language* $\mathcal{L}_{\{\neg, \Rightarrow, \cup, \cap\}}$,
if $\mathcal{G}$ *is consistent under classical semantics, then it is* **L**, **H**, *and* **K** *consistent.*

Exercise 3.28

Give an example of an infinite set $\mathcal{G}$ of formulas of a language $\mathcal{L}_{\mathbf{B}} = \mathcal{L}_{\{\neg, S, \Rightarrow, \cup, \cap\}}$ that is **L**, **H**, **K** *and* **B** *consistent.*

Solution
Observe that for the set $\mathcal{G}$ to be considered to be **L**, **H**, **K** consistent its formulas must belong to the sub language $\mathcal{L}_{\{\neg, \Rightarrow, \cup, \cap\}}$ of the language $\mathcal{L}_{\mathbf{B}}$. Let's take, for example a set

$$\mathcal{G} = \{(a \cup \neg b) : \quad a, b \in VAR\}.$$

$\mathcal{G}$ is infinite since the set VAR is infinite. Consider any $v : VAR \longrightarrow \{F, m, T\}$ or $v : VAR \longrightarrow \{F, \perp, T\}$ such that $v(a) = T, v(b) = F$, we have $v^*(a \cup b) = v(a) \cup v(b) = T \cup T = T$ in all semantics **L**, **H**, **K** and **B**. This proves that $\mathcal{G}$ is **L**, **H**, **K** and **B** consistent.

Exercise 3.29

Prove that the set

$$\mathcal{G} = \{(a \cap \neg a) : \quad a \in VAR\}$$

is **L**, **H**, **K**, *and* **B** *inconsistent..*

Solution
We know that the set $\mathcal{G}$ is classically inconsistent, i.e. $v^*((a \cap \neg a)) \neq T$ for all $v : VAR \longrightarrow \{F, T\}$ under classical semantics. It also holds for We have to show that it also holds for **L**, **H**, **K** and **B** semantics when we restrict the functions defining their connectives to the values T and F only. In order to prove inconsistency under **L**, **H**, **K**, semantics we have to show that $v^*((a \cap \neg a)) \neq T$ for all $v : VAR \longrightarrow \{F, \perp, T\}$ under the respective semantics, i.e. we have to evaluate additional case $v(a) = \perp$ in all of them. Observe that negation $\neg$ is defined in all of them as $\neg \perp = \perp$, and $v^*((a \cap \neg a)) = \perp \cap \neg \perp = \perp \cap \perp = \perp \neq T$. This proves that $\mathcal{G}$ is **L**, **H**, and **K** inconsistent. The case of **B** semantics is similar, except that now we consider all $v : VAR \longrightarrow \{F, m, T\}$ and the additional case is $v(a) = m$. By definition $\neg m = m$ and $v^*((a \cap \neg a)) = m \cap m = m \neq T$.

The examples of **B** consistent, or inconsistent sets $\mathcal{G}$ in exercise 3.28 and exercise 3.29 were restricted to formulas from $\mathcal{L}_{\mathbf{B}} = \mathcal{L}_{\{\neg, S, \Rightarrow, \cup, \cap\}}$ that did not include the connective S. In this sense they were not characteristic to the semantics **B**. We pose hence a natural question whether such examples exist.

Exercise 3.30

Give an example of sets $\mathcal{G}_1, \mathcal{G}_2$ containing some formulas that include the S connective of the language $\mathcal{L}_{\mathbf{B}} = \mathcal{L}_{\{\neg, S, \Rightarrow, \cup, \cap\}}$ such that $\mathcal{G}_1$ is **B** *consistent and $\mathcal{G}_2$ is* **B** *inconsistent*

Solution
There are many such sets $\mathcal{G}$, here are just two simple examples.

$$\mathcal{G}_1 = \{(Sa \cup S\neg a), \quad (a \Rightarrow \neg b), \quad S\neg(a \Rightarrow b), \quad (b \Rightarrow Sa)\}$$

$$\mathcal{G}_2 = \{Sa, \quad (a \Rightarrow b), \quad (\neg b \cup, \quad S\neg a\}.$$

Let $v : VAR \longrightarrow \{F, m, T\}$, be such that $v(a) = T, v(b) = F$. We evaluate $(ST \cup S\neg T) = T \cup T = T$, $(T \Rightarrow \neg F) = T$, $S\neg(T \Rightarrow F) = S\neg F = T$, $(F \Rightarrow ST) = F \Rightarrow T = T$. This proves that v is a **B** model for $\mathcal{G}_1$, i.e. $\mathcal{G}_1$ is consistent.

Assume now that there is $v : VAR \longrightarrow \{F, m, T\}$, such that $v \models_{\mathbf{B}} \mathcal{G}_2$. In particular $v^*(Sa) = T$. This is possible if and only if $v(a) = T$, then $v^*(S\neg a) = SF = F$. This contradicts $v \models_{\mathbf{B}} \mathcal{G}_2$. Hence $\mathcal{G}_2$ is **B** inconsistent.

We introduce, as we did in classical case a notion of a contradiction as follows.

Definition 3.42

Let **M** *be an extensional semantics for* $\mathcal{L}_{CON}$. *We say that a formula* A *is a* **M** *contradiction if it doesn't have a* **M** *model.*

Example 3.25

A formula $(Sa \cap S\neg a)$ *of* $\mathcal{L}_{\mathbf{B}} = \mathcal{L}_{\{\neg, S, \Rightarrow, \cup, \cap\}}$ *is a* **B** *contradiction.*

Proof
Assume that there is v, such that $v \models (Sa \cap S\neg a)$, i.e. $v^*((Sa \cap S\neg a)) = T$ if and only if (shorthand notation) $Sa = T$ and $S\neg a = T$. But $Sa = T$ if and only of $a = T$. In this case $S\neg T = SF = F \neq T$. This contradiction proves that such v does not exist, i.e. that for all v, $v \not\models (Sa \cap S\neg a)$.

This also justifies the following.

Example 3.26 *The set* $\mathcal{G} = \{(Sa \cap S\neg a) : \quad a \in VAR\}$ *is an countably infinite* **B** *inconsistent set.*

Here is a simple problem asking to create your own, specific **M** semantics fulfilling certain specifications. This semantics is different from all of previous semantics defined and examined. We also ask to examine some of its properties, including **M** consistency and **M** inconsistency. We provide an example two different semantics. We encourage the reader to come up with his/hers own and to write down formally its full definition according to definition 3.7 as it was done in the case of **L** semantics.

Review Problem

Part 1. Write the following natural language statement:

One likes to play bridge, or from the fact that the weather is good we conclude the following: one does not like to play bridge or one likes not to play bridge

as a formula of 2 different languages

1. Formula $A_1 \in \mathcal{F}_1$ of a language $\mathcal{L}_{\{\neg,\ L,\ \cup,\ \Rightarrow\}}$, where LA represents statement "one likes A", "A is liked".

2. Formula $A_2 \in \mathcal{F}_2$ of a language $\mathcal{L}_{\{\neg,\ \cup,\ \Rightarrow\}}$.

Part 2. Define formally, following all steps of the defnition 3.7, a 3 valued extensional semantics **LK** for the language $\mathcal{L}_{\{\neg,\ \mathbf{L},\ \cup,\ \Rightarrow\}}$ under the following assumptions.

s1 We assume that the third value is denoted by $\perp$ is *intermediate* between designated value T and F, i.e. that $F < \perp < T$.

s2 We model a situation in which one "likes" only truth, represented by T; i.e. in which

$$LT = T, \quad L\perp = F, \quad LF = F.$$

s3 The connectives $\neg$, $\cup$, $\Rightarrow$ can be defined as one wishes, but they have to be defined in such a way to make sure that always "one likes A or does not like A", i.e. it must be assured that $\models_{\mathbf{LK}} \ (LA \cup \neg LA)$.

Part 3.

1. Verify whether the formulas A_1 and A_2 from the **Part 1.** have a model/ counter model under your semantics **LK**. You can use shorthand notation.

2. Verify whether the following set G is **LK** consistent. You can use **shorthand notation**.

$$\mathcal{G} = \{La, \quad (a \cup \neg Lb), \quad (a \Rightarrow b), \ b \ \}.$$

3. Give an example on an infinite, **LK** consistent set of formulas of the language $\mathcal{L}_{\{\neg,\ L,\ \cap,\ \cup,\ \Rightarrow\}}$. Some formulas must contain the connective L.

Review Problem Solutions

Part 1 Solution

1. We translate the statement into a formula $A_1 \in \mathcal{F}_1$ of a language $\mathcal{L}_{\{\neg,\ \mathbf{L},\ \cap,\ \cup,\ \Rightarrow\}}$ as follows.

Propositional variables: a, b, where a denotes statement: *play bridge*, b denotes a statement: *the weather is good.*

$$A_1 = (\mathbf{L}a \cup (b \Rightarrow (\neg \mathbf{I}a \cup \mathbf{L}\neg a))).$$

2. We translate our statement into a formula $A_2 \in \mathcal{F}_2$ of a language $\mathcal{L}_{\{\neg,\ \cup,\ \Rightarrow\}}$ as follows.

Propositional Variables: a, b, c, where a denotes statement: *One likes to play bridge,* b denotes a statement: *the weather is good*, and c denotes a statement: *one likes not to play bridge.*

$$A_2 = (a \cup (b \Rightarrow (\neg a \cup c))).$$

Part 2 Solution 1

Here is a simple **LK** semantics. We define the logical connectives by writing functions defining connectives in form of the truth tables and skipping other points of the definition 3.7. We leave it to the reader as an exercise to write down a full definition according to the definition 3.7.

LK Semantics 1

L	F	$\perp$	T
	F	*F*	T

$\neg$	F	$\perp$	T
	T	*F*	F

$\cap$	F	$\perp$	T
F	F	F	F
$\perp$	F	$\perp$	$\perp$
T	F	$\perp$	T

$\cup$	F	$\perp$	T
F	F	$\perp$	T
$\perp$	$\perp$	T	T
T	T	*T*	T

$\Rightarrow$	F	$\perp$	T
F	T	T	T
$\perp$	*T*	$\perp$	T
T	F	*F*	T

We verify whether the condition **s3** is satisfied, i.e. whether $\models_{\mathbf{LK}} (LA \cup \neg LA)$ by simple evaluation. Let $v : VAR \longrightarrow \{F, \perp, T\}$ be any truth assignment. For any formula A, $v^*(A) \in \{F, \perp, T\}$ and $LF \cup \neg LF = LF \cup \neg LF = F \cup \neg F \cup T = T$, $L \perp \cup \neg L \perp = F \cup \neg F = F \cup T = T$, $LT \cup \neg LT = T \cup \neg T = F \cup T = T$.

Part 2 Solution 2

Here is another simple **LK** semantics. Writing, yet again, a full definition is left to the reader as an exercise.

LK Semantics 2

The logical connectives are the following funcions in the set $\{F, \bot, T\}$, where $\{F < \bot < T\}$. We define $\neg F = T,\ \neg \bot = T,\ \neg T = F$ and, as by **s2**, $LT = T,\ L\bot = F,\ LF = F$. We define, for any $x, y \in \{F, \bot, T\}$

$$x \cap y = min\{x,\ y\},\ x \cup y = T,\ x \Rightarrow y = T \text{ if } x \leq y,\ x \Rightarrow y = F \text{ if } x > y.$$

From the above definition we can see the **LK** satisfies the requirement **s3** that especially $\models_{\mathbf{LK}} (LA \cup \neg LA)$ since for any truth assignment v, no matter what values $v^*(LA)$ and $v^*(\neg LA)$ are, the combination of them by $\cup$ will always be T.

Part 3

1. Verify whether the formulas A_1 and A_2 from the **Part 1.** have a model/ counter model under your semantics **LK**. You can use shorthand notation.

Solution 1
A model for $A_1 = (La \cup (b \Rightarrow (\neg La \cup L\neg a)))$ under **LK** semantics 1 is any v, such that $v(a) = T$. By easy evaluation, A_1 does not have no counter model, i.e. $\models_{\mathbf{LK}} A_1$. Also any v, such that $v(a) = T$ is a model for A_1 as we have $v^*(A_2) = T \cup v^*((b \Rightarrow (\neg a \cup c))) = T$ by definition of $\cup$.

Solution 2 The main connective of A_1 and A_2 is $\cup$. By definition of $\cup$ in **LK** semantics 2, $x \cup y = T$ for all $x, y \in \{F, \bot, T\}$, and hence any v is a model for both A_1 and A_2, i.e. they are both tautologies under **LK** semantics 2.

Part 3

2. Verify whether the following set G is **LK** consistent. You can use **shorthand notation**.

$$\mathcal{G} = \{La,\ \ (a \cup \neg Lb),\ \ (a \Rightarrow b),\ \ b\ \}.$$

Solution 1
$\mathcal{G}$ is **LK** consistent under semantics 1 because any v, such that $v(a) = T,\ v(b) = T$ is a **LK** model for $\mathcal{G}$ under semantics 1 by straightforward evaluation.

Solution 2
Consider any v, such that $v(a) = v(b) = T$. We evaluate: $v^*(La) = LT = T,\ v^*((a \cup \neg Lb)) = T \cup F = T,\ \ v^*(a \Rightarrow b)) = T \Rightarrow T = T$. This proves $v \models_{\mathbf{LK}} \mathcal{G}$, i.e. $\mathcal{G}$ is consistent.

Part 3

3. Give an example on an infinite, **LK** consistent set of formulas of the language $\mathcal{L}_{\{\neg,\ L,\ \cap,\ \cup,\ \Rightarrow\}}$. Some formulas must contain the connective L.

Solution
The infinite set $\mathcal{G} = \{La : \ a \in VAR\}$ is consistent under both **LK** semantics, as any v, such that $v(a) = T$ we get $v^*(La) = LT = T$ by **s2**.

The infinite set $\mathcal{G} = \{(La \cup (b \cap L\neg c)) : \ a, \ b, c \in VAR\}$ is consistent under the semantics 2 by its definition of $\cup$. Any v, such that $v(a) = T$ is its model.

3.6.3 M Equivalence of Languages

Given an extensional semantics **M** defined for a propositional language $\mathcal{L}_{CON}$ with the set $\mathcal{F}$ of formulas and a set $LV \neq \emptyset$ of logical values. We extend now the classical notions of **logical equivalence** and **equivalence of languages** introduced in section 3.4 to the extensional semantics **M**.

Definition 3.43 (M Equivalence)

For any formulas $A, B \in \mathcal{F}$, we say that A, B are **M** *logically equivalent if and only if they always have the same logical value assigned by the semantics* **M**, *i.e. when $v^*(A) = v^*(B)$ for all $v : VAR \ \rightarrow \ LV$. We write*

$$A \equiv_{\mathbf{M}} B$$

to denote that A, B are **M logically equivalent**.

Remember that $\equiv_{\mathbf{M}}$ is not a logical connective. It is just a **metalanguage symbol** for saying "formulas A, B are logically equivalent under the semantics **M**". We use symbol $\equiv$ for classical logical equivalence only.

Exercise 3.31

The classical logical equivalence $(A \cup B) \equiv (\neg A \Rightarrow B)$ holds for all formulas A, B and is defining $\cup$ in terms of negation and implication. Show that it does not hold under **L** *semantics, i.e. that there are formulas A, B, such that*

$$(A \cup B) \not\equiv_{\mathbf{L}} (\neg A \Rightarrow B)$$

Solution
Consider a case when A = a and B = b. By definition 3.43 we have to show $v^*((a \cup b)) \neq v^*((\neg a \Rightarrow b))$ for some $v : VAR \ \rightarrow \ \{F, \perp, T\}$. Observe that $v^*((a \cup b)) = v^*((\neg a \Rightarrow b))$ for all $v : VAR \ \rightarrow \ \{F, T\}$. So we have to check only truth assignments that involve $\perp$. Let v be any v such that $v(a) = v(b) = \perp$. We evaluate $v^*((a \cup b) = \perp \cup \perp = \perp$ and $v^*((\neg a \Rightarrow b)) = \neg \perp \Rightarrow \perp = F \Rightarrow \perp = T$. This proves that $(a \cup b) \not\equiv_{\mathbf{L}} (\neg a \Rightarrow b)$.. and hence we have proved $(A \cup B) \not\equiv_{\mathbf{L}} (\neg A \Rightarrow B)$.

We proved that the classical equivalence defining disjunction in terms of negation and implication can't be used for the same goal in **L** semantics. It does not mean that we can't define **L** disjunction in terms of **L** implication. In fact, we prove by simple evaluation that the following holds.

Fact 3.10

The **L** *disjunction is definable in terms of* **L** *implication only, i.e. for any formulas* $A, B \in \mathcal{F}$

$$(A \cup B) \equiv_{\mathbf{L}} ((A \Rightarrow B) \Rightarrow B).$$

The classical equivalence substitution theorem 3.12 extends to any semantics **M** as follows.

Theorem 3.28 (M Equivalence)

Let a formula B_1 *be obtained from a formula* A_1 *by a substitution of a formula* B *for one or more occurrences of a sub-formula* A *of* A_1*, what we denote as*

$$B_1 = A_1(A/B).$$

Then the following holds for any formulas $A,\ A_1,\ B,\ B_1 \in \mathcal{F}$.

$$\textit{If} \quad A \equiv_{\mathbf{M}} B, \quad \textit{then} \quad A_1 \equiv_{\mathbf{M}} B_1.$$

We leave the proof to the reader as an exercise.

Example 3.27

Let $A_1 = (a \Rightarrow (\neg a \cup b))$ *and consider a sub formula* $A = (\neg a \cup b)$ *of* A_1. *By Fact 3.10,* $(\neg a \cup b) \equiv_{\mathbf{L}} ((\neg a \Rightarrow b) \Rightarrow b)$. *Take* $B = ((\neg a \Rightarrow b) \Rightarrow b)$ *and let* $B_1 = A_1(A/B) = A_1((\neg a \cup b)/((\neg \Rightarrow b) \Rightarrow b)) = (a \Rightarrow ((\neg a \Rightarrow b) \Rightarrow b))$. *By the M Equivalence Theorem 3.28*

$$(a \Rightarrow (\neg a \cup b)) \equiv_{\mathbf{L}} (a \Rightarrow ((\neg \Rightarrow b) \Rightarrow b)).$$

M Equivalence of Languages

We extend now, in a natural way, the classical notion equivalence of languages introduced and examined in section 3.4.

Definition 3.44

Given two languages: $\mathcal{L}_1 = \mathcal{L}_{CON_1}$ *and* $\mathcal{L}_2 = \mathcal{L}_{CON_2}$, *for* $CON_1 \neq CON_2$. *We say that* $\mathcal{L}_1$ *and* $\mathcal{L}_2$ *are* **M** *logically equivalent and denote it by*

$$\mathcal{L}_1 \equiv_{\mathbf{M}} \mathcal{L}_2$$

if and only if the following conditions **C1, C2** *hold.*

C1 *For any formula A of $\mathcal{L}_1$, there is a formula B of $\mathcal{L}_2$, such that $A \equiv_{\mathbf{M}} B$,*

C2 *For any formula C of $\mathcal{L}_2$, there is a formula D of $\mathcal{L}_1$, such that $C \equiv_{\mathbf{M}} D$.*

Exercise 3.32 *Prove that*

$$\mathcal{L}_{\{\neg,\Rightarrow\}} \equiv_{\mathbf{L}} \mathcal{L}_{\{\neg,\Rightarrow,\cup\}}$$

Solution
Condition **C1** holds because any formula of $\mathcal{L}_{\{\neg,\Rightarrow\}}$ is a formula of $\mathcal{L}_{\{\neg,\Rightarrow,\cup\}}$. Condition **C2** holds because the Fact 3.10 equivalence $(A \cup B) \equiv_{\mathbf{L}} ((A \Rightarrow B) \Rightarrow B)$ and the Theorem 3.28.

3.7 Homework Problems

Formal Propositional Languages

For the problems below do the following.

(i) Determine which of the formulas is, and which is not a well formed formula. Determine a formal language of $\mathcal{L}$ to which the formula or set of formulas belong.

(ii) If a formula is correct, write what its *main connective* is. If it is not correct, write the corrected formula and then write its *main connective.* If there is more then one way to correct the formula, write all possible corrected formulas.

(iii) If a formula is correct, write what it says. If it is not correct, write the corrected formula and then write what it says.

(iv) For each of correct formula determine its degree and write down its all sub-formulas of the degree 0 and 1.

Problems

1. $((a \uparrow b) \uparrow (a \uparrow b) \uparrow a)$
2. $(a \Rightarrow \neg b) \Rightarrow \neg a$
3. $\Diamond(a \Rightarrow \neg b) \cup a, \quad \Diamond(a \Rightarrow (\neg b \cup a), \quad \Diamond a \Rightarrow \neg b \cup a$
4. $(\Box\neg\Diamond a \Rightarrow \neg a), \quad \Box(\neg\Diamond a \Rightarrow \neg a), \quad \Box\neg\Diamond(a \Rightarrow \neg a)$
5. $((a \cup \neg K\neg a)), \quad KK(b \Rightarrow \neg a), \quad \neg K(a \cup \neg a)$

6. $(B(a \cap b) \Rightarrow Ka)$, $\quad B((a \cap b) \Rightarrow Ka)$

7. $G(a \Rightarrow b) \Rightarrow Ga \Rightarrow Gb)$, $\quad a \Rightarrow HFa$, $\quad FFa \Rightarrow Fa$

8. $(a \Rightarrow ((\neg b \Rightarrow (\neg a \cup c)) \Rightarrow \neg a))$

9. $\Diamond((a \cap \neg a) \Rightarrow (a \cap b))$

10. $\Box\neg\Diamond(a \Rightarrow \neg a)$

11. $\Diamond(\Diamond a \Rightarrow (\neg b \cup \Diamond a))$

12. $(\neg(a \cap b) \cup a)$

13. Write the natural language statement:

    From the fact that it is not necessary that an elephant is not a bird we deduce that:
    it is not possible that an elephant is a bird or, if it is possible that an elephant is a bird, then it is not necessary that a bird flies.

    in the following two ways.

    1. As a formula $A_1 \in \mathcal{F}_1$ of a language $\mathcal{L}_{\{\neg,\mathbf{C},\mathbf{I},\cap,\cup,\Rightarrow\}}$.

    2. As a formula $A_2 \in \mathcal{F}_2$ of a language $\mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow\}}$.

14. Write the natural language statement

    If it is not believed that quiz is easy or quiz is not easy, then from the fact that $2 + 2 = 5$ *we deduce that it is believed that quiz is easy.*

    in the following two ways.

    1. As a formula A_1 of a language $\mathcal{L}_1 = \mathcal{L}_{\{\neg,\mathbf{B},\cap,\cup,\Rightarrow\}}$, where **B** is a believe connective. Statement **B**A says: *It is believed that A.*

    2. As a formula A_2 of a language $\mathcal{L}_2 = \mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow\}}$.

Formal Classical Semantics

1. Find and prove definability formula defining implication in terms of conjunction and negation.

2. Find and prove definability formula defining conjunction in terms of disjunction and negation.

3. Find and prove definability formula defining conjunction in terms of implication and negation.

4. Prove that $\cup$ can be defined in terms of $\Rightarrow$ alone.

5. Find and prove definability formula defining $\Rightarrow$ in terms of $\uparrow$.

6. Find definability formula defining $\Rightarrow$ in terms of $\downarrow$.

7. Define $\cap$ in terms of $\Rightarrow$ and $\neg$.

8. Find definability formula defining $\cap$ in terms of $\downarrow$ alone.

9. Given a formula A: $(((a \cap b) \cup \neg c) \Rightarrow b)$. Evaluate (do not use shorthand notation) $v^*(A)$ for truth assignments $v : VAR \longrightarrow \{T,\ F\}$ such that

   (i)$v(a) = T,\ v(b) = F,\ v(c) = F,\ v(x) = T$ for all $x \in VAR - \{a, b, c\}$,

   (ii)$v(a) = F,\ v(b) = T,\ v(c) = T,\ v(x) = F$ for all $x \in VAR - \{a, b, c\}$.

10. Given a formula A: $(((a \Rightarrow \neg b) \cup b) \Rightarrow a)$. Evaluate (use shorthand notation) $v^*(A)$ for all truth assignments restricted to A.

11. Given a formula A: $(((a \downarrow \neg b) \cup b) \uparrow a)$. Evaluate (do not use shorthand notation) $v^*(A)$ for truth assignments $v : VAR \longrightarrow \{T,\ F\}$ such that

    (i) v(a)=T, v(b)=F, v(c) =F for all $c \in VAR - \{a, b\}$,

    (ii) v(a)=F, v(b)=T, v(c) =T for all $c \in VAR - \{a, b\}$.

    (iii) List all restricted models and counter-models for A.

    Write the following natural language statement *From the fact that it is possible that* $2+2 \neq 4$ *we deduce that it is not possible that* $2+2 \neq 4$ *or, if it is possible that* $2+2 \neq 4$*, then it is not necessary that you go to school.* as a formula . $A \in \mathcal{F}$ of a language $\mathcal{L} = \mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow\}}$.

    (i) Find a restricted model v for the formula A.

    (ii) Find 3 models w of A such that $v^*(A) = w^*(A)$ the for v from (i). How many of such models exist?

    (iii) Find all models, counter-models (restricted) for A. Use shorthand notation.

    (iv) Is $A \in \mathbf{C}$?, is $A_2 \in \mathbf{T}$? Justify your answers.

12. Given $v : VAR \longrightarrow \{T, F\}$ such that $v^*((\neg a \cup b) \Rightarrow (a \Rightarrow \neg c)) = F$. Evaluate: $v^*(((b \Rightarrow a) \Rightarrow (a \Rightarrow \neg c)) \cup (a \Rightarrow b))$.

13. Show that all of the truth assignments v_1, v_2, v_3 defined below are **models** for the formula $A : ((a \cap \neg b) \cup \neg c)$.

    $v_1 : VAR \longrightarrow \{T, F\}$, is such that $v_1(a) = T,\ v_1(b) = F,\ v_1(c) = T,$ and $v_1(x) = F,$ for all $x \in VAR - \{a, b, c\}$;
    $v_2 : VAR \longrightarrow \{T, F\}$ is such that $v_2(a) = T,\ v_2(b) = F,\ v_2(c) = T,$ $v_2(d) = T,$ and $v_2(x) = F$ for all $x \in VAR - \{a, b, c, d\}$;
    $v_3 : VAR \longrightarrow \{T, F\}$ is such that $v_3(a) = T,\ v_3(b) = F,\ v_3(c) = T,$ $v_3(d) = T,\ v_3(e) = T,$ and $v_3(x) = F$, for all $x \in VAR - \{a, b, c, d, e\}$.

14. Prove that for any formula $A \in \mathcal{F}$, if A has a model (counter- model), then it has uncountably many models (counter-models). More precisely, as many as there are real numbers. *Hint* Use the Counting Functions Theorem 3.4.

15. Use Generalization Method to determine whether
$\models (\neg((a \cup b) \Rightarrow ((c \Rightarrow d) \cup e)) \Rightarrow ((a \cup b) \cap (\neg(c \Rightarrow d) \cap \neg e)))$.

16. Prove $\models (\neg((a \cup b) \Rightarrow (c \Rightarrow d)) \Rightarrow (\neg((a \cup b) \Rightarrow (c \Rightarrow d)) \Rightarrow (\neg e \cap a)))$.

17. Use Proof by Contradiction Method to determine whether
$\models (((A \Rightarrow (B \Rightarrow C)) \cap (A \Rightarrow B)) \Rightarrow (A \Rightarrow C))$.

18. Use Truth Table and Substitution Methods to prove $\models (\neg\neg A \Leftrightarrow A)$.

19. Use Truth Table and Substitution Methods to prove to prove the *Reductio ad Absurdum* tautology $((\neg A \Rightarrow (B \cap \neg B)) \Rightarrow A)$.

20. Use Proof by Contradiction Method to prove the *Exportation and Importation* tautology $(((A \cap B) \Rightarrow C) \Leftrightarrow (A \Rightarrow (B \Rightarrow C)))$.

21. For the formulas listed below determine whether they are tautologies or not. If a formula is not a tautology list its counter-model (restricted). Use shorthand notation.

    (i) $A_1 = (\neg(a \Rightarrow (b \cap \neg c)) \Rightarrow (a \cap \neg(b \cap \neg c)))$

    (ii) $A_2 = ((a \cap \neg b) \Rightarrow ((c \cap \neg d) \Rightarrow (a \cap \neg b)))$

    (iii) $A_3 = (\neg(A \cap \neg B) \cup (A \cap \neg B))$

22. Find all models and a counter-model restricted to $\mathcal{G}$ (if exist) for the following sets $\mathcal{G}$ of formulas. Use shorthand notation.

    (i) $\mathcal{S}_1 = \{a, (a \cap \neg b), (\neg a \Rightarrow (a \cup b))\}$

    (ii) $\mathcal{S}_2 = \{(a \Rightarrow b), (c \cap \neg a), b\}$

    (iii) $\mathcal{S}_3 = \{a, (a \cap \neg b), \neg a, c\}$

23. Give an example of an infinite set $\mathcal{G} \subseteq \mathcal{F}$, such that $\mathcal{G} \neq \mathbf{T}$ and $\mathcal{G}$ has a model, i.e. is consistent.

24. Give an example of an infinite consistent set $\mathcal{G} \subseteq \mathcal{F}$, such that $\mathcal{G} \cap \mathbf{T} = \emptyset$.

25. Give an example of an infinite set $\mathcal{G} \subseteq \mathcal{F}$, such that $\mathcal{G} \neq \mathbf{C}$ and $\mathcal{G}$ does not have a model, i.e.is inconsistent.

26. Give an example of an infinite set $\mathcal{G} \subseteq \mathcal{F}$, such that $\mathcal{G} \cap \mathbf{C} = \emptyset$.

27. Find an infinite number of formulas that are independent from a set $\mathcal{G} = \{(a \Rightarrow (a \cup b)),\ (a \cup b),\ \neg b,\ (c \Rightarrow b)\}$. Use shorthand notation.

28. Given an infinite set $\mathcal{G} = \{(a \cup \neg a) :\ a \in VAR\}$. Find 3 formulas $A \in \mathcal{F}$ that are independent from $\mathcal{G}$.

29. Give an example of an infinite set $\mathcal{G}$ and an infinite set of formulas independent from it.

Equivalence of Languages

1. Prove that $\mathcal{L}_{\{\cap,\neg\}} \equiv \mathcal{L}_{\{\cup,\neg\}}$.
2. Transform a formula $A = \neg(\neg(\neg a \cap \neg b) \cap a)$ of $\mathcal{L}_{\{\cap,\neg\}}$ into a logically equivalent formula B of $\mathcal{L}_{\{\cup,\neg\}}$.
3. Transform a formula $A = (((\neg a \cup \neg b) \cup a) \cup (a \cup \neg c))$ of $\mathcal{L}_{\{\cup,\neg\}}$ into a formula B of $\mathcal{L}_{\{\cap,\neg\}}$, such that $A \equiv B$.
4. Prove, using proper logical equivalences (list them at each step) that

   (i) $\neg(A \Leftrightarrow B) \equiv ((A \cap \neg B) \cup (\neg A \cap B))$.

   (ii) $((B \cap \neg C) \Rightarrow (\neg A \cup B)) \equiv ((B \Rightarrow C) \cup (A \Rightarrow B))$.
5. Prove that $\mathcal{L}_{\{\neg,\cap\}} \equiv \mathcal{L}_{\{\neg,\Rightarrow\}}$.
6. Prove by using proper logical equivalences that

   (i) $\neg(\neg A \cup \neg(B \Rightarrow \neg C)) \equiv (A \cap \neg(B \cap C))$,

   (ii) $(\neg A \cap (\neg A \cup B)) \equiv (\neg A \cup (\neg A \cap B))$.
7. Prove that $\mathcal{L}_{\{\cap,\cup.\neg\}} \equiv \mathcal{L}_{\{\Rightarrow,\neg\}}$.
8. Prove that $\mathcal{L}_{\{\cap,\cup,\Rightarrow,\neg\}} \equiv \mathcal{L}_{\{\cup,\neg\}}$.
9. (i) Transform a formula $A = (((a \cup \neg b) \Rightarrow a) \cap (\neg a \Rightarrow \neg b))$ of $\mathcal{L}_{\{\cap,\cup,\Rightarrow,\neg\}}$ into a logically equivalent formula B of $\mathcal{L}_{\{\cup,\neg\}}$.

   (ii) Find all B of $\mathcal{L}_{\{\cup,\neg\}}$, such that $B \equiv A$, for A from (i).
10. (i) Transform a formula $A = (((\neg a \cup \neg b) \cup a) \cup (a \cup \neg c))$ of $\mathcal{L}_{\{\cup,\neg\}}$ into a formula B of $\mathcal{L}_{\{\cap,\cup,\Rightarrow,\neg\}}$, such that $A \equiv B$.

    (ii) Find all B of $\mathcal{L}_{\{\cap,\cup,\Rightarrow,\neg\}}$, such that $B \equiv A$, for A from (i)
11. Prove that $\mathcal{L}_{\{\cap,\cup,\Rightarrow,\neg\}} \equiv \mathcal{L}_{\{\uparrow\}}$.
12. Prove that $\mathcal{L}_{\{\cap,\cup,\Rightarrow,\neg\}} \equiv \mathcal{L}_{\{\ \downarrow\}}$.
13. Prove that $\mathcal{L}_{\{\uparrow\}} = \mathcal{L}_{\{\ \downarrow\}}$.

Many Valued Semantics

1. In all 3-valued semantics presented here we chose the language without the equivalence connective "$\Leftrightarrow$". Extend t **L**, $\mathbf{L}_4$ semantics to a language containing the equivalence connective. Prove that your semantics is well defined as by definition 3.14.
2. Extend **H**, **K**, semantics to a language containing the equivalence connective. Are your semantics well defined as by definition 3.14?

3. Extend **B**, semantics to a language containing the equivalence connective. Are your semantics well defined as by definition 3.14?

4. Let $v : VAR \longrightarrow \{F, \bot, T\}$ be any v, such that $v^*((a \cup b) \Rightarrow (a \Rightarrow c)) = \bot$ under H semantics. Evaluate $v^*(((b \Rightarrow a) \Rightarrow (a \Rightarrow \neg c)) \cup (a \Rightarrow b))$.

5. Verify which of the classical tautologies (3.12) are, and which are not **L** tautologies.

6. Verify which of the classical tautologies (3.13) are, and which are not **L** tautologies.

7. Give an example of 3 formulas

8. For each of 3-valued logic semantics presented in this chapter, find 5 classical tautologies that are tautologies of that logic.

9. Examine the notion of *definability of connectives* as defined in section 3.3, definition 3.16 for **L** semantics. semantics.

10. Examine the notion of *definability of connectives* as defined in section 3.3, definition 3.16 for **H** semantics. semantics.

11. Given a set $\mathcal{G} = \{((a \cap b) \Rightarrow b),\ (a \cup b), a\}$. Verify whether $\mathcal{G}$ is consistent under **H** semantics.

12. Given a set $\mathcal{G} = \{((a \cap b) \Rightarrow b),\ (a \cup b), a\}$. Verify whether $\mathcal{G}$ is consistent under **L** semantics.

13. Given a language $\mathcal{L}_{\{\neg, \Rightarrow, \cup, \cap\}}$. We define: A formula $A \in \mathcal{F}$ is called **M independent** from a set $\mathcal{G} \subseteq \mathcal{F}$ if and only if the sets $\mathcal{G} \cup \{A\}$ and $\mathcal{G} \cup \{\neg A\}$ are both **M** consistent. I.e. when there are truth assignments $v_1,\ v_2$ such that $v_1 \models_{\mathbf{M}} \mathcal{G} \cup \{A\}$ and $v_2 \models_{\mathbf{M}} \mathcal{G} \cup \{\neg A\}$.

    Given a set $\mathcal{G} = \{((a \cap b) \Rightarrow b),\ (a \cup b), a\}$.

    (i) Find a formula A that is **L independent** from a set $\mathcal{G}$.

    (ii) Find a formula A that is **H independent** from a set $\mathcal{G}$.

    (iii) Find an infinite number of that are **L independent** from a set $\mathcal{G}$.

    (iv) Find an infinite number of that are **H independent** from a set $\mathcal{G}$.

14. By exercise 3.31 the classical logical equivalence $(A \cup B) \equiv (\neg A \Rightarrow B)$ does not hold under **L** semantics, i.e. that there are formulas A, B, such that $(A \cup B) \not\equiv_{\mathbf{L}} (\neg A \Rightarrow B)$. Show 3 formulas A,B such that it does hold for **L** semantics, i.e. such that are formulas A, B, such that $(A \cup B) \equiv_{\mathbf{L}} (\neg A \Rightarrow B)$.

Chapter 4

General Proof Systems: Syntax and Semantics

Proof systems are built to prove, construct formal proofs of statements formulated in a given language formulated in a given language. First component of any proof system is hence its formal language $\mathcal{L}$. Proof systems can be thought as an *inference machine* with special statements, called *provable statements*, or *theorems* being its final products. The starting points are called *axioms* of the proof system. We distinguish two kinds of axioms: *logic* LA and *specific* SA.

When building a proof system for a given language and its semantics i.e. for a logic defined semantically we choose as a set of logical axioms LA some subset of tautologies, i.e. statements always true. This is why we call them logical axioms. A proof system with only logic axioms LA is also called *logic* proof systems, or just *proof systems* for short. If we build a proof system for which there is no known semantics, as it has happened in the case of classical, intuitionistic, and modal logics, we think about the logical axioms as statements universally true. We choose as axioms (finite set) the statements we for sure want to be universally true, and whatever semantics follows they must be tautologies with respect to it. Logical axioms are hence not only tautologies under an established semantics, but they also guide us how to establish a semantics, when it is yet unknown.

For the set of specific axioms SA we choose these formulas of the language that describe our knowledge of a universe we want to prove facts about. They are not universally true, they are true only in the universe we are interested to describe and investigate. This is why we call them specific axioms. A proof system with logical axioms LA and specific axioms SA is called *a formal theory* based on a proof system with logic axioms LA.

The inference machine is defined by a finite set of rules, called *inference rules.*

The inference rules describe the way we are allowed to transform the information within the system with axioms as a staring point. The process of this transformation is called *a formal proof.* The provable formulas for which we have a formal proof are called consequences of the axioms, or theorem, or just simple *provable formulas.* We use proof systems not only to be able to build formal proofs in them, but also to search for proofs of given statements of their the language. We distinguish special proof systems for which it is possible to define a mechanical method for determining, given any statement of A, but which also generates a proof, is called *syntactically decidable* or *automatically decidable*, or *an automated system*

When building a proof system we choose not only axioms of the system, but also specific rules of inference. The choice of rules is often linked, as was the choice of axioms, with a given semantics. We want the rules to preserve the truthfulness of what we are proving from axioms via the rules. Rules with this property are called *sound rules* and the system *a sound proof system.* The notion of truthfulness is always defined by a given propositional, or predicate language $\mathcal{L}$ semantics $\mathbf{M}$. Rules of inference can be sound under one semantics and not sound under another. When developing a proof system S the first goal is prove a theorem, called *Soundness Theorem* about its relationship with its semantics $\mathbf{M}$. It states that the following holds for any formula A of the language $\mathcal{L}$ of the system S. *If a formula A is provable from logical axioms LA of S only, then A is a tautology under the semantics* $\mathbf{M}$.

A proof system can be sound under one semantics, and not sound under the other. For example a set of axioms and rules sound under classical logic semantics might not be sound under intuitionistic semantics, $\mathbf{H}$, $\mathbf{L}$, $\mathbf{K}$ semantics, or others. This is why we talk about proof systems for classical logic, intuitionistic logic, for modal logics etc. In general there are many proof systems that are sound under a given semantics, i.e. there are many sound proof systems for a given logic semantically defined. We present some examples at the end of the chapter. Given a proof system S with logical axioms LA that is sound under a given semantics M. Let $\mathbf{T}_M$ be a set of all tautologies defined by the semantics M, i.e. $\mathbf{T}_M = \{A : \ \models_M A\}$. A natural questions arises: are all tautologies defined by the semantics M, provable in the system S that is sound under the semantics M. The positive answer to this question is called a *completeness* property of the system S. Because we ask the completeness property question for sound systems only we put it in a form of a theorem called a *Completeness Theorem* for a proof system S, under a semantics M. It states that the following holds for any formula A of the language $\mathcal{L}$ of the system S. *A formula A is provable in S if and only if A is a tautology under the semantics M.* We write it symbolically as: $\vdash_S A$ if and only if $\models_M A$. The Completeness Theorem is composed from two parts: the Soundness Theorem and the *completeness part* that proves the completeness property of a sound system.

Proving the Soundness Theorem for S under a semantics M is usually a straightforward and not a very difficult task. We first prove that all logical axioms are

M tautologies, and then that all inference rules of the system preserve the notion of the **M** truth (**M** model). Proving the *completeness part* of the Completeness Theorem is always a crucial and very difficult task.

We will study two proofs of the Completeness Theorem for classical propositional Hilbert style proof system in chapter 5, and a constructive proofs for automated theorem proving systems for classical logic the chapter 6.

Observe that we formulated all these basic theorems linking semantics and syntax (provability) in a general manner. As we first consider propositional languages (chapters 5, 6, 7) and hence we use proof systems for propositional logics as examples. The case of predicate logics will be discussed in chapters 8, 9, 10, 11.

4.1 Syntax

In this section we formulate a definition of a proof system S by specifying and defining and all its components. We define a notion of a formal proof in a given proof system, and give simple examples of different proof systems. When defining a proof system S we specify, as the first step, its formal language $\mathcal{L}$. When It can be a propositional, or a predicate language. It is a first component of the proof system S. Given a set $\mathcal{F}$ of well formed formulas, of the language $\mathcal{L}$, we often extend this set, and hence the language $\mathcal{L}$ to a set $\mathcal{E}$ of *expressions* build out of the language $\mathcal{L}$, and some additional symbols, if needed. It is a second component of the proof system S. Proof systems act as an inference machine, with provable expressions being its final products. This inference machine is defined by setting, as a starting point a certain non-empty, proper subset LA of $\mathcal{E}$, called a set of *logical axioms* of the system S. The production of provable formulas is to be done by the means of *inference rules.* The inference rules transform an expression, or finite string of expressions, called premisses, into another expression, called conclusion. At this stage the rules don't carry any meaning - they define only how to transform strings of symbols of our language into another string of symbols. This is a reason why investigation of proof systems is called *syntax* or *syntactic investigation* as opposed to *semantcal methods*, which deal with semantics of the language and hence of the proof system. The *syntax- semantics* connection within proof systems is established by Soundness and Completeness theorems and will be discussed in detail in the section 4.2.

Definition 4.1 (Proof System)

By a proof system we understand a triple

$$S = (\mathcal{L}, \mathcal{E}, LA, \mathcal{R}),$$

where $\mathcal{L} = (\mathcal{A}, \mathcal{F})$ *is a formal language, called the language of* S *with a set* $\mathcal{F}$ *of* *formulas; $\mathcal{E}$ is a set of expressions of S; $LA \subseteq \mathcal{E}$ is a non empty set of logical axioms of the system; $\mathcal{R}$ is a finite set of rules of inference.*

The components of the proof systems S are defined as follows.

1. The language $\mathcal{L}$ of S

In the propositional case, the formal language $\mathcal{L}$ consists of two components: an alphabet $\mathcal{A}$ and a set $\mathcal{F}$ of formulas. In predicate case the language $\mathcal{L}$ consists of thee components: an alphabet $\mathcal{A}$, a set $\mathbf{T}$ of terms and a set $\mathcal{F}$ of formulas. The set $\mathbf{T}$ of terms is needed to define properly the set of $\mathcal{F}$ of formulas and we list it as to distinguish it the propositional case. We will denote the language $\mathcal{F}$ of S uniformly as $\mathcal{L} = (\mathcal{A}, \mathcal{F})$ and specify if it is propositional or a predicate language accordingly. We assume that the both sets $\mathcal{A}$ and $\mathcal{F}$ are enumerable, i.e. we will deal here with enumerable languages only.

Semantical Link. Any semantics $\mathbf{M}$ for the language $\mathcal{L}$ is called the semantics for the proof system S.

2. The set $\mathcal{E}$ of expressions of S

Given a set $\mathcal{F}$ of well formed formulas, of the language $\mathcal{L}$, we often extend this set (and hence the language $\mathcal{L}$ to some set $\mathcal{E}$ of *expressions* build out of the language $\mathcal{L}$, and some additional symbols, if needed.

Automated theorem proving systems use as their basic components expressions build out of formulas of the language $\mathcal{L}$. They are for example sequences of formulas in the proof systems **RS** and **RQ** presented in chapter 5.72 and —in chapter **??**, respectively. The first of such systems Gentzen's systems **LK** for classical logic and **LK** for intuitionistic logic and their followers use expressions called Gentzen sequents. They are presented and discussed in chapter **??**. There also are resolution based proof systems that use different form of expressions to represent for clauses and sets of clauses to mention the few. In many proof system we choose the set of formulas $\mathcal{F}$ as expressions, i.e. we put $\mathcal{E} = \mathcal{F}$.

Semantical Link. We always have to extend a given semantics $\mathbf{M}$ of the language $\mathcal{L}$ of the system S to the set $\mathcal{E}$ of expression.

3. The set LA of logical axioms of S

The logical axioms LA of S form a non-empty subset of the set $\mathcal{E}$ of expressions. In particular, LA is a non-empty subset of formulas, i.e. $LA \subseteq \mathcal{F}$. We assume here that the set LA of logical axioms is finite, i.e. we consider here only *finitely axiomatizable* proof systems.

Semantical Link. Set LA of logical axioms is always a subset of expressions that are *tautologies* under the semantics $\mathbf{M}$ of the language $\mathcal{L}$ of S.

4. The set $\mathcal{R}$ of rules of inference of S

We assume that the proof system S contains a finite number of inference rules. We assume that each rule has a finite number of premisses and one conclusion. We also assume that one can effectively decide, for any inference rule, whether a given string of expressions form its premisses and conclusion or do not, i.e. that all rules $r \in \mathcal{R}$ are primitivvely recursive.

We put it in a formal definition as follows.

Definition 4.2 (Rule of Inference)

Given a non- empty set $\mathcal{E}$ of expressions of a proof system S. Each rule of inference $r \in \mathcal{R}$ is a relation defined in the set $\mathcal{E}^m$, where $m \geq 1$ with values in $\mathcal{E}$, i.e. $r \subseteq \mathcal{E}^m \times \mathcal{E}$.

Elements $P_1, P_2, \ \dots \ P_m$ of a tuple $(P_1, P_2, \ \dots \ P_m, C) \in r$ are called **premisses** *of the rule r, and C is called its* **conclusion**.

We usually write the inference rules in a following convenient way.

If r is a one premiss rule and $(P_1, C) \in r$, then we write it as

$$(r) \quad \frac{P_1}{C}.$$

If r is a two premisses rule and $(P_1, P_2, C) \in r$, then we write it as

$$(r) \quad \frac{P_1 \ \ ; \ \ P_2}{C},$$

P_1 is called a left premiss of r and P_2 is called a right premiss.

In general, if r is an m- premisses rule and $(P_1, P_2, ...P_m, C) \in r$, then we will write it as

$$(r) \quad \frac{P_1 \ \ ; \ \ P_2 \ \ ; \ \ .... \ \ ; \ \ P_m}{A}.$$

Semantical Link. We want the rules of inference to preserve truthfulness i.e. to be sound under the semantics **M**.

Formal Proofs in S

Given a proof system $S = (\mathcal{L}, \mathcal{E}, LA, \mathcal{R})$. Final products of a single or multiple use of the inference rules of S, with logical axioms LA taken as a starting point are called provable expressions of the system S. A single use of an inference rule is called a *direct consequence.* A multiple application of rules of inference with axioms taken as a starting point is called a *formal proof.* Formal definitions are as follows.

Definition 4.3 (DirectConsequence)

A conclusion of a rule of inference is called a direct consequence of its premisses. I.e. for any rule of inference $r \in \mathcal{R}$, if $(P_1, ...P_n, C) \in r$, then C is called a direct consequence of $P_1, ...P_n$ by virtue of r.

Definition 4.4 (Formal Proof)

Given a proof system $S = (\mathcal{L}, \mathcal{E}, LA, \mathcal{R})$. Any sequence $E_1,\ E_2, ,\ E_n$ of expressions from $\mathcal{E}$, such that $n \geq 1$,

$$E_1 \in LA, \qquad E_n = E,$$

and for each $1 < i \leq n$, either $E_i \in LA$ or E_i is a **direct consequence** *of some of the preceding expressions in $E_1,\ E_2, ,\ E_n$ by virtue of one of the rules of inference $r \in \mathcal{R}$ is called a* **formal proof** *of E in S.*

The number $n \geq 1$ is the length of the proof $E_1,\ E_2, ,\ E_n$. We write

$$\vdash_S E$$

to denote that $E \in \mathcal{E}$ **has a formal proof** *in S. When the proof system S is fixed we write $\vdash E$.*

Any expression E such that E has a proof in S, is called a *provable expression* of S. The set of **all provable expressions** of S is denoted by $\mathbf{P}_S$ and is defined as follows.

$$\mathbf{P}_S = \{E \in \mathcal{E} : \ \vdash_S E\}. \tag{4.1}$$

Consider a simple proof system system S_1 with a language $\mathcal{L} = \mathcal{L}_{\{P,\ \Rightarrow\}}$, where P is one argument connective. We take $\mathcal{E} = \mathcal{F}, LA = \{(A \Rightarrow A)\}$, and the set of rules o inference $\mathcal{R} = \{(r)\ \frac{B}{PB}\}$. We write our proof system as

$$S_1 = (\mathcal{L}_{\{P,\ \Rightarrow\}},\ \mathcal{F},\ \{(A \Rightarrow A)\},\ \ (r)\ \frac{B}{PB}\ ) \tag{4.2}$$

where A, B are any formulas. Observe that even the system S_1 has only one axiom, it represents an infinite number of formulas. We call such axiom an *axiom schema.*

Consider now a system S_2

$$S_2 = (\mathcal{L}_{\{P, \Rightarrow\}},\ \ \mathcal{F},\ \ \ \{(a \Rightarrow a)\},\ \ \ (r)\ \ \frac{B}{PB}\ ), \tag{4.3}$$

where $a \in VAR$ is any variable (atomic formula) and $B \in \mathcal{F}$ is any formula. Observe that the system S_2 also has only one axiom similar to the axiom of S_1, both systems have the same rule of inference but they are very different proof systems. For example a formula $((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c)))$ is an axiom of the system S_1 for $A = (Pa \Rightarrow (b \Rightarrow c)$ but is not an axiom of the system S_2, as this systems permits axioms of the form: $(a \Rightarrow a)$ for a being a propositional variable.

A formal proof in a system S carries, as the proof system S does, a semantical meaning but it is nevertheless purely *syntactical* in its nature. The rules of inference of a proof system define only how to transform strings of symbols of our language into another string of symbols. The definition of a formal proof says that in order to prove an expression E of a system one has to construct of s sequence of proper transformations as defined by the rules of inference. Here some examples of provable formulas and their formal proofs in both S_1 and S_2 systems. Observe that we do not know the semantics for these systems.

Exercise 4.1 *Let S_1, S_2 be proof systems (7.2), (7.3), respectively. Show that*

$$\vdash_{S_1} ((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c)))$$

$$\vdash_{S_1} P(a \Rightarrow a), \ \ \vdash_{S_2} P(a \Rightarrow a), \ \ \vdash_{S_1} PP(a \Rightarrow a), \ \ \vdash_{S_2} PP(a \Rightarrow a)$$

$$\vdash_{S_1} PPP((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c)).$$

Solution Formal proof of $((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c)))$ in S_1 is one element sequence $A_1 = ((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c))$. It is a proof because the formula A_1 an axiom of S_1. It is not a proof in S_2.

The formulas $P(a \Rightarrow a)$, and $PP(a \Rightarrow a)$ are provable formulas of both proof systems. The formal proofs in both systems of above formulas are identical and are as follows.

Formal proof of $P(a \Rightarrow a)$ in S_1 and S_2 is a sequence $A_1, \ A_2$ for

$$\begin{array}{cc} A_1 = (a \Rightarrow a), & A_2 = P(a \Rightarrow a). \\ \text{axiom} & \text{rule (r) application} \\ & \text{for } B = (a \Rightarrow a) \end{array}$$

Formal proof of $PP(a \Rightarrow a)$ in S_1 and S_2 is a sequence $A_1, \ A_2, \ A_3$ for

$$\begin{array}{ccc} A_1 = (a \Rightarrow a), & A_2 = P(a \Rightarrow a), & A_3 = PP(a \Rightarrow a). \\ \text{axiom} & \text{rule } (r) \text{ application} & \text{rule } (r) \text{ application} \\ & \text{for } B = (a \Rightarrow a) & \text{for } B = P(a \Rightarrow a) \end{array}$$

Formal proof of $PPP((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c)))$ in S_1 is a sequence $A_1, \ A_2, \ A_3, \ A_4$ for

$$\begin{array}{c} A_1 = ((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c))), \\ \text{axiom} \end{array}$$

$A_2 = P((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c))),$
rule (r) application

$A_3 = PP((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c))),$
rule (r) application

$A_4 = PPP((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c))).$
rule (r) application

It is not a proof in S_2. Moreover

$$\nvdash_{S_2} PPP((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c)).$$

Observe that even if the set of axioms and the inference rules of the proof system are primitively recursive it doesn't mean that the notion of "provable expression" is also primitively recursive, i.e. that there always will be an effective, mechanical method (effective procedure) for determining, given any expression A of the system, whether there is a proof of A. We define the following notions

Definition 4.5 (Decidable system)

A proof system $S = (\mathcal{L}, \mathcal{E}, LA, \mathcal{R})$ for which there is an effective decision procedure for determining, for any expression E of the system, whether there is, or there is no proof of E in S is called a **decidable proof system***, otherwise it is called* **undecidable***.*

Observe that the above notion of decidability of the system S does not require us to find a proof, it requires only a mechanical procedure of deciding whether *there is*, or there is no such a proof. We hence introduce a following notion.

Definition 4.6 (Syntactic Decidability)

A proof system $S = (\mathcal{L}, \mathcal{E}, LA, \mathcal{R})$ for which there is an effective mechanical, procedure that finds (generates) a formal proof of any E in S, if it exists, is called **syntactically semi- decidable***. If additionally there is an effective method of deciding that if a proof of E in S not found, it does not exist, the system S is called* **syntactically decidable***. Otherwise S is* **syntactically undecidable***.*

The existence of prove systems for classical logic and mathematics that are syntactically decidable or syntactically semi-decidable was stated (in a different form) by German mathematician David Hilbert in early 1900 as a part of what is called Hilbert's program. The main goal of Hilbert's program was to provide secure foundations for all mathematics. In particular it addressed the problem of decidability; it said that here should be an algorithm for deciding the truth or falsity of any mathematical statement. Moreover, it should use only "finitistic" reasoning methods. Kurt Gdel showed in 1931 that most of the goals of Hilbert's program were impossible to achieve, at least if interpreted in the most obvious way. Nevertheless, Gerhard Gentzen in his work published in 1934/1935 gave a positive answer to existence of syntactical decidability. He invented proof systems for classical and intiutionistic logics, now called Gentzen style formalizations. They formed a basis for development of Automated Theorem Proving area of mathematics and computer science. We will study the Gentzen style formalizations in chapter **??**.

Definition 4.7 (Automated Systems)

A proof system $S = (\mathcal{L}, \mathcal{E}, LA, \mathcal{R})$ that is proven to be syntactically decidable or semi-decidable is called an automated proof systems.

Automated proof systems are also called *automated theorem proving systems, Gentzen style formalizations, syntactically decidable systems* and and we use all of these terms interchangeably.

Example 4.1 *Any complete Hilbert style proof system for classical propositional logic is an example of a decidable, but not syntactically decidable proof system. We conclude its decidability from the Completeness Theorem (to be proved in next chapter) and the decidability of the notion of classical tautology (proved in chapter 3).*

Example 4.2 *The Gentzen style proof systems for classical and intuiionistic propositional logics presented in chapter* **??**, *are examples of proof systems that are of both decidable and syntactically decidable.*

W are going to prove now, as a simple example t the following

Fact 4.1

The systems proof systems S_1 and S_2 defined by (7.2) and (7.3), respectively are syntactically decidable.

Proof Let's now to think how we can search for a proof in S_2 of a formula

$$PP((Pa \Rightarrow (b \Rightarrow c)).$$

If $PP((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c)))$ had the proof, the only last step in this proof would have been the application of the rule (r) to the formula $PP((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c)))$. This formula, in turn, if it had the proof, the only last step in its proof would have been the application of the rule r to the formula $P((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c)))$. And again, this one could be obtained only from the formula $((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c)))$ by the virtue of the rule r. Here the search process stops; the rule r puts P in front of the formulas, hence couldn't be applied here. The formula

$((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c)))$ isn't an axiom of S_2, what means that the only possible way of finding the proof has failed, i.e. we have proved that $\nvdash_{S_1} PPP((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c)))$.

The above example of proof search in S_2 defines the following an effective, automatic **Procedure** S_1, S_2 of searching for a proof of our formula in both our proof systems. If the search ends with an axiom, we have a proof, if it doesn't end with an axiom it means that the proof does not exists. We have described it, as an example, for one particular formula. It can be easily extended to any formula A of $\mathcal{L}_{\{P,\Rightarrow\}}$ as follows.

Procedure S_1, S_2

Step : Check the main connective of A.

If main connective is P, it means that A was obtained by the rule r.

Erase the main connective P.

Repeat until no P left.

If the main connective is $\Rightarrow$,check if a formula A is an axiom.

If it is an axiom , STOP and YES, we have a proof.

If it is not an axiom , STOP and NO, proof does not exist.

It is an effective, automatic procedure of searching for a proof of our formula in both our proof systems. This **ends the proof** .

Observe also, that the systems S_1 and S_2 are such that we can easily describe a general form of their provable formulas defined by (4.1) as $\mathbf{P}_S = \{E \in \mathcal{E} : \vdash_S E\}$. Namely we have the following.

$$\mathbf{P}_{S_1} = \{P^n(A \Rightarrow A) : \quad n \in N, \quad A \in \mathcal{F}\},$$

$$\mathbf{P}_{S_2} = \{P^n(a \Rightarrow a) : \quad n \in N, \quad a \in VAR\},$$

where P^n denotes n-iteration of P for $n \geq 1$ and P^0 denotes absence of P.

Obviously we have that $\mathbf{P}_{S_1} \neq \mathbf{P}_{S_2}$, and $\mathbf{P}_{S_2} \subseteq \mathbf{P}_{S_1}$.

The proof systems S_1 and S_2 are very simple, indeed. Here is an example of another two, similar but slightly more complex proof systems.

Consider two proof systems S_3 and S_4 of the language $\mathcal{L}_{\{\cup,\neg\}}$ with the set of expressions $\mathcal{E} = \mathcal{F}$ and is defined as follows.

$$S_3 = (\mathcal{L}_{\{\cup,\neg\}},\ \mathcal{F},\ \ \{(A\cup\neg A)\},\ (r)\ \frac{(A \cup \neg A)}{(B \cup (A \cup \neg A))},\ \text{for any } A, B \in \mathcal{F}\ ).\ (4.4)$$

$$S_4 = (\mathcal{L}_{\{\cup,\neg\}},\ \mathcal{F},\ \{(A\cup\neg A)\},\ (r)\ \frac{B}{(B\cup(A\cup\neg A))},\ \text{for any } A, B \in \mathcal{F}\ ),\ (4.5)$$

Exercise 4.2 *Given proof systems S_3 and S_4 defined by (4.4), (7.40), respectively.*

1. Describe the sets $\mathbf{P}_{S_3}, \mathbf{P}_{S_4}$ of provable formulas of S_3 and S_4.

2. Decide whether is it true/ false *that $\mathbf{P}_{S_3} = \mathbf{P}_{S_4}$. If yes, prove it, if not, give an example of a formula A such that $A \in \mathbf{P}_{S_4}$ and $A \notin \mathbf{P}_{S_3}$, or vice versa.*

Solution 1.
Let's first describe the set of provable formulas of both systems. Consider proof system S_3. Obviously, for any formula $A \in \mathcal{F}$, $(A \cup \neg A)$, as it is the axiom. It constitutes a proof of length 1 $A_1 = (A \cup \neg A)$and we have that

$$\vdash_{S_3}(A \cup \neg A).$$

One application of the inference rule (r) to axiom$(A \cup \neg A)$ gives us a proof $A_1 = (A \cup \neg A),\ A_2 = ((A \cup \neg A) \cup (A \cup \neg A))$, and hence

$$\vdash_{S_3}((A \cup \neg A) \cup (A \cup \neg A)).$$

The application of the rule (r) to the already proven above formula A_2 give us the proof $A_1 = (A \cup \neg A),\ A_2 = ((A \cup \neg A) \cup (A \cup \neg A)), A_3 = (((A \cup \neg A) \cup (A \cup \neg A)) \cup (A \cup \neg A))$, and

$$\vdash_{S_3}(((A \cup \neg A) \cup (A \cup \neg A)) \cup (A \cup \neg A)).$$

It is easy to see that all provable formulas of S_3 will be of the form of the proper disjunction of the axiom of S_3, what we denote as follows:

$$\mathbf{P}_{S_3} = \{\bigcup\nolimits_{n\in N}(A \cup \neg A)^n : A \in \mathcal{F}\}, \qquad (4.6)$$

where $(A \cup \neg A)^n$ denotes a disjunction of n formulas of the form $(A \cup \neg A)$ defined recursively as follows. $(A \cup \neg A)^0 = (A \cup \neg A)$, $(A \cup \neg A)^{n+1} = ((A \cup \neg A)^n \cup (A \cup \neg A))$.

Consider now system S_4. Obviously, as in the case of S_4, $\vdash_{S_4}(A \cup \neg A)$. One application of the inference rule to the axiom gives us the proof $A_1 = (A \cup \neg A),\ A_2 = (B \cup (A \cup \neg A))$ and we have that

$$\vdash_{S_4}(B \cup (A \cup \neg A)), \qquad (4.7)$$

where B can be any formula from $\mathcal{F}$.

The rule (r) can't be, by its definition, applied to already proved $B \cup (A \cup \neg A))$. We can continue with the proof A_1, A_2 by constructing for example a proof

A_1, A_2, A_3, A_4 by inserting axiom $(C \cup \neg C)$ (or axiom $(A \cup \neg A)$, if we wish as A_3 step of the proof. We have to remember that the definition 4.4 of the formal proof allows us to insert an axiom in any place within the proof. $A_1 = (A \cup \neg A),\ A_2 = (B \cup (A \cup \neg A)),\ A_3 = (C \cup \neg C),\ A_4 = (A \cup (C \cup \neg C))$ and hence

$$\vdash_{S_4} (A \cup (C \cup \neg C)),\ \vdash_{S_4} (B \cup (A \cup \neg A)), \vdash_{S_4} (C \cup (B \cup \neg B)), ...\text{etc...}$$

Multiple application of the rule (r) in S_4 means its application to multiple forms of the axiom. Finally it is clear that we can only construct formal proofs of all possible formulas of the form $(B \cup (A \cup \neg A))$, and of course of a form of any axiom (proofs of the length 1) $(A \cup \neg A)$ for A, B being all possible formulas. Remark that by saying $A, B \in \mathcal{F}$ we do not say that $A \neq B$, that we do not exclude that case $A = B$. In particular case we have that

$$\vdash_{S_4} (A \cup (A \cup \neg A)),\ \vdash_{S_4} (B \cup (B \cup \neg B)), \vdash_{S_4} (C \cup (C \cup \neg C)), ...\text{etc...}$$

Hence

$$\mathbf{P}_{S_4} = \{(B \cup (A \cup \neg A)) : \quad A, B \in \mathcal{F}\} \cup \{(A \cup \neg A) : \quad A \in \mathcal{F}\}. \tag{4.8}$$

Solution 2.
We prove now that $\mathbf{P}_{S_3} \subseteq \mathbf{P}_{S_4}$. Let $D \in \mathbf{P}_{S_3}$. By (4.6) $D = \bigcup_{n \in N} (A \cup \neg A)^n$. Observe that by definition $D = \bigcup_{n \in N} (A \cup \neg A)^n = (\bigcup_{n \in N} (A \cup \neg A)^{n-1} \cup (A \cup \neg A))$ and $\bigcup_{n \in N} (A \cup \neg A)^{n-1}$ is a formula of $\mathcal{L}_{\{\cup, \neg\}}$. We can denote it by B. We have proved in (4.7) that for any $B \in \mathcal{F}$, $\vdash_{S_4} (B \cup (A \cup \neg A))$. But by definition $D = (B \cup (A \cup \neg A))$, hence we proved that $D \in \mathbf{P}_{S_4}$.This ends the proof.

Consider a formula $((a \cup \neg b) \cup (a \cup \neg a))$ of $\mathcal{L}_{\{\cup, \neg\}}$. It has a following formal proof A_1, A_2 in S_4.

$$\begin{array}{cc} A_1 = (a \cup \neg a), & A_2 = ((a \cup \neg b) \cup (a \cup \neg a)). \\ \text{axiom for A=a} & \text{rule (r) application} \\ & \text{for } B = (a \cup \neg b) \end{array}$$

This proves that $((a \cup \neg b) \cup (a \cup \neg a)) \in \mathbf{P}_{S_4}$. Obviously $\nvdash_{S_3} ((a \cup \neg b) \cup (a \cup \neg a))$ and $((a \cup \neg b) \cup (a \cup \neg a)) \notin \mathbf{P}_{S_3}$. We have proved that the proof systems S_3 and S_4 defined by (4.4), (7.40) are such that $\mathbf{P}_{S_3} \subseteq \mathbf{P}_{S_4}$ and $\mathbf{P}_{S_3} \neq \mathbf{P}_{S_4}$.

Consider now a following proof system S_5.

$$S_5 = (\ \mathcal{L}_{\{\Rightarrow, \cup, \neg\}},\ \mathcal{F},\ \{(A \Rightarrow (A \cup B))\},\ \ \{(r1),\ (r2)\}\ ) \tag{4.9}$$

where the rules of inference are defined as follows.

$$(r1)\ \frac{A\ ; B}{(A \cup \neg B)}, \qquad (r2)\ \frac{A\ ; (A \cup B)}{B}.$$

Exercise 4.3

Given proof systems S_5 defined by (7.41).

1. Find a formal proof of a formula $\neg(A \Rightarrow (A \cup B))$ in S_5, i.e. show that $\vdash_{S_5} \neg(A \Rightarrow (A \cup B))$.

2. Find a formal proof of a formula $\neg((a \cup \neg b) \Rightarrow ((a \cup \neg b) \cup (a \cup \neg a)))$.

Solution

1. We construct a formal proof $B_1,\ B_2,\ B_3,\ B_4$ of the formula $\neg(A \Rightarrow (A \cup B))$ as follows. We write comments next to each step of the proof.

$B_1 = (A \Rightarrow (A \cup B))$ axiom, $B_2 = (A \Rightarrow (A \cup B))$ axiom,

$B_3 = ((A \Rightarrow (A \cup B)) \cup \neg(A \Rightarrow (A \cup B)))$ rule (r1) application to B_1 and B_2,

$B_4 = \neg(A \Rightarrow (A \cup B))$ rule (r2) application to B_3

for $A = (A \Rightarrow (A \cup B))$ and $B = \neg(A \Rightarrow (A \cup B))$.

2. We construct a formal proof $B_1,\ B_2,\ B_3,\ B_4$ of the formula $\neg((a \cup \neg b) \Rightarrow ((a \cup \neg b) \cup (a \cup \neg a)))$ as follows.

$B_1 = ((a \cup \neg b) \Rightarrow ((a \cup \neg b) \cup (a \cup \neg a)))$ axiom for $A = (a \cup \neg b)$ and $B = (a \cup \neg a)$,

$B_2 = ((a \cup \neg b) \Rightarrow ((a \cup \neg b) \cup (a \cup \neg a)))$ axiom for $A = (a \cup \neg b)$ and $B = (a \cup \neg a)$,

$B_3 = (((a \cup \neg b) \Rightarrow ((a \cup \neg b) \cup (a \cup \neg a))) \cup \neg((a \cup \neg b) \Rightarrow ((a \cup \neg b) \cup (a \cup \neg a))))$ rule (r1) application to B_1 and B_2,

$B_4 = \neg((a \cup \neg b) \Rightarrow ((a \cup \neg b) \cup (a \cup \neg a)))$ rule (r2) application to B_3

for $A = ((a \cup \neg b) \Rightarrow ((a \cup \neg b) \cup (a \cup \neg a)))$ and $B = \neg((a \cup \neg b) \Rightarrow ((a \cup \neg b) \cup (a \cup \neg a)))$.

Observation 4.1 *Observe that the formula $\neg((a \cup \neg b) \Rightarrow ((a \cup \neg b) \cup (a \cup \neg a)))$ is a particular case of the formula $\neg(A \Rightarrow (A \cup B))$ for $A = (a \cup \neg b)$ and $B = (a \cup \neg a)$ and its proof is just a particular case of the proof constructed in case 1.*

We wrote down independently a complicated proof of the particular case to make reader aware of a need of generalizing particular formulas, if it possible, and writing simpler proofs for the general case instead of the perticular.

4.1.1 Consequence Operation

Given a proof system $S = (\mathcal{L},\ \mathcal{E},\ LA, \mathcal{R})$. While proving expressions we often use some extra information available, besides the axioms of the proof system. This extra information is called *hypotheses* in the proof.

Let $\Gamma \subseteq \mathcal{E}$ be a set expressions called hypotheses. A proof from a *set* Γ *of hypothesis* of an expression $E \in \mathcal{E}$ in $S = (\mathcal{L}, LA, \mathcal{R})$ is a formal proof in S, where the expressions from Γ are treated as additional hypothesis added to the set LA of the logical axioms of the system S. We define it formally as follows.

Definition 4.8 (Proof from Hypotheses)

Given a proof system $S = (\mathcal{L},\ \mathcal{E},\ LA, \mathcal{R})$ and let Γ be any set of expressions of S, i.e. let $\Gamma \subseteq \mathcal{E}$.

A proof of an expression $E \in$ from the set Γ of expressions is a sequence

$$E_1, E_2,\ \dots\ E_n$$

of expressions, such that

$$E_1 \in LA \cup \Gamma, \qquad E_n = E$$

and for each i, $1 < i \leq n$, either $E_i \in LA \cup \Gamma$ or E_i is a direct consequence of some of the preceding expressions in the sequence $E_1, E_2,\ \dots\ E_n$ by virtue of one of the rules of inference from $\mathcal{R}$.

We write

$$\Gamma \vdash_S E$$

to denote that the expression E has a proof from Γ in S and $\Gamma\ \vdash\ E$, when the system S is fixed.

When the set of hypothesis Γ is a *finite set* and $\Gamma = \{B_1, B_2, ..., B_n\}$, then we write

$$B_1, B_2, ..., B_n \vdash_S E$$

instead of $\{B_1, B_2, ..., B_n\} \vdash_S E$. The case when Γ is an empty set i.e. when $\Gamma = \emptyset$ is a special one. By the definition of a proof of E from Γ, $\emptyset \vdash E$ means that in the proof of E only logical axioms LA of S were used. We hence write it as we did before

$$\vdash_S E$$

to denote that E has a proof from the empty set Γ. The set of all expressions provable from Γ (and logical axioms LA in S is denoted by $\mathbf{P}_S(\Gamma)$, i.e.

$$\mathbf{P}_S(\Gamma) = \{E \in \mathcal{E} :\ \ \Gamma \vdash_S E\}. \tag{4.10}$$

When discussing properties of provability in proof systems we often use a notion of a *consequence operation.* In order to follow this tradition we call provable expressions from Γ in S *consequences* of Γ. The set of all expressions provable is then called the *set of all consequences* from Γ. We observe that when talking about consequences of Γ in S, we define in fact a function which to every set $\Gamma \subseteq \mathcal{E}$ assigns a set of all its consequences. We denote this function by $\mathbf{Cn}_S$ and adopt the following definition.

Definition 4.9 (Consequence)

Given a proof system $S = (\mathcal{L},\ \mathcal{E},\ LA, \mathcal{R})$. *Any function* $\mathbf{Cn}_S\ : 2^{\mathcal{E}} \longrightarrow 2^{\mathcal{E}}$ *such that for every* $\Gamma \in 2^{\mathcal{E}}$,

$$\mathbf{Cn}_S(\Gamma) = \{E \in \mathcal{E} : \quad \Gamma \vdash_S E\} \tag{4.11}$$

is called a **consequence determined by** *S.*

Directly from definition 4.14 and (4.10) we have the following.

Fact 4.2 *For any proof system* $S = (\mathcal{L},\ \mathcal{E},\ LA, \mathcal{R})$.

$$\mathbf{P}_S(\Gamma) = \mathbf{Cn}_S(\Gamma). \tag{4.12}$$

It proves that the notions of provability from a set Γ in S and consequence determined by S coincide . It means that we can, and we often use in the literature the both terms interchangeably.

The definition 4.14 does do more then just re-naming "provability" by "consequence". We are going to prove now that the consequence $\mathbf{Cn}_S$ determined by S a special (an important) case of a notion a classic consequence operation as defined by Alfred Tarski in 1930 as a general model of deductive reasoning. Tarski definition is a formalization of the intuitive concept of the deduction as a consequence, and therefore it has all the properties which our intuition attribute to this notion. Here is the definition.

Definition 4.10 (Tarski)

By a **consequence operation** *in a formal language* $\mathcal{L} = (\mathcal{A}, \mathcal{F})$ *we understand any mapping* $\mathbf{C} : 2^{\mathcal{F}} \longrightarrow 2^{\mathcal{F}}$ *satisfying the following conditions (t1) - (t3) expressing properties of reflexivity, monotonicity, and transitivity of the consequence.*

For any sets $F,\ F_0,\ F_1,\ F_2,\ F_3 \in 2^{\mathcal{F}}$,

(t1) $F \subseteq \mathbf{C}(F)$ **reflexivity**,

(t2) if $F_1 \subseteq F_2$, *then* $\mathbf{C}(F_1) \subseteq \mathbf{C}(F_2)$, **monotonicity**.

(t3) if $F_1 \subseteq \mathbf{C}(F_2)$ *and* $F_2 \subseteq \mathbf{C}(F_3)$ *, then* $F_1 \subseteq \mathbf{C}(F_3)$*,* **transitivity**.

We say that the consequence operation **C** *has a* **finite character** *if additionally it satisfies the following condition* **t4**.

(t4) if a formula $B \in \mathbf{C}(F)$*, then there exists a finite set* $F_0 \subseteq F$*, such that* $B \in \mathbf{C}(F_0)$*.* **finiteness**.

The monotonicity condition (t2) and transitivity condition (t3) are often replaced by the following conditions **(t2')**, **(t3')**, respectively.

For any formula $B \in \mathcal{F}$, any any sets $F,\ F', \in 2^{\mathcal{F}}$,

$$(t2') \quad \text{if } B \in \mathbf{C}(F), \quad then\ B \in \mathbf{C}(F \cup F'), \tag{4.13}$$

$$(t3') \quad \mathbf{C}(F) = \mathbf{C}(\mathbf{C}(F)). \tag{4.14}$$

We express the correctness of the replacement conditions (4.13) and (4.14) in a form of a following theorem.

Theorem 4.1

The Tarski definition 4.10 is equivalent with definitions where one, or both conditions (t2), (t3) are replaced respectively by conditions (t2'), (t3')given by equations (4.13) and (4.14).

Proof We prove the equivalency of conditions (t1) - (t3) and (t1) - (t3'). We leave the proof of the other equivalency to the reader.

Assume (t3). By substituting

$$F_1 = \mathbf{C}(\mathbf{C}(F)),\ F_2 = \mathbf{C}(F),\ F_3 = F$$

in (t3) we obtain

$$\mathbf{C}(\mathbf{C}(F)) \subseteq \mathbf{C}(F).$$

On the other hand, it follows from (t1) and (t2)

$$\mathbf{C}(F) \subseteq \mathbf{C}(\mathbf{C}(F)),$$

which with the previous inclusion gives (t3'). Conversely, suppose that (t3') is satisfied. If $F_2 \subseteq \mathbf{C}(F_3)$, then by (t2) we obtain $\mathbf{C}(F_2) \subseteq \mathbf{C}(\mathbf{C}(F_3))$. By (t3') $\mathbf{C}(\mathbf{C}(F_3)) = (\mathbf{C}(F_3)$, hence $\mathbf{C}(F_2) \subseteq (\mathbf{C}(F_3)$ and we proved (t3).

The consequence operation provides a model describing what we intuitively call a deduction. It formalizes the basic, intuitively obvious and accepted properties of reasoning by which we obtain (deduce) new facts from already known, or assumed. We hence use it to define, after Tarski, a following notion of a deductive system.

Definition 4.11 (Deductive System)

Given a formal language $\mathcal{L} = (\mathcal{A}, \mathcal{F})$ and a Tarski consequence $\mathbf{C}$ *(definition 4.10). A system*

$$D = (\mathcal{L}, \mathbf{C})$$

is called a Tarski deductive system for the language $\mathcal{L}$.

Tarski's deductive system as a model of reasoning does not provide a method of actually defining a consequence operation in the language $\mathcal{L}$; it assumes that it is given. We are going to prove now that our definition 4.14 of consequence operation $\mathbf{Cn}_S$ determined by a proof system S is a Tarski consequence operation $\mathbf{C}$ in the definition 4.10 sense. It justifies, together with Fact 4.2 the common use the consequence notion when talking about provability. It means that each proof system S provides a different example of a consequence operation. They are all purely syntactic in nature and all defined by the notion of provability. Hence each proof system can be treated and a syntactic Tarski deductive system from definition 4.11.

Theorem 4.2

Given a proof system $S = (\mathcal{L},\ \mathcal{E},\ LA, \mathcal{R})$. The consequence operation $\mathbf{Cn}_S$ *is a Tarski consequence* $\mathbf{C}$ *in the language $\mathcal{L}$ of the system S and the system*

$$D_S = (\mathcal{L},\ \mathbf{Cn}_S)$$

is Tarski deductive system. We call it a **syntactic deductive system** *determined by S. Moreover, the consequence operation* $\mathbf{Cn}_S$ *that has a* **finite character**.

Proof
By definition 4.14, the consequence operation $\mathbf{Cn}_S\ :\ 2^{\mathcal{E}}\ \longrightarrow\ 2^{\mathcal{E}}$ is given by a formula $\mathbf{Cn}_S(\Gamma) = \{E \in \mathcal{E} :\ \Gamma \vdash_S E\}$. We have to show that for any $\Gamma,\ \Gamma_0,\ \Gamma_1,\ \Gamma_2,\ \Gamma_3 \in 2^{\mathcal{F}}$ conditions (t1) - (t4) of the definition 4.14 hold. The reflexivity condition (t1) becomes $\Gamma \subseteq \mathbf{Cn}_S(\Gamma)$. Let $E \in \Gamma$. The one element sequence E is a proof of E from Γ, hence we proved that $E \in\ \mathbf{C}(\Gamma)$ and (t1) holds. To prove the transitivity condition (t2) assume now that $\Gamma_1 \subseteq \Gamma_2$. Let $E \in \mathbf{Cn}_S(\Gamma_1)$. It means that $\Gamma_1 \vdash_S E$, i.e E has a formal proof from Γ_1, but $\Gamma_1 \subseteq \Gamma_2$, hence this proof also is a proof from Γ_2, and $E \in \mathbf{Cn}_S(\Gamma_2)$. This proves that $\mathbf{Cn}_S(\Gamma_1) \subseteq \mathbf{Cn}_S(\Gamma_2)$ and the condition (t2) holds. Let now $E \in \Gamma_1$ and $\Gamma_1 \subseteq \mathbf{Cn}_S(\Gamma_2)$, so $E \in \mathbf{Cn}_S(\Gamma_)2$. Let $E_1, \ldots, E_n$ be a formal proof of E from Γ_2. But $\Gamma_2 \subseteq \mathbf{Cn}_S(\Gamma_3)$. It means that any expression from Γ_2 has a formal proof from Γ_3. In particular, all expression in the proof $E_1, \ldots, E_n$ that belong to Γ_2 have their formal proofs from Γ_3. Replacing all these expressions by their proofs from Γ_3 we obtain a proof of E from Γ_3. This proves that $\Gamma_1 \subseteq \mathbf{Cn}_S\Gamma_3$ and the transitivity condition (t3) holds. Let now $E \in \mathbf{Cn}_S\Gamma$. This means that E has a proof $E_1, \ldots, E_n$ from Γ. The set $\Gamma_0 = \{E_1, \ldots, E_n\}$ is obviously a

finite subset of Γ and $E \in \mathbf{Cn}_S\Gamma_0$ and (t4) holds.

Non - Monotonic Logics
The Tarski definition 4.10 of a consequence models reasoning which is called after its condition (t2) or (t2') a monotonic reasoning. The monotonicity of reasoning was, since antiquity is the the basic assumption while developing models for classical and well established non-classical logics. Recently many of new non- classical logics were developed and are being developed by computer scientists. For example new modal logics of agents and temporal logics. Temporal logics are essential in developing theory of programs. Nevertheless they all are built following the Tarski definition of consequence and were and are called *monotonic logics*. A new type of important

Non-Monotonic logics have been proposed at the beginning of the 80s. Historically the most important proposals are Non-monotonic logic, by McDermott and Doyle, Default Logic, by Reiter, Circumscription, by McCarthy, and Autoepistemic logic, by Moore.

The term non-monotonic logic covers a family of formal frameworks devised to capture and represent defeasible inference. It is an inference in which it is possible to draw conclusions tentatively, reserving the right to retract them in the light of further information. We included most standard examples in Chapter 1, Introduction.

Non-monotonic logics describe commonsense reasoning which is neither a restriction nor an extension of classical logic. Consequences of premises are drawn as much due to the absence as to the presence of knowledge. When more knowledge is acquired, conclusions previously drawn may have to be withdrawn because the rules of inference that led to them no longer are active. Intelligent decision makers use this form of commonsense reasoning to infer actions to be performed from premises which cannot be made by classical logic inference, because they simply have to make decisions whether or not there is enough information for a classical logical deduction. Non-monotonic reasoning deals with the problem of deriving plausible conclusions, but not infallible, from a knowledge base (a set of formulas). Since the conclusions are not certain, it must be possible to retract some of them if new information shows that they are wrong. Example: let the KB contain: Typically birds fly. Penguins do not fly. Tweety is a bird. It is plausible to conclude that Tweety flies. However if the following information is added to KB Tweety is a penguin the previous conclusion must be retracted and, instead, the new conclusion that Tweety does not fly will hold.

The statement "typically A" can be read as: "in the absence of information to the contrary, assume A". The problem is to define the precise meaning of "in the absence of information to the contrary". The meaning could be: "there is nothing in KB that is inconsistent with assumption A". Other interpretations are possible Different interpretations give rise to different non-monotonic logics.

4.1.2 Syntactic Consistency

Formal theories play crucial role in mathematics and were historically defined for classical first order logic and consequently for other first and higher order logics. They are routinely called *first order theories.* We will discuss them in more detail in chapter 10 dealing formally with classical predicate logic. First order theories are hence based on proof systems S with a predicate (first order) language $\mathcal{L}$. We will call them for short *first order proof systems.*

We can and we sometimes consider formal theories based on propositional logics, i.e. based on proof systems with language $\mathcal{L}$ being propositional. We will call them *propositional theories.*

Given a proof system $S = (\mathcal{L},\ \mathcal{E},\ LA,\ \mathcal{R})$. We build (define) a **formal theory** based on S as follows.

1. We select a certain *finite subset* SA of expressions of S, disjoint with the logical axioms LA of S, i.e. such that $LA \cap SA = \emptyset$.. The set SA is called a set of *specific axioms* of the formal theory based on S.

2. We use set SA of specific axioms to define a language

$$\mathcal{L}_{SA}, \tag{4.15}$$

called a *language of the formal theory.* Here we have two cases.

c1. S is a *first order proof system*, i.e. $\mathcal{L}$ of S is a predicate language. We define the language $\mathcal{L}_{SA}$ by restricting the sets of constant, functional, predicate symbols of $\mathcal{L}$ to constant, functional, predicate symbols appearing in the set SA of specific axioms. Both languages $\mathcal{L}_{SA}$ and $\mathcal{L}$ share the same set of *propositional connectives.*

c2. S is a *propositional proof system*, i.e. $\mathcal{L}$ of S is a propositional language. $\mathcal{L}_{SA}$ is defined by restricting $\mathcal{L}$ to connectives appearing in the set SA.

Definition 4.12 (Formal Theory)

The system

$$T = (\mathcal{L},\ \mathcal{E},\ LA,\ SA,\ \mathcal{R}) \tag{4.16}$$

is called a **formal theory** *based on a proof system S.*

The set SA of the set of **specific axioms** *of T. The language $\mathcal{L}_{SA}$ defined by (4.15) is called the* **language of the theory** *T.*

The set $\mathcal{E}_{SA}$ of all expressions of the language $\mathcal{L}_{SA}$ provable from the set specific axioms SA (and logical axioms LA) i.e. the set

$$\mathbf{T}(SA) = \{E \in \mathcal{E}_{SA} :\quad SA\ \vdash_S E\ \} \tag{4.17}$$

is called the set of all **theorems** of the theory T.

If the set SA of specific axioms of T is empty, then the theory T is, by definition, identified with the system S, i.e. $T = S = (\mathcal{L},\ \mathcal{E},\ LA,\ \mathcal{R})$.

Definition 4.13 (Consistent Theory)

A theory $T = (\mathcal{L},\ \mathcal{E},\ LA,\ SA,\ \mathcal{R})$ *is* **consistent** *if there exists an expression* $E \in \mathcal{E}_{SA}$ *such that* $E \notin \mathbf{T}(SA)$*, i.e. such that*

$$SA\ \nvdash_S E;$$

otherwise the theory T *is* **inconsistent***.*

Observe that the definition 4.13 has purely syntactic meaning. It also reflexes our intuition what proper provability should mean. it says that a formal theory T based on a proof system S is consistent only when it does not prove all expressions (formulas in particular cases) of $\mathcal{L}_{SA}$. The theory T such that it proves everything stated in $\mathcal{L}_{SA}$ obviously should be, and its defined as inconsistent. In particular, we have the following *syntactic definition* of consistency-inconsistency for any proof system S.

Definition 4.14 (Consistent Proof System)

A proof system $S = (\mathcal{L},\ \mathcal{E},\ LA,\ \mathcal{R})$ *is* **consistent** *if there exists* $E \in \mathcal{E}$ *such that* $E \notin \mathbf{P}_S$*, i.e. such that* $\nvdash_S E$; *otherwise* S *is* **inconsistent***.*

4.2 Semantics

We define formally a semantics for a given proof system $S = (\mathcal{L}, \mathcal{E}, LA, \mathcal{R})$ by specifying the semantic links of all its components as follows.

Semantic Link1: Language $\mathcal{L}$

The language $\mathcal{L}$ of S can be propositional or predicate. Let denote my $\mathbf{M}$ a semantic for L. We call it, for short, a semantics for the proof system S. The semantics $\mathbf{M}$ can be a propositional, a predicate, extensional, or not extensional. We use $\mathbf{M}$ as a general symbol for a semantics.

Semantic Link 2: Set $\mathcal{E}$ of Expressons

We always have to extend a given semantics $\mathbf{M}$ of the language $\mathcal{L}$ to the set of expressions $\mathcal{E}$ of the system S. We often do it by establishing a semantic equivalency under semantics $\mathbf{M}$ of $\mathcal{E}$ and the set of all formulas $\mathcal{F}$ of $\mathcal{L}$. It means we prove that for a given semantics M under which we build our proof system S, and for any expression $E \in \mathcal{E}$ there is a formula $A \in \mathcal{F}$, such that $E \equiv_M A$. For example, in the automated theorem proving system **RS** presented in chapter 6 the expressions are *finite sequences of formulas* of $\mathcal{L} = \mathcal{L}_{\neg,\cap,\cup,\Rightarrow}$. We extend our classical semantics for $\mathcal{L}$ to the set $\mathcal{F}^*$ of all finite sequences of formulas as follows: for any $v : VAR \longrightarrow \{F, T\}$ and any $\Delta \in \mathcal{F}^*$, $\Delta = A_1, A_2, ..A_n$, $v^*(\Delta) = v^*(A_1, A_2, ..A_n) = v^*(A_1) \cup v^*(A_2) \cup .... \cup v^*(A_n)$, i.e. $\Delta \equiv (A_1 \cup A_2 \cup ... \cup A_n)$. Sometimes, like in case of Resolution based proof systems we have also to prove a semantic equivalency of a given formula A of $\mathcal{L}$ with some set $\mathcal{E}_A$ of expressions (sets of clauses) representing the formula A.

Semantic Link 3: Logical Axioms LA

Given a semantics **M** for $\mathcal{L}$ and its extension to the set $\mathcal{E}$ of all expressions. We extend the notion of tautology to the set $\mathcal{L}$ of expressions and write $\models_{\mathbf{M}} E$ to denote that the expression $E \in \mathcal{E}$ is a tautology under semantics **M**. We denote

$$\mathbf{T_M} = \{E \in \mathcal{E} : \ \models_{\mathbf{M}} E\}$$

While designing a proof system S we want the logical axioms LA to be a subset of expressions that are tautologies of under the semantics **M**, i.e.

$$LA \subseteq \mathbf{T_M}.$$

We can, and we often do, invent proof systems with languages without yet established semantics. In this case the logical axioms LA serve as description of properties of tautologies under a future semantics yet to be built. We want to choose as logical axioms of a proof system S are not only tautologies under an already known semantics **M**, but they can also guide us how to define a semantics when it is yet unknown.

Semantic Link 4: Rules of Inference $\mathcal{R}$

We want the rules of inference $r \in \mathcal{R}$ to preserve truthfulness. Rules that preserve the truthfulness are called sound under a given semantics **M**. We put it in a general formal definition as follows.

Definition 4.15 (Sound Rule under M)

Given an inference rule $r \in \mathcal{R}$ of the form

$$(r) \quad \frac{P_1 \ ; \ P_2 \ ; \ .... \ ; \ P_m}{C}.$$

We say that the rule (r) is **sound** *under a semantics* **M** *if the following condition holds for all* **M** *models $\mathcal{M}$.*

$$\textit{If } \ \mathcal{M} \models_{\mathbf{M}} \{P_1, P_2, .P_m\} \ \textit{ then } \ \mathcal{M} \models_{\mathbf{M}} C. \tag{4.18}$$

In case of a propositional language $\mathcal{L}_{CON}$ and an extensional semantics **M** the **M** models $\mathcal{M}$ are defined in terms of the truth assignment $v : VAR \longrightarrow LV$, where LV is the set of logical values with a distinguished value T. The general definition 4.15 becomes a following definition for a propositional language $\mathcal{L}$ and its extensional semantics **M**.

Definition 4.16 (Sound Propositional Rule under M)

Given a propositional language $\mathcal{L}_{CON}$ and an extensional semantics **M***, an inference rule of the form*

$$(r) \qquad \frac{P_1 \;\; ; \;\; P_2 \;\; ; \;\; .... \;\; ; \;\; P_m}{C}$$

is **sound** *under the semantics* **M** *if the following condition holds for any $v : VAR \longrightarrow LV$.*

$$\text{If } \; v \models_{\mathbf{M}} \{P_1, P_2, \ldots, P_m\}, \;\; \text{then } \; v \models_{\mathbf{M}} C. \qquad (4.19)$$

Observe that we can rewrite the condition (4.19) as follow.

$$\text{If } \; v^*(P_1) = v^*(P_2) = \ldots. = v^*(P_m) = T, \;\; \text{then } \; v^*(C) = T. \qquad (4.20)$$

A rule of inference be sound under different semantics, but also rules of inference can be sound under one semantics and not sound under the other.

Example 4.3 *Given a propositional language $\mathcal{L}_{\{\neg,\cup,\Rightarrow\}}$. Consider two rules of inference:*

$$(r1) \; \frac{(A \Rightarrow B)}{(B \Rightarrow (A \Rightarrow B))} \qquad and \qquad (r2) \; \frac{\neg\neg A}{A}.$$

The rule (r1) is sound under classical, **H** *and* **L** *semantics. The (r2) is sound under classical and* **L** *semantics but is not sound under* **H** *semantics.*

Consider the rule (r1).

Let $v : VAR \longrightarrow \{F, T\}$ be any truth assignment, such that $v^*((A \Rightarrow B)) = T$. We use condition (4.20) and evaluate logical value of the conclusion under v as follows. $v^*((B \Rightarrow (A \Rightarrow B))) = v^*(B) \Rightarrow T = T$ for any formula B and any value of $v^*(B)$. This proves that $v^*(B \Rightarrow (A \Rightarrow B)) = T$ and hence the soundness of (r1). Consider now the **H** semantics. Let $v : VAR \longrightarrow \{F, \perp, T\}$ be any truth assignment, such that $v \models_{\mathbf{H}} (A \Rightarrow B)$, i.e. such that $v^*((A \Rightarrow B)) = T$. We evaluate under **H, L** semantics $v^*((B \Rightarrow (A \Rightarrow B))) = v^*(B) \Rightarrow T$. Now $v^*(B)$ can be T, F as in classical case, or $v^*(B) = \perp$. The case when $v^*(B)$ is T, F is like in classical semantics, so we have to check the case $v^*(B) = \perp$. But in both **H** and **L** semantics $\perp \Rightarrow T = T$. This proves that (r1) is also sound under **H** and **L** semantics.

Consider the rule (r2).

The rule (r2) is sound under classical and **L** by straightforward eveluation. Assume now $v : VAR \longrightarrow \{F, \perp, T\}$ be any truth assignment, such that $v \models_{\mathbf{M}} \neg\neg A$, i.e. such that $v^*(\neg\neg A) = T$ under **H** semantics. We have that $v^*(\neg\neg A) = \neg\neg v^*(A) = T$ if and only if $\neg v^*(A) = F$ if and only if $v^*(A) = T$ or $v^*(A) = \perp$. This proves that that it is possible to have $v \models_{\mathbf{M}} \neg\neg A$ and $v \not\models_H A$, i.e. that (r2) is not sound.

Definition 4.17 (Strongly Sound Rule under M)

An inference rule $r \in \mathcal{R}$ of the form

$$(r) \qquad \frac{P_1 \ ; \ P_2 \ ; \ .... \ ; \ P_m}{C}$$

is **strongly sound** *under a semantics* **M** *if the following condition holds for all* **M** *structures $\mathcal{M}$,*

$$\mathcal{M} \models_{\mathbf{M}} \{P_1, P_2, .P_m\} \quad \text{if and only if} \quad \mathcal{M} \models_{\mathbf{M}} C. \tag{4.21}$$

In case of a propositional language $\mathcal{L}_{CON}$ and an extensional semantics **M** *the condition (4.21) is as follows. For for any $v : VAR \longrightarrow LV$,*

$$v \models_{\mathbf{M}} \{P_1, P_2, .P_m\} \quad \text{if and only if} \quad v \models_{\mathbf{M}} C. \tag{4.22}$$

We introduce also a property of a rule being **s-strongly sound**. We state it informally as follows. An inference rule $r \in \mathcal{R}$ is s-strongly sound when the conjunction of all its premisses is logically equivalent (under a semantics **M**) to its conclusion. We denote it informally as

$$P_1 \cap P_2 \cap \ \dots \ \cap P_m \equiv_{\mathbf{M}} C. \tag{4.23}$$

Of course any s-strongly sound rule is strongly sound.

Example 4.4

Given a propositional language $\mathcal{L}_{\{\neg,\cup,\Rightarrow\}}$. Consider two rules of inference:

$$(r1) \ \frac{A \ ; \ B}{(A \cup \neg B)} \qquad \text{and} \qquad (r2) \ \frac{A}{\neg\neg A}.$$

Both rules (r1) and (r2) are sound under classical and **H** *semantics. The rule (r2) is strongly and s-strongly sound under classical semantics but is not strongly sound under* **H** *semantics. The rule (r1) in not strongly sound under either semantics.*

Consider (r1). Take (in shorthand notation) for $A = T$ and $B = T$. We evaluate $v^*((A \cup \neg B)) = T \cup F = F$ in both semantics. This proves soundness of (r1) under both semantics.. Take now v such that $v(A) = T$ and $v(B) = F$, we get $v^*((A \cup \neg B)) = F \cup T = T$.This proves that $v \models (A \cup \neg B)$ and $v \models_{\mathbf{H}} (A \cup \neg B)$. Obviously $v \not\models \{A, B\}$ and $v \not\models_{\mathbf{H}} \{A, B\}$. This proves that (r1) in *not strongly sound* under either semantics.

Consider (r2). It is strongly sound under classical semantic. By (8.77) and the fact that $A \equiv \neg\neg A$ (r2) is *s-strongly sound* and so is strongly sound.

(r2) is sound under **H** semantics. Assume $A = T$. We evaluate (in shorthand notation) $\neg\neg A = \neg\neg T = \neg F = T$. (r2) is not strongly sound under **H** semantics. Take v such that $v^*(A) = \perp$, then $v^*(\neg\neg A) = \neg\neg \perp = \neg F = T$. This proves that there is v such that $v \models \neg\neg A$ and $v \not\models A$ and (4.22) does not hold and so (r2) is *not strongly sound.*

This also proves that $A \not\equiv_H \neg\neg A$, i.e. (r2) is not s-strongly sound (4.22).

Now we are ready to define a notion of a sound and strongly sound proof system. Strongly sound proof systems play a role in constructive proofs of completeness theorem. This is why we introduced and singled them out here.

Definition 4.18 (Sound, Strongly Sound)

Given a proof system $S = (\mathcal{L},\ \mathcal{E},\ LA,\ \mathcal{R})$.

We say that the proof system S is **sound** *under a semantics* **M** *if the following conditions hold.*

C1. $LA \subseteq \mathbf{T_M}$; **C2.** *Each rule of inference $r \in \mathcal{R}$ is* **sound** *under* **M**.

The proof system S is **strongly sound** *under a semantics* **M** *if the condition* **C2.** *is replaced by the following condition*

C2'. *Each rule of inference $r \in \mathcal{R}$ is* **strongly sound** *under* **M**.

Here is a simple but an important fact about the notion of strong soundness. The strong soundness of proof systems guarantee a correctness of of constructive proofs of completeness theorems for automated theorem proving systems introduced and examined in chapter 6 and in chapter 10.

Theorem 4.3 (Counter-Model)

Given a proof system $S = (\mathcal{L},\ \mathcal{E},\ LA,\ \mathcal{R})$ that is **strongly sound** *under a semantics* **M**. *For any $r \in \mathcal{R}$, for any $\mathcal{M}$ that is a* **M** *counter model for one of its premisses, $\mathcal{M}$ is also the* **M** *counter model for its conclusion.*

Proof
Consider a rule

$$(r) \qquad \frac{P_1 \ ; \ P_2 \ ; \ .... \ ; \ P_m}{C}.$$

By definition 4.17, (r) is sound and additionally the following implication holds for any **M** structures $\mathcal{M}$,

$$\text{if } \mathcal{M} \models_{\mathbf{M}} C, \text{ then } \mathcal{M} \models_{\mathbf{M}} \{P_1, P_2, .P_m\}.$$

It means that inverse implication

$$\text{if } \mathcal{M} \not\models_{\mathbf{M}} \{P_1, P_2, .P_m\}, \text{ then } \mathcal{M} \not\models_{\mathbf{M}} C$$

holds for all **M** structures $\mathcal{M}$. But $\mathcal{M} \not\models_{\mathbf{M}} \{P_1, P_2, .P_m\}$ if and only if there is $1 \leq i \leq m$ such that $\mathcal{M} \not\models_{\mathbf{M}} P_i$. This ends the proof.

Example 4.5 *The proof system S defined below as follows*

$$S = (\mathcal{L}_{\{\neg,\Rightarrow\}},\ \mathcal{F},\ \{(\neg\neg A \Rightarrow A),\ (A \Rightarrow (\neg A \Rightarrow B))\},\ \mathcal{R} = \{(r)\})$$

where

$$(r)\ \frac{(A \Rightarrow B)}{(B \Rightarrow (A \Rightarrow B))}.$$

S is sound, but not strongly sound under classical and **L** *semantics. It is not sound under* **H** *semantics.*

C1. Both axioms are basic classical tautologies. Hence to prove that first axiom is **L** tautology we we have to verify only the case (shorthand notation) $A = \perp$. But $\neg\neg \perp \Rightarrow \perp = \neg \perp \Rightarrow \perp = \perp \Rightarrow \perp = T$ and we proved $\models_{\mathbf{L}} (\neg\neg A \Rightarrow A)$. Observe that $(A \Rightarrow (\neg A \Rightarrow B)) = \perp$ if and only if $A = T$ and $(\neg A \Rightarrow B) = \perp$ if and only if $(\neg T \Rightarrow B) = \perp$ if and only if $(F \Rightarrow B) = \perp$, what is impossible undef **L** semantics. Hence $\models_{\mathbf{L}} (A \Rightarrow (\neg A \Rightarrow B))$. We prove, as in example 4.3, that $\not\models_{\mathbf{H}} (\neg\neg A \Rightarrow A)$, and hence S is not sound under **H** semantics.

C2. The rule (r) is not strongly sound under neither classical nor **L** semantic. Let $v : VAR \longrightarrow \{F,\ \perp,\ T\}$ be any truth assignment, such that $v^*(B) = F$ and $v^*(A) = F$. Obviously $v \models (B \Rightarrow (A \Rightarrow B))$ and $v \models_{\mathbf{L}} (B \Rightarrow (A \Rightarrow B))$, but $v \not\models (A \Rightarrow B)$ and $v \not\models_{\mathbf{L}} (A \Rightarrow B)$.

Nevertheless, (r) is sound under the both semantics by example 4.3.

Let $\mathbf{P}_S$ be the set of all provable expressions of S, i.e. $\mathbf{P}_S = \{E \in \mathcal{E} :\ \vdash_S E\}$. Let $\mathbf{T}_{\mathbf{M}}$ be a set of all expressions of S that are tautologies under a semantics **M**, $\mathbf{T}_{\mathbf{M}} = \{E \in \mathcal{E} :\ \models_{\mathbf{M}} E\}$.

When we define (develop) a proof system S our first goal is to make sure that it a "sound one", i.e. that all we prove in it is true (with respect to a given semantics). Proving the following theorem establishes this goal.

Theorem 4.4 (Soundness Theorem)

Given a predicate proof system S and a semantics $\mathbf{M}$*.*
The following holds.

$$\mathbf{P}_S \subseteq \mathbf{T}_{\mathbf{M}}, \tag{4.24}$$

i.e. for any $E \in \mathcal{E}$*, the following implication holds*

$$\textit{if} \ \vdash_S E \quad \textit{then} \quad \models_{\mathbf{M}} E.$$

Proof We prove by Mathematical Induction over the length of a proof that if S is sound as stated in the definition 8.27, the Soundness Theorem holds for S. It means that in order to prove the Soundness Theorem (under semantics bf M) for a proof system we have to verify the two conditions: **1.** $LA \subseteq \mathbf{T}_{\mathbf{M}}$ and **2.** Each rule of inference $r \in \mathcal{R}$ is **sound** under $\mathbf{M}$.

The next step in developing a logic is to answer next necessary and a difficult question: *Given a proof system S, about which we know that all it proves it true (tautology)with respect to a given semantics. Can we prove all we know to be true (all tautologies) with respect to the given semantics?*

Theorem 4.5 (Completeness Theorem)

Given a predicate proof system S and a semantics $\mathbf{M}$*.*
The following holds

$$\mathbf{P}_S = \mathbf{T}_{\mathbf{M}} \tag{4.25}$$

i.e. for any $E \in \mathcal{E}$*, the following holds*

$$\vdash_S E \quad \textit{if and only if} \quad \models_{\mathbf{M}} E.$$

The Completeness Theorem consists of two parts:

Part 1: Soundness Theorem: $\mathbf{P}_S \subseteq \mathbf{T}_{\mathbf{M}}$.

Part 2: Completeness Part of the Completeness Theorem: $\mathbf{T}_{\mathbf{M}} \subseteq \mathbf{P}_S$.

Proving the Soundness Theorem for S under a semantics M is usually a straightforward and not a very difficult task. Proving the *Completeness Part* of the Completeness Theorem is always a crucial and very difficult task. There are many methods and techniques for doing so, even for classical proof systems (logics) alone. Non-classical logics often require new sometimes very sophisticated methods. We will study two proofs of the Completeness Theorem for classical propositional Hilbert style proof system in chapter 5, and a constructive proofs for automated theorem proving systems for classical logic the chapter 6. We prove provide the proofs of the Completeness Theorem for classical predicate logic in chapter 9 (Hilbert style) and chapter 10(Gentzen style).

4.3 Exercises and Examples

Exercise 4.4

Given a proof system:

$$S = (\mathcal{L}_{\{\neg,\Rightarrow\}},\ \mathcal{E} = \mathcal{F}\ LA = \{(A \Rightarrow A), (A \Rightarrow (\neg A \Rightarrow B))\},\ (r)\ \frac{(A \Rightarrow B)}{(B \Rightarrow (A \Rightarrow B))}).$$

1. Prove that S is sound, but not **strongly sound** *under classical semantics.*

2. Prove that S is not sound under **K** *semantics.*

3. Write a formal proof in S with 2 applications of the rule (r).

Solution
Parts 1 and 2. In order to prove 1. and 2. we have to verify conditions **1.**, **2.** and bf 2.' of definition 8.27. Observe that both axioms of S are basic classical tautologies. Consider the rule of inference of S.

$$(r)\ \frac{(A \Rightarrow B)}{(B \Rightarrow (A \Rightarrow B))}.$$

Take any v such that $v^*((A \Rightarrow B))) = T$. We evaluate logical value of the conclusion under the truth assignment v as follows.

$$v^*(B \Rightarrow (A \Rightarrow B)) = v^*(B) \Rightarrow T = T$$

for any B and any value of $v^*(B)$. This proves that S is sound under classical semantics. S is not strongly sound as $(A \Rightarrow B) \not\equiv (B \Rightarrow (A \Rightarrow B))$.

System S is *not sound* under **K** semantics because axiom $(A \Rightarrow A)$ is not a **K** semantics tautology.

Part 3. There are many solutions, i.e. one can construct many forrmal proofs. Here is one of them. For example, one of the formal proofs is a sequence $A_1, A_2, A_3,$ where
$A_1 = (A \Rightarrow A)$
(Axiom)
$A_2 = (A \Rightarrow (A \Rightarrow A))$
Rule (r) application 1 for $A = A,\ B = A$.
$A_3 = ((A \Rightarrow A) \Rightarrow (A \Rightarrow (A \Rightarrow A)))$
Rule (r) application 2 for $A = A, B = (A \Rightarrow A)$.

Exercise 4.5

Prove, by constructing a formal proof that

$$\vdash_S ((\neg A \Rightarrow B) \Rightarrow (A \Rightarrow (\neg A \Rightarrow B))),$$

where S is the proof system from Exercise 5.9.

Solution
Required formal proof is a sequence $A_1, A_2,$ where
$A_1 = (A \Rightarrow (\neg A \Rightarrow B))$
Axiom
$A_2 = ((\neg A \Rightarrow B) \Rightarrow (A \Rightarrow (\neg A \Rightarrow B)))$
Rule (r) application for $A = A, B = (\neg A \Rightarrow B)$.

Observe that we needed only one application of the rule (r). One more application of the rule (r) to A_2 gives another solution to Exercise 5.9, namely a proof A_1, A_2, A_3 for A_1, A_2 defined above and
$A_3 = ((A \Rightarrow (\neg A \Rightarrow B)) \Rightarrow (\neg A \Rightarrow B) \Rightarrow (A \Rightarrow (\neg A \Rightarrow B)))$
Rule (r) application for $A = (\neg A \Rightarrow B)$ and $B = (A \Rightarrow (\neg A \Rightarrow B))$.

Exercise 4.6

Given a proof system:

$$S = (\mathcal{L}_{\{\cup,\Rightarrow\}}, \quad \mathcal{F}, \quad LA = \{A1, A2\}, \quad (r)\ \frac{(A \Rightarrow B)}{(A \Rightarrow (A \Rightarrow B))}\ ),$$

where $A1 = (A \Rightarrow (A \cup B)), \quad A2 = (A \Rightarrow (B \Rightarrow A)).$

*1. Prove that S is sound under classical semantics and determine whether S it is sound or not sound under **K** semantics.*

2. Write a formal proof B_1, B_2, B_3 in S with 2 applications of the rule (r) that starts with axiom A1, i.e such that $B_1 = A1$.

3. Write a formal proof B_1, B_2 in S with 1 application of the rule (r) that starts with axiom A2, i.e such that $A_1 = A2$.

Solution
Part 1. Axioms of S are basic classical tautologies. The proof (in shorthand notation) of soundness of the rule of inference is the following. Assume $(A \Rightarrow B) = T$. Hence the logical value of conclusion is $(A \Rightarrow (A \Rightarrow B)) = (A \Rightarrow T) = T$ for all A. S is *not sound* under **K** semantics. Let's take truth assignment such that $A = \perp, B = \perp$. The logical value of axiom A1 is as follows.
$(A \Rightarrow (A \cup B)) = (\perp \Rightarrow (\perp \cup \perp)) = \perp$ and $\not\models_{\mathbf{K}} (A \Rightarrow (A \cup B))$. Observe that the v such that $A = \perp, B = \perp$ is not the only v that makes $A1 \neq T$, i.e. proves that $\not\models_{\mathbf{K}} A1$.
$(A \Rightarrow (A \cup B)) \neq T$ if and only if $(A \Rightarrow (A \cup B)) = F$ or $(A \Rightarrow (A \cup B)) = \perp$. The first case is impossible because $A1$ is a classical tautology. Consider the second case. $(A \Rightarrow (A \cup B)) = \perp$ in two cases. c1. $A = \perp$ and $(A \cup B) = F$, i.e. $(\perp \cup B) = F$, what is impossible. c2. $A = T$ and $(A \cup B) = \perp$, i.e. $(T \cup B) = \perp$, what is impossible. c3. $A = \perp$ and $(A \cup B) = \perp$, i.e. $(\perp \cup B) = \perp$. This is possible for $B = \perp$ or $B = F$, i.e when $A = \perp, B = \perp$ or $A = \perp, B = F$. From the above observation we get a *second solution.* S is not sound under **K** semantics. Axiom A1 is not **K** semantics tautology. There are exactly two truth assignments v,

such that $v \not\models A1$. One is, as defined in the first solution, namely $A = \perp, B = \perp$. The second is $A = \perp, B = F$.

Part 2. The formal proof B_1, B_2, B_3 is as follows.
$B_1 = (A \Rightarrow (A \cup B))$
Axiom
$B_2 = (A \Rightarrow (A \Rightarrow (A \cup B)))$
Rule (r) application for $A = A$ and $B = (A \cup B)$
$B_3 = (A \Rightarrow (A \Rightarrow (A \Rightarrow (A \cup B))))$
Rule (r) application for $A = A$ and $B = (A \Rightarrow (A \cup B))$.

Part 3. The formal proof B_1, B_2 is as follows.
$B_1 = (A \Rightarrow (B \Rightarrow A))$
Axiom
$B_2 = (A \Rightarrow (A \Rightarrow (B \Rightarrow A)))$.
Rule (r) application for $A = A$ and $B = (B \Rightarrow A)$.

Exercise 4.7

Let S be the following proof system:

$$S = (\ \mathcal{L}_{\{\Rightarrow, \cup, \neg\}},\ \ \mathcal{F},\ \ A1,\ \ (r1),\ (r2)\ ),$$

where the logical axiom A1 is: $A1 = (A \Rightarrow (A \cup B))$,

Rules of inference (r1), (r2) are:

$$(r1)\ \frac{A\ ;B}{(A \cup \neg B)}, \qquad (r2)\ \frac{A\ ;(A \cup B)}{B}.$$

1. *Verify whether S is sound/not sound under classical semantics.*
2. *Find a formal proof of* $\neg(A \Rightarrow (A \cup B))$ *in S, ie. show that*

$$\vdash_S \neg(A \Rightarrow (A \cup B)).$$

3. *Does* $\vdash_S \neg(A \Rightarrow (A \cup B))$ *prove that* $\models \neg(A \Rightarrow (A \cup B))$*?*

Solution
Part 1. The system is not sound. Take any v such that $v^*(A) = T$ and $v^*(B) = F$. The premiss $(A \cup B$ of the rule $(r2)$ is T under v, and the conclusion under v is $v^*(B) = F$.

Part 2. The proof of $\neg(A \Rightarrow (A \cup B))$ is as follows.
B_1: $(A \Rightarrow (A \cup B))$,
axiom
B_2: $(A \Rightarrow (A \cup B))$,
axiom

B_3: $((A \Rightarrow (A \cup B)) \cup \neg(A \Rightarrow (A \cup B)))$,
rule (r1) application to B_1 and B_2
B_4: $\neg(A \Rightarrow (A \cup B))$,
rule (r2) application to B_1 and B_3).

Part 3. System S is not sound, so existence of a proof does not guarantee that what we proved is a tautology. Moreover, the proof of $\neg(A \Rightarrow (A \cup B))$ used rule $(r2)$ that is not sound.

Exercise 4.8

Create a 3 valued extensional semantics $\mathbf{M}$ *for the language* $\mathcal{L}_{\{\neg,\ \mathbf{L},\ \cup,\ \Rightarrow\}}$ *by defining the connectives* $\neg$, $\cup$, $\Rightarrow$ *on a set* $\{F, \bot, T\}$ *of logical values. You must follow the following* **assumptions a1, a2**.
a1 *The third logical value value is intermediate between truth and falsity, i.e. the set of logical values is ordered as follows:* $F < \bot < T$.
a2 *The value* T *is the designated value. The semantics has to model a situation in which one "likes" only truth; i.e. the connective* $\mathbf{L}$ *must be such that* $\mathbf{L}T = T$, $\mathbf{L}\bot = F$, *and* $\mathbf{L}F = F$. *The connectives* $\neg$, $\cup$, $\Rightarrow$ *can be defined as you wish, but you have to define them in such a way to make sure that*

$$\models_{\mathbf{M}} (\mathbf{L}A \cup \neg\mathbf{L}A).$$

Solution
Here is a simple **M** semantics. We define the logical connectives by writing functions defining connectives in form of the truth tables.

M Semantics

L	F	$\bot$	T
	F	F	T

$\neg$	F	$\bot$	T
	T	F	F

$\cap$	F	$\bot$	T
F	F	F	F
$\bot$	F	$\bot$	$\bot$
T	F	$\bot$	T

$\cup$	F	$\bot$	T
F	F	$\bot$	T
$\bot$	$\bot$	T	T
T	T	T	T

$\Rightarrow$	F	$\bot$	T
F	T	T	T
$\bot$	T	$\bot$	T
T	F	F	T

We verify whether the condition **s3** is satisfied, i.e. whether $\models_{\mathbf{LK}} (LA \cup \neg LA)$ by simple evaluation. Let $v : VAR \longrightarrow \{F, \bot, T\}$ be any truth assignment. For any formula A, $v^*(A) \in \{F, \bot, T\}$ and $LF \cup \neg LF = LF \cup \neg LF = F \cup \neg F \cup T = T$, $L\bot \cup \neg L\bot = F \cup \neg F = F \cup T = T$, $LT \cup \neg LT = T \cup \neg T = F \cup T = T$.

We verify whether $\models_{\mathbf{M}} (LA \cup \neg LA)$ by simple evaluation. Let $v : VAR \longrightarrow \{F, \bot, T\}$ be any truth assignment. For any formula A, $v^*(A) \in \{F, \bot, T\}$ and $LF \cup \neg LF = LF \cup \neg LF = F \cup \neg F \cup T = T$, $L\bot \cup \neg L\bot = F \cup \neg F = F \cup T = T$, $LT \cup \neg LT = T \cup \neg T = F \cup T = T$.

Exercise 4.9

Let S be the following proof system

$$S = (\ \mathcal{L}_{\{\neg,\mathbf{L},\cup,\Rightarrow\}},\ \mathcal{F},\ \ \{A1, A2\},\ \ \{(r1),\ (r2)\}\ )$$

where the logical axioms A1, A2 and rules of inference (r1), (r2) defined for any formulas $A, B \in \mathcal{F}$ as follows.

A1 $(\mathbf{L}A \cup \neg\mathbf{L}A)$,
A2 $(A \Rightarrow \mathbf{L}A)$,

$$(r1)\ \frac{A\ ;B}{(A\cup B)},\qquad\qquad (r2)\ \frac{A}{\mathbf{L}(A\Rightarrow B)}.$$

1. Show, by constructing a proper formal proof that

$$\vdash_S\ ((\mathbf{L}b \cup \neg\mathbf{L}b) \cup \mathbf{L}((\mathbf{L}a \cup \neg\mathbf{L}a) \Rightarrow b))).$$

Please, write comments how each step of the proof was obtained

2. Verify whether the system S is **M***-sound funder the semantics* **M** *you have developed in Exercise 4.8. You can use shorthand notation.*

3. If the system S **is not sound/ sound** *under your semantics* **M** *then re-define the connectives in a way that such obtained new semantics* **N** *would make S* **sound/not sound**

Solution
Part 1. Here is the formal proof $B_1, B_2,\ B_3,\ B_4$.

B_1: $(\mathbf{L}a \cup \neg\mathbf{L}a)$, axiom $A1$ for $A = a$,
B_2: $\mathbf{L}((\mathbf{L}a \cup \neg\mathbf{L}a) \Rightarrow b)$, rule (r2) for $B = b$ applied to B_1,
B_3: $(\mathbf{L}b \cup \neg\mathbf{L}Ab)$, axiom $A1$ for A= b,
B_4: $((\mathbf{L}b \cup \neg\mathbf{L}b) \cup \mathbf{L}((\mathbf{L}a \cup \neg\mathbf{L}a) \Rightarrow b))$, rule (r1) applied to B_3 and B_2.

Part 2. Observe that both logical axioms of S are **M** tautologies. A1 is **M** tautology by definition of the semantics, A2 is **M** tautology by direct evaluation. Rule (r1) is **sound** because when $A = T$ and $B = T$ we get $A \cup B = T \cup T = T$. Rule (r2) is **not sound** because when $A = T$ and $B = F$ (or $B = \perp$) we get $\mathbf{L}(A \Rightarrow B) = \mathbf{L}(T \Rightarrow F) = \mathbf{L}F = F$ (or $\mathbf{L}(T \Rightarrow \perp) = \mathbf{L}\perp = F$).

Part 3. In order to make the rule $(r2)$ sound while preserving the soundness of axioms A1, A2 we have to modify only the definition of implication. Here is the **N** semantics implication

$\Rightarrow$	F	$\perp$	T
F	T	T	T
$\perp$	T	$\perp$	T
T	T	T	T

Observe that it would be hard to convince anybody to use our sound proof system S, as it would be hard to convince anybody to adopt our **N** semantics.

4.4 Homework Problems

1. Given a proof system $S = (\mathcal{L}_{\{\cup,\Rightarrow\}},\ \mathcal{F},\ LA = \{A1, A2\},\ (r)\ \frac{(A\Rightarrow B)}{(A\Rightarrow(A\Rightarrow B))}\ )$, where $A1 = (A \Rightarrow (A \cup B)),\ A2 = (A \Rightarrow (B \Rightarrow A))$. Prove, by constructing a formal proof in S that $\vdash_S (A \Rightarrow (A \Rightarrow (A \Rightarrow (A \Rightarrow A))))$.

   Does it prove that $\models (A \Rightarrow (A \Rightarrow (A \Rightarrow (A \Rightarrow A))))$.

2. Given a proof system: $S = (\mathcal{L}_{\{\neg,\Rightarrow\}},\quad \mathcal{E} = \mathcal{F}\quad LA = \{(A \Rightarrow A), (A \Rightarrow (\neg A \Rightarrow B))\},\quad (r)\ \frac{(A\Rightarrow B)}{(B\Rightarrow(A\Rightarrow B))})$.

   (i) Prove that S is sound under classical semantics.

   (ii) Prove that S is not sound under **K** semantics.

   (iii) Write a formal proof in S with 3 applications of the rule (r).

   (iv) Prove, by constructing a formal proof that $\vdash_S ((\neg A \Rightarrow B) \Rightarrow (A \Rightarrow (\neg A \Rightarrow B)))$.

3. Given a proof system: $S = (\mathcal{L}_{\{\cup,\Rightarrow\}},\quad \mathcal{E} = \mathcal{F}\quad LA = \{A1, A2\},\quad \mathcal{R} = \{(r)\}\ )$, where $A1 = (A \Rightarrow (A \cup B)),\ A2 = (A \Rightarrow (B \Rightarrow A))$ and $(r)\ \frac{(A\Rightarrow B)}{(A\Rightarrow(A\Rightarrow B))}$.

   (i) Prove that S is *sound* under classical semantics.

   (ii) Determine whether S is *sound* or *not sound* under **L** semantics.

   (iii) Write a formal proof in S with 3 applications of the rule (r) that starts with axiom A1.

   (iv) Does it prove/ that $\models_{\mathbf{L}} A$ for a formula A obtained in (iii).

   (v) Prove, by constructing a formal proof in S that $\vdash_S (A \Rightarrow (A \Rightarrow (A \Rightarrow (A \Rightarrow A))))$. Does it prove (and why) that $\models (A \Rightarrow (A \Rightarrow (A \Rightarrow (A \Rightarrow A))))$.

4. S is the following proof system: $S = (\ \mathcal{L}_{\{\Rightarrow,\cup,\neg\}},\quad \mathcal{F},\quad LA = \{(A \Rightarrow (A \cup B))\}\quad (r1),\ (r2)\ )$, where $(r1)\ \frac{A\ ;B}{(A\cup\neg B)}$, $(r2)\ \frac{A\ ;(A\cup B)}{B}$.

   (i) Verify whether S is sound/not sound under classical semantics. (ii) Find a formal proof of $\neg(A \Rightarrow (A \cup B))$ in S, i. e. show that $\vdash_S \neg(A \Rightarrow (A \cup B))$.

   (iii) Does above (ii) prove that $\models \neg(A \Rightarrow (A \cup B))$?

5. By a m-valued semantics S_m, for a propositional language $\mathcal{L} = \mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow\}}$ we understand any definition of of connectives $\neg, \cap, \cup, \Rightarrow$ as operations on a set $L_m = \{l_1, l_2, ...l_m\}$ of logical values (for $m \geq 2$).

We assume that $l_1 \leq l_2 \leq ... \leq l_m$, i.e. L_m is totally ordered by a certain relation $\leq$ with l_1, l_m being smallest and greatest elements, respectively. We denote $l_1 = F$, $l_m = T$ and call them (total) False and Truth, respectively. For example, when $m = 2$, $L_2 = \{F, T\}$, $F \leq T$. Semantics S_2 is called a classical semantics if the connectives are defined as $x \cup y = max\{x, y\}$, $x \cap y = min\{x, y\}$, $\neg T = F, \neg F = T$, and $x \Rightarrow y = \neg x \cup y$, for any $x, y \in L_2$.

Let VAR be a set of propositional variables of $\mathcal{L}$ and let S_m be any m-valued semantics for $\mathcal{L}$. A truth assignment $v : VAR \longrightarrow L_m$ is called a S_m model for a formula A of $\mathcal{L}$ if and only if $v^*(A) = T$ and logical value $v^*(A)$ is evaluated accordingly to the semantics S_m. We denote is symbolically as $v \models_{S_m} A$.

Let $S = (\mathcal{L},\ \mathcal{F},\ \{A1, A2, A3\},\ \ MP\ \frac{A\ ;(A \Rightarrow B)}{B})$ be a proof system with logical axioms:

A1: $(A \Rightarrow (B \Rightarrow A))$,

A2: $((A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C)))$,

A3: $((\neg B \Rightarrow \neg A) \Rightarrow ((\neg B \Rightarrow A) \Rightarrow B))$.

(i) Prove that S is sound under S_2 classical semantics.

(ii) Define your own S_2 semantics under which S is not sound.

(iii) Define your own S_4 semantics under which S is sound and other S_4 semantics under which S is not sound.

(iv) Define your own S_n semantics such that S is sound for all for $2 \leq n \leq m$.

(v) Show, by construction a formal proof, that $\vdash_S (A \Rightarrow A)$.

Chapter 5

Hilbert Proof Systems Completeness of Classical Propositional Logic

The Hilbert proof systems are systems based on a language with implication and contain a Modus Ponens rule as a rule of inference. They are usually called Hilbert style formalizations. We will call them here Hilbert style proof systems, or Hilbert systems, for short.

Modus Ponens is probably the oldest of all known rules of inference as it was already known to the Stoics (3rd century B.C.). It is also considered as the most natural to our intuitive thinking and the proof systems containing it as the inference rule play a special role in logic. The Hilbert proof systems put major emphasis on logical axioms, keeping the rules of inference to minimum, often in propositional case, admitting only Modus Ponens, as the sole inference rule.

There are many proof systems that describe classical propositional logic, i.e. that are complete proof systems with the respect to the classical semantics.

We present here, after Elliott Mendelson's book *Introduction to Mathematical Logic* (1987), a Hilbert proof system for the classical propositional logic and discuss two ways of proving the Completeness Theorem for it.

Any proof of the Completeness Theorem consists always of two parts. First we have show that *all formulas that have a proof are tautologies.* This implication is also called a Soundness Theorem, or soundness part of the Completeness Theorem. The second implication says: *if a formula is a tautology then it has a proof.* This alone is sometimes called a Completeness Theorem (on assumption that the system is sound). Traditionally it is called a completeness part of the

Completeness Theorem.

The proof of the soundness part is standard. We concentrate here on the completeness part of the Completeness Theorem and present two proofs of it.

The first proof is based on the one presented in the Mendelson's book *Introduction to Mathematical Logic* (1987). It is is a straightforward constrictive proof that shows how one can use the assumption that a formula A is a tautology in order to construct its formal proof. It is hence called *a proof - construction method*. It is a beautiful proof

The second proof is non-constrictive. Its strength and importance lies in a fact that the methods it uses can be applied to the proof of completeness for classical predicate logic. We will discuss and apply them in Chapter 9.

It proves the completeness part of the Completeness Theorem by proving the converse implication to it. It shows how one can deduce that *a formula A is not a tautology from the fact that it does not have a proof.* It is hence called a *counter-model construction proof.*

Both proofs of the Completeness Theorem relay on the Deduction Theorem and so it is the first theorem we are going to prove.

5.1 Deduction Theorem

We consider first a very simple Hilbert proof system based on a language with implication as the only connective, with two logical axioms (axiom schemas) which characterize the implication, and with Modus Ponens as a sole rule of inference. We call it a Hilbert system H_1 and define it as follows.

$$H_1 = (\ \mathcal{L}_{\{\Rightarrow\}},\ \ \mathcal{F},\ \ A1, A2,\ \ (MP)\ ), \tag{5.1}$$

where

A1 $(A \Rightarrow (B \Rightarrow A))$,

A2 $((A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C)))$,

(MP) is the following rule of inference, called Modus Ponens

$$(MP)\ \frac{A\ ;\ (A \Rightarrow B)}{B},$$

and $A, B, C \in \mathcal{F}$ are any formulas of the propositional language $\mathcal{L}_{\{\Rightarrow\}}$.

Finding formal proofs in this system requires some ingenuity. Let's construct, as an example, the formal proof of such a simple formula as $A \Rightarrow A$.

The formal proof of $(A \Rightarrow A)$ in H_1 is a sequence

$$B_1,\ B_2,\ B_3,\ B_4, B_5 \tag{5.2}$$

as defined below.

$B_1 = ((A \Rightarrow ((A \Rightarrow A) \Rightarrow A)) \Rightarrow ((A \Rightarrow (A \Rightarrow A)) \Rightarrow (A \Rightarrow A)))$,
axiom A2 for $A = A$, $B = (A \Rightarrow A)$, and $C = A$

$B_2 = (A \Rightarrow ((A \Rightarrow A) \Rightarrow A))$,
axiom A1 for $A = A$, $B = (A \Rightarrow A)$

$B_3 = ((A \Rightarrow (A \Rightarrow A)) \Rightarrow (A \Rightarrow A)))$,
MP application to B_1 and B_2

$B_4 = (A \Rightarrow (A \Rightarrow A))$,
axiom A1 for $A = A$, $B = A$

$B_5 = (A \Rightarrow A)$
MP application to B_3 and B_4

We have hence proved the following.

Fact 5.1

For any $A \in \mathcal{F}$,

$$\vdash_{H_1} (A \Rightarrow A)$$

and the sequence 5.2 constitutes its formal proof.

It is easy to see that the above proof wasn't constructed automatically. The main step in its construction was the choice of a proper form (substitution) of logical axioms to start with, and to continue the proof with. This choice is far from obvious for un-experienced prover and impossible for a machine, as the number of possible substitutions is infinite.

Observe that the systems $S_1 - S_4$ from the previous Chapter 4 had inference rules such that it was possible to "reverse" their use; to use them in the reverse manner in order to search for proofs, and we were able to do so in a blind, fully automatic way. We were able to conduct an argument of the type: *if this formula has a proof the only way to construct it is from such and such formulas by the means of one of the inference rules, and that formula can be found automatically.* We called proof systems with such property syntactically decidable and defined them formally as follows.

Definition 5.1 (Syntactic Decidability)

A proof system $S = (\mathcal{L}, \mathcal{E}, LA, \mathcal{R})$ for which there is an effective mechanical, procedure that finds (generates) a formal proof of any E in S, if it exists, is *called* **syntactically semi- decidable**. *If additionally there is an effective method of deciding that if a proof of E in S not found, it does not exist, the system S is called* **syntactically decidable**. *Otherwise S is* **syntactically undecidable**.

We will argue now, that one can't apply the above argument to the proof search in Hilbert proof systems as they which contain Modus Ponens as an inference rule.

A *general procedure* for searching for proofs in a proof system S can be stated is as follows. Given an expression B of the system S. If it has a proof, it must be conclusion of the inference rule. Let's say it is a rule r. We find its premisses, with B being the conclusion, i.e. we evaluate $r^{-1}(B)$. If all premisses are axioms, the proof is found. Otherwise we repeat the procedure for any non-axiom premiss.

Search for proof in Hilbert Systems must involve the Modus Ponens. The rule says: given two formulas A and $(A \Rightarrow B)$ we can conclude a formula B. Assume now that we have a formula B and want to find its proof. If it is an axiom, we have the proof: the formula itself. If it is not an axiom, it had to be obtained by the application of the Modus Ponens rule, to certain two formulas A and $(A \Rightarrow B)$. But there is infinitely many of formulas A and $(A \Rightarrow B)$. I.e. for any B, the inverse image of B under the rule MP, $MP^{-1}(B)$ is countably infinit Obviously, we have the following.

Fact 5.2

Any Hilbert proof system is not syntactically decidable, in particular, the system H_1 is not syntactically decidable.

Semantic Link 1 System H_1 is obviously sound under classical semantics and is sound under **Ł**, **H** semantics and not sound under **K** semantics.

We leave the proof of the following theorem (by induction with respect of the length of the formal proof) as an easy exercise to the reader.

Theorem 5.1 (Soundness of H_1)

For any $A \in \mathcal{F}$ of H_1, if $\vdash_{H_1} A$, then $\models A$.

Semantic Link 2 System H_1 **is not complete** under classical semantics. It means that not all classical tautologies have a proof in H_1. We have proved in Chapter 3 that one needs negation and one of the other connectives $\cup, \cap, \Rightarrow$ to express all classical connectives, and hence all classical tautologies. Our language contains only implication and one can't express negation in terms of implication and hence we can't provide a proof of any tautology i.e. its logically equivalent form in our language. It means we have proved the following.

Fact 5.3

The proof system H_1 is sound, but **not complete** *under the classical semantics.*

We have constructed a formal proof (5.2) of $(A \Rightarrow A)$ in H_1 on a base of logical axioms, as an example of complexity of finding proofs in Hilbert systems.

In order to make the construction of formal proofs easier by the use of previously proved formulas we use the notions of a formal proof from some *hypotheses* (and logical axioms) in any proof system $S = (\mathcal{L},\ \mathcal{E},\ LA, \mathcal{R})$ defined as follows in chapter 4.

Definition 5.2 (Proof from Hypotheses)

Given a proof system $S = (\mathcal{L},\ \mathcal{E}, LA, \mathcal{R})$ and let Γ be any set of expressions of S, i.e. let $\Gamma \subseteq \mathcal{E}$.

A proof of an expression $E \in$ from the set Γ of expressions is a sequence

$$E_1, E_2,\ \dots\ E_n$$

of expressions, such that

$$E_1 \in LA \cup \Gamma, \qquad E_n = E$$

and for each i, $1 < i \leq n$, either $E_i \in LA \cup \Gamma$ or E_i is a direct consequence of some of the preceding expressions in the sequence $E_1, E_2,\ \dots\ E_n$ by virtue of one of the rules of inference from $\mathcal{R}$.

We write

$$\Gamma \vdash_S E$$

to denote that the expression E has a proof (is provable) from Γ in S and we write $\Gamma\ \vdash\ E$, when the system S is fixed.

When the set of hypothesis Γ is a *finite set* and $\Gamma = \{B_1, B_2, ..., B_n\}$, then we write

$$B_1, B_2, ..., B_n \vdash_S E$$

instead of $\{B_1, B_2, ..., B_n\} \vdash_S E$. The case when Γ is an empty set i.e. when $\Gamma = \emptyset$ is a special one. By the definition of a proof of E from Γ, $\emptyset \vdash_S E$ means that in the proof of E only logical axioms LA of S were used. We hence write as we did before

$$\vdash_S E$$

to denote that E has a proof from the empty set Γ.

Definition 5.3 (Consequence in S)

Given a proof system $S = (\mathcal{L},\ \mathcal{F}, LA, \mathcal{R})$ and a set $\Gamma \subseteq \mathcal{F}$. Any formula $A \in \mathcal{F}$ **provable** *from Γ, i.e. such that*

$$\Gamma \vdash_S A$$

is called a **consequence** *of Γ in S. Formulas from Γ are called* **hypotheses** *or premisses of a proof of A from Γ in S.*

The following are simple, but very important properties of the notion of consequence.

Fact 5.4 (Consequence Properties)

Given a proof system $S = (\mathcal{L},\ \mathcal{F}, LA, \mathcal{R})$. For any sets $\Gamma, \Delta \subseteq \mathcal{F}$ the following holds.

1. *If $\Gamma \subseteq \Delta$ and $\Gamma \vdash_S A$, then $\Delta \vdash_S A$.* **monotonicity**
2. *$\Gamma \vdash_S A$ if and only if there is a* **finite** *subset Γ_0 of Γ such that $\Gamma_0 \vdash_S A$.* **finiteness**
3. *If $\Delta \vdash_S A$, and, for each $B \in \Delta$, $\Gamma \vdash_S B$, then $\Gamma \vdash_S A$.* **transitivity**

Proof
The properties follow directly from the definition 5.2 and their proofs are left to the reader as an exercise.

The **monotonicity** property represents the fact that if a formula A is provable from a set Γ of premisses (hypotheses), then if we add still more premisses, A is still provable. It hence is often stated as follows,

$$\text{If } \Gamma \vdash_S A, \text{ then } \Gamma \cup \Delta \vdash_S A, \text{ for any set } \Delta \subseteq \mathcal{F}. \tag{5.3}$$

The detailed investigation of Tarski general notion of consequence operation, its relationship with proof systems, and hence with the consequence in S introduced here is included in Chapter 4. Here is an application of the proof from hypotheses definition 5.2 to the system H_1.

Exercise 5.1

Construct a proof in H_1 of a formula $(A \Rightarrow C)$ from the set of hypotheses $\Gamma = \{(A \Rightarrow B), (B \Rightarrow C)\}$. I.e. show that

$$(A \Rightarrow B), (B \Rightarrow C) \vdash_{H_1} (A \Rightarrow C).$$

Solution
The required formal proof is a sequence

$$B_1, B_2, ...B_7 \tag{5.4}$$

such that

$B_1 = (B \Rightarrow C)$,
hypothesis

$B_2 = (A \Rightarrow B)$,
hypothesis

$B_3 = ((B \Rightarrow C) \Rightarrow (A \Rightarrow (B \Rightarrow C)))$,
axiom A1 for $A = (B \Rightarrow C),\ B = A$

$B_4 = (A \Rightarrow (B \Rightarrow C))$
$B_1,\ B_3$ and MP

$B_5 = ((A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C)))$,
axiom A2

$B_6 = ((A \Rightarrow B) \Rightarrow (A \Rightarrow C))$,
B_5 and B_4 and MP

$B_7 = (A \Rightarrow C)$.
B_2 and B_6 and MP

Exercise 5.2

Show, by constructing a formal proof that $A \vdash_{H_1} (A \Rightarrow A)$.

Solution
The required formal proof is a sequence

$$B_1, B_2, B_3 \tag{5.5}$$

such that

$B_1 = A$,
hypothesis

$B_2 = (A \Rightarrow (A \Rightarrow A))$,
axiom A1 for $B = A$,

$B_3 = (A \Rightarrow A)$
$B_1,\ B_2$ and MP.

We can further simplify the task of constructing formal proofs in H_1 by the use of the following Deduction Theorem.

In mathematical arguments, one often assumes a statement A on the assumption (hypothesis) of some other statement B and then concludes that we have proved the implication "if A, then B". This reasoning is justified by the following theorem, called a Deduction Theorem. It was first formulated and proved for a certain Hilbert proof system S for the classical propositional logic by Herbrand in 1930 in a form stated below.

Theorem 5.2 (Deduction Theorem for S) *(Herbrand,1930)*

For any formulas A, B of the language of S,

$$\text{if} \quad A \vdash_S B, \quad \text{then} \ \vdash_S (A \Rightarrow B).$$

We are going to prove now that for our system H_1 is strong enough to prove the Herbrand Deduction Theorem for it. In fact we formulate and prove a more general version of the Theorem 5.2.

To formulate it we introduce the following notation. We write $\Gamma, A \vdash_S B$ for $\Gamma \cup \{A\} \vdash_S B$, and in general we write $\Gamma, A_1, A_2, ..., A_n \vdash_S B$ for $\Gamma \cup \{A_1, A_2, ..., A_n\} \vdash_S B$. We are now going to prove the following.

Theorem 5.3 (Deduction Theorem for H_1)

For any subset Γ of the set of formulas $\mathcal{F}$ of H_1 and for any formulas $A, B \in \mathcal{F}$,

$$\Gamma, \ A \vdash_{H_1} B \ \ if\ and\ only\ if \ \ \Gamma \vdash_{H_1} (A \Rightarrow B).$$

In particular,

$$A \vdash_{H_1} B \ \ if\ and\ only\ if \ \ \vdash_{H_1} (A \Rightarrow B).$$

Proof

We use we use the symbol $\vdash$ instead of $\vdash_{H_1}$. for simplicity.

Part 1

We first prove the "if" part:

$$\text{If} \quad \Gamma, \ A \vdash B \qquad \text{then} \qquad \Gamma \vdash (A \Rightarrow B).$$

Assume that $\Gamma, \ A \ \vdash B$, i.e. that we have a formal proof

$$B_1, B_2, ..., B_n \tag{5.6}$$

of B from the set of formulas $\Gamma \cup \{A\}$. In order to prove that $\Gamma \vdash (A \Rightarrow B)$ we will prove the following a little bit stronger statement **S**.

$$\mathbf{S}: \quad \Gamma \vdash (A \Rightarrow B_i) \text{ for all } B_i \ (1 \leq i \leq n) \ \text{ in the proof (5.6) of } B.$$

Hence, in particular case, when $i = n$, we will obtain that also

$$\Gamma \vdash (A \Rightarrow B).$$

The proof of **S** is conducted by induction on i ($1 \leq i \leq n$).

Base Step $i = 1$.
When $i = 1$, it means that the formal proof (5.6) contains only one element B_1. By the definition of the formal proof from $\Gamma \cup \{A\}$, we have that $B_1 \in LA$, or $B_1 \in \Gamma$, or $B_1 = A$, i.e.

$$B_1 \in \{A1, A2\} \cup \Gamma \cup \{A\}.$$

Here we have two cases.

Case 1. $B_1 \in \{A1, A2\} \cup \Gamma$.
Observe that $(B_1 \Rightarrow (A \Rightarrow B_1))$ is the axiom $A1$ and by assumption $B_1 \in \{A1, A2\} \cup \Gamma$, hence we get the required proof of $(A \Rightarrow B_1)$ from Γ by the following application of the Modus Ponens rule

$$(MP)\ \frac{B_1\ ;\ (B_1 \Rightarrow (A \Rightarrow B_1))}{(A \Rightarrow B_1)}.$$

Case 2. $B_1 = A$.
When $B_1 = A$, then to prove $\Gamma \vdash (A \Rightarrow B)$ means to prove $\Gamma \vdash (A \Rightarrow A)$. This holds by the monotonicity of the consequence in H_1 (Fact 5.4), and the fact that we have proved (Fact 5.1) that $\vdash (A \Rightarrow A)$. The above cases conclude the proof of the Base case $i = 1$.

Inductive step
Assume that $\Gamma \vdash (A \Rightarrow B_k)$ for all $k < i$, we will show that using this fact we can conclude that also $\Gamma \vdash (A \Rightarrow B_i)$.

Consider a formula B_i in the sequence 5.6. By the definition, $B_i \in \{A1, A2\} \cup \Gamma \cup \{A\}$ or B_i follows by MP from certain B_j, B_m such that $j < m < i$. We have to consider again two cases.

Case 1. $B_i \in \{A1, A2\} \cup \Gamma \cup \{A\}$.
The proof of $(A \Rightarrow B_i)$ from Γ in this case is obtained from the proof of the Base Step for $i = 1$ by replacement B_1 by B_i and will be omitted here as a straightforward repetition.

Case 2. B_i is a conclusion of MP.
If B_i is a conclusion of MP, then we must have two formulas B_j, B_m in the sequence 5.6 such that $j < i, m < i,\ j \neq m$ and

$$(MP)\ \frac{B_j\ ;\ B_m}{B_i}.$$

By the inductive assumption, the formulas B_j, B_m are such that

$$\Gamma \vdash (A \Rightarrow B_j) \tag{5.7}$$

and

$$\Gamma \vdash (A \Rightarrow B_m). \tag{5.8}$$

Moreover, by the definition of the Modus Ponens rule, the formula B_m has to have a form $(B_j \Rightarrow B_i)$, i.e. $B_m = (B_j \Rightarrow B_i)$, and the the inductive assumption (5.8) can be re-written as follows.

$$\Gamma \vdash (A \Rightarrow (B_j \Rightarrow B_i)), \quad for\ j < i. \tag{5.9}$$

Observe now that the formula

$$((A \Rightarrow (B_j \Rightarrow B_i)) \Rightarrow ((A \Rightarrow B_j) \Rightarrow (A \Rightarrow B_i)))$$

is a substitution of the axiom schema A2 and hence has a proof in our system. By the monotonicity of the consequence (5.3), it also has a proof from the set Γ, i.e.

$$\Gamma \vdash ((A \Rightarrow (B_j \Rightarrow B_i)) \Rightarrow ((A \Rightarrow B_j) \Rightarrow (A \Rightarrow B_i))). \tag{5.10}$$

Applying the rule MP to formulas (5.10) and (5.9,) i.e. performing the following

$$(MP)\ \frac{(A \Rightarrow (B_j \Rightarrow B_i))\ ;\ ((A \Rightarrow (B_j \Rightarrow B_i)) \Rightarrow ((A \Rightarrow B_j) \Rightarrow (A \Rightarrow B_i)))}{((A \Rightarrow B_j) \Rightarrow (A \Rightarrow B_i))}$$

we get that also

$$\Gamma \vdash ((A \Rightarrow B_j) \Rightarrow (A \Rightarrow B_i)). \tag{5.11}$$

Applying again the rule MP to formulas 5.7 and 5.11, i.e. performing the following

$$(MP)\ \frac{(A \Rightarrow B_j)\ ;\ ((A \Rightarrow B_j) \Rightarrow (A \Rightarrow B_i))}{(A \Rightarrow B_i)}$$

we get that

$$\Gamma \vdash (A \Rightarrow B_i)$$

what ends the proof of the Inductive Step. By the mathematical induction principle, we hence have proved that $\Gamma \vdash (A \Rightarrow B_j)$ for all i such that $1 \leq i \leq n$. In particular it is true for $i = n$, what means for $B_n = B$ and we have proved that

$$\Gamma \vdash (A \Rightarrow B).$$

This ends the proof of the **Part 1**.

Part 2

The proof of the inverse implication

$$\text{if } \Gamma \vdash (A \Rightarrow B) \text{ then } \Gamma, A \vdash B$$

is straightforward. Assume that $\Gamma \vdash (A \Rightarrow B)$, hence by the monotonicity of the consequence (5.3) we have also that $\Gamma, A \vdash (A \Rightarrow B)$. Obviously, $\Gamma, A \vdash A$. Applying Modus Ponens to the above, we get the proof of B from $\{\Gamma, A\}$ i.e. we have proved that $\Gamma,\ A \vdash B$. That ends the proof of the deduction theorem for any set $\Gamma \subseteq \mathcal{F}$ and any formulas $A, B \in \mathcal{F}$. The particular case is obtained from the above by assuming that the set Γ is empty. This ends the proof of the Deduction Theorem for H_1.

The proof of the following useful lemma provides a good example of multiple applications of the Deduction Theorem 5.3.

Lemma 5.1

For any $A, B, C \in \mathcal{F}$,

(a) $(A \Rightarrow B), (B \Rightarrow C) \vdash_{H_1} (A \Rightarrow C)$,

(b) $(A \Rightarrow (B \Rightarrow C)) \vdash_{H_1} (B \Rightarrow (A \Rightarrow C))$.

Proof of **(a)**.
Deduction theorem says:
$(A \Rightarrow B), (B \Rightarrow C) \vdash_{H_1} (A \Rightarrow C)$ if and only if $(A \Rightarrow B), (B \Rightarrow C), A \vdash_{H_1} C$.

We construct a formal proof

$$B_1,\ B_2,\ B_3,\ B_4,\ B_5$$

of $(A \Rightarrow B), (B \Rightarrow C), A \vdash_{H_1} C$ as follows.

$B_1 = (A \Rightarrow B)$,
hypothesis

$B_2 = (B \Rightarrow C)$,
hypothesis

$B_3 = A$,
hypothesis

$B_4 = B$,
$B_1,\ B_3$ and MP

$B_5 = C$.
$B_2,\ B_4$ and MP

Thus $(A \Rightarrow B), (B \Rightarrow C) \vdash_{H_1} (A \Rightarrow C)$ by Deduction Theorem.

Proof of **(b)**.

By Deduction Theorem,

$(A \Rightarrow (B \Rightarrow C)) \vdash_{H_1} (B \Rightarrow (A \Rightarrow C))$ if and only if $(A \Rightarrow (B \Rightarrow C)), B \vdash_{H_1} (A \Rightarrow C)$.

We construct a formal proof

$$B_1,\ B_2,\ B_3,\ B_4,\ B_5,\ B_6,\ B_7$$

of $(A \Rightarrow (B \Rightarrow C)), B \vdash_{H_1} (A \Rightarrow C)$. as follows.

$B_1 = (A \Rightarrow (B \Rightarrow C))$,
hypothesis

$B_2 = B$,
hypothesis

$B_3 = ((B \Rightarrow (A \Rightarrow B))$,
$A1$ for $A = B,\ B = A$

$B_4 = (A \Rightarrow B)$,
$B_2,\ B_3$ and MP

$B_5 = ((A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C)))$,
$axiom A2$

$B_6 = ((A \Rightarrow B) \Rightarrow (A \Rightarrow C))$,
$B_1,\ B_5$ and MP

$B_7 = (A \Rightarrow C)$.

Thus $(A \Rightarrow (B \Rightarrow C)) \vdash_{H_1} (B \Rightarrow (A \Rightarrow C))$ by Deduction Theorem.

Hilbert System H_2

The proof system H_1 is sound and strong enough to admit the Deduction Theorem, but is *not complete*as proved in Fact 5.3. We define now a proof system H_2 that is complete with respect to classical semantics. The proof of Completeness Theorem for H_2 is to be presented in the next section.

H_2 is defined as follows.

$$H_2 = (\ \mathcal{L}_{\{\neg,\ \Rightarrow\}},\ \mathcal{F},\ A1, A2, A3,\ \ MP\ \frac{A\ ;\ (A \Rightarrow B)}{B}), \tag{5.12}$$

where for any formulas $A, B, C \in \mathcal{F}$ of $\mathcal{L}_{\{\neg,\ \Rightarrow\}}$ we define

A1 $(A \Rightarrow (B \Rightarrow A))$,

A2 $((A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C)))$,

A3 $((\neg B \Rightarrow \neg A) \Rightarrow ((\neg B \Rightarrow A) \Rightarrow B)))$,

Observation 5.1 *Here are some simple, straightforward facts about the proof system H_2.*

1. The language of H_2 is obtained from the language of H_1 by adding the connective $\neg$ to it.

2. H_2 is obtained from H_1 by adding axiom to it the axiom A_3 that characterizes negation.

3. The use of axioms $A1, A2$ in the proof of Deduction Theorem 5.3 for H_1 is independent of the negation connective $\neg$ added to the language of H_1.

4. The proof of Deduction Theorem 5.3 for the system H_1 can be repeated as it is for the system H_2.

Directly from the above Observation 5.1 we get the following.

Theorem 5.4 (Deduction Theorem for H_2)

For any subset Γ of the set of formulas $\mathcal{F}$ of H_2 and for any formulas $A, B \in \mathcal{F}$,

$$\Gamma,\ A \vdash_{H_2} B\ \ if\ and\ only\ if\ \ \Gamma \vdash_{H_2} (A \Rightarrow B).$$

In particular,

$$A \vdash_{H_2} B\ \ if\ and\ only\ if\ \ \vdash_{H_2} (A \Rightarrow B).$$

Observe that for the same reason the Lemma 5.1 holds also for H_2. It is a very i useful lemma for creating proofs in H_2 so we re-state it for it here.

Lemma 5.2

For any $A, B, C \in \mathcal{F}$,

(a) $(A \Rightarrow B), (B \Rightarrow C) \vdash_{H_2} (A \Rightarrow C)$,

(b) $(A \Rightarrow (B \Rightarrow C)) \vdash_{H_2} (B \Rightarrow (A \Rightarrow C))$.

We know that the axioms $A1, A2$ are tautologies and the Modus Ponens rule is sound. We get by simple verification that $\models A3$, hence the proof system H_2 is *sound*, and the following holds.

Theorem 5.5 (Soundness Theorem for H_2)

For every formula $A \in \mathcal{F}$, if $\vdash_{H_2} A$, then $\models A$.

The soundness theorem proves that the system "produces" only tautologies. We show, in the next chapter, that our proof system H_2 "produces" not only tautologies, but that all tautologies are provable in it. This is called a *completeness theorem for classical logic.*

Theorem 5.6 (Completeness Theorem for H_2)

For every $A \in \mathcal{F}$,

$$\vdash_{H_2} A \quad \textit{if and only if} \quad \models A.$$

The proof of completeness theorem (for a given semantics) is always a main point in any logic creation. There are many ways (techniques) to prove it, depending on the proof system, and on the semantics we define for it.

We present in the next sections two proofs of the completeness theorem for our system H_2. The proofs use very different techniques, hence the reason of presenting both of them. Both proofs relay heavily on some of the formulas proved in the next section 5.1.1 and stated in Lemma 5.3.

5.1.1 Formal Proofs

We present here some examples of formal proofs in H_2. There are two reasons for presenting them. First reason is that all formulas we prove here to be provable play a crucial role in the proof of Completeness Theorem for H_2, or are needed to find formal proofs of those needed. The second reason is that they provide a "training" ground for a reader to learn how to develop formal proofs. For this second reason we write some proofs in a full detail and we leave some others for the reader to complete in a way explained in the following example.

We write, were needed $\vdash$ instead of $\vdash_{H_2}$.

Example 5.1

We prove that

$$\vdash_{H_2} (\neg\neg B \Rightarrow B) \tag{5.13}$$

by constructing its formal proof $B_1, \ldots, B_5, B_6$ as follows.

$B_1 = ((\neg B \Rightarrow \neg\neg B) \Rightarrow ((\neg B \Rightarrow \neg B) \Rightarrow B))$,

$B_2 = ((\neg B \Rightarrow \neg B) \Rightarrow ((\neg B \Rightarrow \neg\neg B) \Rightarrow B))$,

$B_3 = \neg B \Rightarrow \neg B)$,

$B_4 = ((\neg B \Rightarrow \neg\neg B) \Rightarrow B)$,

$B_5 = \neg\neg B \Rightarrow (\neg B \Rightarrow \neg\neg B))$,

$B_6 = (\neg\neg B \Rightarrow B)$.

Exercise 5.3

Complete the proof $B_1, \ldots, B_5, B_6$ *of (8.3) by providing comments how each step of the proof was obtained.*

Solution
The proof of (8.3) with comments complementing it is as follows.

$B_1 = ((\neg B \Rightarrow \neg\neg B) \Rightarrow ((\neg B \Rightarrow \neg B) \Rightarrow B))$,
axiom $A3$ for $A = \neg B,\ B = B$

$B_2 = ((\neg B \Rightarrow \neg B) \Rightarrow ((\neg B \Rightarrow \neg\neg B) \Rightarrow B))$,
B_1 and Lemma 5.2 **b** for $A = (\neg B \Rightarrow \neg\neg B),\ B = (\neg B \Rightarrow \neg B),\ C = B$.
Lemma 5.2 application is: $((\neg B \Rightarrow \neg\neg B) \Rightarrow ((\neg B \Rightarrow \neg B) \Rightarrow B)) \vdash ((\neg B \Rightarrow \neg B) \Rightarrow ((\neg B \Rightarrow \neg\neg B) \Rightarrow B))$

$B_3 = (\neg B \Rightarrow \neg B)$,
Fact 5.1 for $A = \neg B$

$B_4 = ((\neg B \Rightarrow \neg\neg B) \Rightarrow B)$,
$B_2,\ B_3$ and MP

$B_5 = (\neg\neg B \Rightarrow (\neg B \Rightarrow \neg\neg B))$,
axiom $A1$ for $A = \neg\neg B,\ B = \neg B$

$B_6 = (\neg\neg B \Rightarrow B)$
$B_4,\ B_5$ and Lemma 5.2 **a** for $A = \neg\neg B, B = (\neg B \Rightarrow \neg\neg B), C = B$.
Lemma 5.2 application is:
$(\neg\neg B \Rightarrow (\neg B \Rightarrow \neg\neg B)),\ ((\neg B \Rightarrow \neg\neg B) \Rightarrow B) \vdash (\neg\neg B \Rightarrow B)$

Remark 5.1

Observe that in In step B_2, B_3, B_5, B_6 *of the proof* $B_1, \ldots, B_5, B_6$ *we call previously proved results and use their results as a part of our proof. We can* *insert previously constructed formal proofs of the results we call upon into our formal proof.*

For example we adopt previously constructed proof (5.2) of $(A \Rightarrow A)$ in H_1 to the proof of $(\neg B \Rightarrow \neg B)$ in H_2 by replacing A by $\neg B$ and we insert the proof of $(\neg B \Rightarrow \neg B)$ after B_2.

The "old" step B_3 becomes now B_7, the "old" step B_4 becomes now B_8, etc..... Such "completed" original proof $B_1, \ldots, B_5, B_6$ is now $B_1, \ldots, B_9, B_{10}$ looks now as follows.

$B_1 = ((\neg B \Rightarrow \neg\neg B) \Rightarrow ((\neg B \Rightarrow \neg B) \Rightarrow B))$, (original B_1),
axiom $A3$ for $A = \neg B, B = B$

$B_2 = ((\neg B \Rightarrow \neg B) \Rightarrow ((\neg B \Rightarrow \neg\neg B) \Rightarrow B))$, (original B_2)
B_1 and Lemma 5.2 **b** for $A = (\neg B \Rightarrow \neg\neg B), B = (\neg B \Rightarrow \neg B), C = B$,

$B_3 = ((\neg B \Rightarrow ((\neg B \Rightarrow \neg B) \Rightarrow \neg B)) \Rightarrow ((\neg B \Rightarrow (\neg B \Rightarrow \neg B)) \Rightarrow (\neg B \Rightarrow \neg B)))$, (new proof of B_3 inserted)
axiom A2 for $A = \neg B$, $B = (\neg B \Rightarrow \neg B)$, and $C = \neg B$

$B_4 = (\neg B \Rightarrow ((\neg B \Rightarrow \neg B) \Rightarrow \neg B))$,
axiom A1 for $A = \neg B$, $B = (\neg B \Rightarrow \neg B)$

$B_5 = ((\neg B \Rightarrow (\neg B \Rightarrow \neg B)) \Rightarrow (\neg B \Rightarrow \neg B)))$,
MP application to B_4 and B_3

$B_6 = (\neg B \Rightarrow (\neg B \Rightarrow \neg B))$, (end of proof inserted)
axiom A1 for $A = \neg B$, $B = \neg B$

$B_7 = (\neg B \Rightarrow \neg B)$ ("old" B_3),
MP application to B_5 and B_4

$B_8 = ((\neg B \Rightarrow \neg\neg B) \Rightarrow B)$, ("old" B_4) ("old" B_4)
B_2, B_3 and MP

$B_9 =$ ("old B_5) $(\neg\neg B \Rightarrow (\neg B \Rightarrow \neg\neg B))$, ("old" B_5) Axiom $A1$ for $A = \neg\neg B, B = \neg B$

$B_{10} = (\neg\neg B \Rightarrow B)$. ("old B_6)
B_8, B_9 and Lemma 5.2 **a** for $A = \neg\neg B, B = (\neg B \Rightarrow \neg\neg B), C = B$

We repeat our procedure by replacing the step B_2 by its formal proof as defined in the proof of the Lemma 5.1 **b**, and continue the process for all other steps which involved application of Lemma 5.2 until we get a full **formal proof from the axioms** of H_2 only.

Usually we don't need to do it, but it is important to remember that it always can be done, if we wished to take time and space to do so.

Example 5.2

We prove that

$$\vdash_{H_2} (B \Rightarrow \neg\neg B) \tag{5.14}$$

by constructing its formal proof $B_1, \ \dots, \ B_5$ *as follows.*

$B_1 = ((\neg\neg\neg B \Rightarrow \neg B) \Rightarrow ((\neg\neg\neg B \Rightarrow B) \Rightarrow \neg\neg B))$,

$B_2 = (\neg\neg\neg B \Rightarrow \neg B)$,

$B_3 = ((\neg\neg\neg B \Rightarrow B) \Rightarrow \neg\neg B)$,

$B_4 = (B \Rightarrow (\neg\neg\neg B \Rightarrow B))$,

$B_5 = (B \Rightarrow \neg\neg B)$.

Exercise 5.4

Complete the proof $B_1, \ \dots, \ B_5$ *of (8.17) by providing comments how each step of the proof was obtained.*

Solution
The proof of (8.17) with comments complementing it is as follows.

$B_1 = ((\neg\neg\neg B \Rightarrow \neg B) \Rightarrow ((\neg\neg\neg B \Rightarrow B) \Rightarrow \neg\neg B))$,
axiom $A3$ for $A = B, \ B = \neg\neg B$

$B_2 = (\neg\neg\neg B \Rightarrow \neg B)$,
Example 5.10 for $B = \neg B$

$B_3 = ((\neg\neg\neg B \Rightarrow B) \Rightarrow \neg\neg B)$,
B_1, B_2 and MP, i.e.

$$\frac{(\neg\neg\neg B \Rightarrow \neg B); ((\neg\neg\neg B \Rightarrow \neg B) \Rightarrow ((\neg\neg\neg B \Rightarrow B) \Rightarrow \neg\neg B))}{((\neg\neg\neg B \Rightarrow B) \Rightarrow \neg\neg B)}$$

$B_4 = (B \Rightarrow (\neg\neg\neg B \Rightarrow B))$,
axiom $A1$ for $A = B, B = \neg\neg\neg B$

$B_5 = (B \Rightarrow \neg\neg B)$,
B_3, B_4 and Lemma 5.2**a** for $A = B, B = (\neg\neg\neg B \Rightarrow B), C = \neg\neg B$, i.e.

$$(B \Rightarrow (\neg\neg\neg B \Rightarrow B)), ((\neg\neg\neg B \Rightarrow B) \Rightarrow \neg\neg B) \vdash_{H_2} (B \Rightarrow \neg\neg B)$$

Example 5.3

We prove that

$$\vdash_{H_2} (\neg A \Rightarrow (A \Rightarrow B)) \tag{5.15}$$

by constructing its formal proof $B_1, \ \dots, \ B_{12}$ *as follows.*

$B_1 = \neg A,$

$B_2 = A,$

$B_3 = (A \Rightarrow (\neg B \Rightarrow A)),$

$B_4 = (\neg A \Rightarrow (\neg B \Rightarrow \neg A)),$

$B_5 = (\neg B \Rightarrow A),$

$B_6 = (\neg B \Rightarrow \neg A),$

$B_7 = ((\neg B \Rightarrow \neg A) \Rightarrow ((\neg B \Rightarrow A) \Rightarrow B)),$

$B_8 = ((\neg B \Rightarrow A) \Rightarrow B),$

$B_9 = B,$

$B_{10} = \neg A, A \vdash B,$

$B_{11} = \neg A \vdash (A \Rightarrow B),$

$B_{12} = (\neg A \Rightarrow (A \Rightarrow B)).$

Example 5.4

We prove that

$$\vdash_{H_2} ((\neg B \Rightarrow \neg A) \Rightarrow (A \Rightarrow B)) \tag{5.16}$$

by constructing its formal proof $B_1, \ldots, B_7$ *as follows. Here are consecutive steps*

$B_1 = (\neg B \Rightarrow \neg A),$

$B_2 = ((\neg B \Rightarrow \neg A) \Rightarrow ((\neg B \Rightarrow A) \Rightarrow B)),$

$B_3 = (A \Rightarrow (\neg B \Rightarrow A)),$

$B_4 = ((\neg B \Rightarrow A) \Rightarrow B),$

$B_5 = (A \Rightarrow B),$

$B_6 = (\neg B \Rightarrow \neg A) \vdash (A \Rightarrow B),$

$B_7 = ((\neg B \Rightarrow \neg A) \Rightarrow (A \Rightarrow B)).$

Example 5.5

We prove that

$$\vdash_{H_2} ((A \Rightarrow B) \Rightarrow (\neg B \Rightarrow \neg A)) \tag{5.17}$$

by constructing its formal proof $B_1, \ldots, B_9$ *as follows. Here are consecutive steps*

$B_1 = (A \Rightarrow B)$,

$B_2 = (\neg\neg A \Rightarrow A)$,

$B_3 = (\neg\neg A \Rightarrow B)$,

$B_4 = (B \Rightarrow \neg\neg B)$,

$B_5 = (\neg\neg A \Rightarrow \neg\neg B)$,

$B_6 = ((\neg\neg A \Rightarrow \neg\neg B) \Rightarrow (\neg B \Rightarrow \neg A))$,

$B_7 = (\neg B \Rightarrow \neg A)$,

$B_8 = (A \Rightarrow B) \vdash (\neg B \Rightarrow \neg A)$,

$B_9 = ((A \Rightarrow B) \Rightarrow (\neg B \Rightarrow \neg A))$.

Exercise 5.5

Complete the proof $B_1, \ldots, B_9$ *of (5.17) by providing comments how each step of the proof was obtained.*

Solution
The proof of (5.17) with comments complementing it is as follows.

$B_1 = (A \Rightarrow B)$,
hypothesis

$B_2 = (\neg\neg A \Rightarrow A)$,
Example 5.10 for $B = A$

$B_3 = (\neg\neg A \Rightarrow B)$,
Lemma 5.2 **a** for $A = \neg\neg A,\ B = A,\ C = B$

$B_4 = (B \Rightarrow \neg\neg B)$,
Example 5.11

$B_5 = (\neg\neg A \Rightarrow \neg\neg B)$,
Lemma 5.2 **a** for $A = \neg\neg A,\ B = B,\ C = \neg\neg B$

$B_6 = ((\neg\neg A \Rightarrow \neg\neg B) \Rightarrow (\neg B \Rightarrow \neg A))$,
Example 5.4 for $B = \neg A,\ A = \neg B$

$B_7 = (\neg B \Rightarrow \neg A)$,
$B_5,\ B_6$ and MP

$B_8 = (A \Rightarrow B) \vdash (\neg B \Rightarrow \neg A)$,
$B_1 - B_7$

$B_9 = ((A \Rightarrow B) \Rightarrow (\neg B \Rightarrow \neg A))$.
  Deduction Theorem 5.31

Example 5.6

We prove that

$$\vdash_{H_2} ((A \Rightarrow B) \Rightarrow ((\neg A \Rightarrow B) \Rightarrow B)) \tag{5.18}$$

by constructing its formal proof $B_1, \ldots, B_{12}$ *as follows. Here are consecutive steps.*

$B_1 = (A \Rightarrow B)$,

$B_2 = (\neg A \Rightarrow B)$,

$B_3 = ((A \Rightarrow B) \Rightarrow (\neg B \Rightarrow \neg A))$,

$B_4 = (\neg B \Rightarrow \neg A)$,

$B_5 = ((\neg A \Rightarrow B) \Rightarrow (\neg B \Rightarrow \neg\neg A))$,

$B_6 = (\neg B \Rightarrow \neg\neg A)$,,

$B_7 = ((\neg B \Rightarrow \neg\neg A) \Rightarrow ((\neg B \Rightarrow \neg A) \Rightarrow B)))$,

$B_8 = ((\neg B \Rightarrow \neg A) \Rightarrow B)$,

$B_9 = B$,

$B_{10} = (A \Rightarrow B), (\neg A \Rightarrow B) \vdash B$,

$B_{11} = (A \Rightarrow B) \vdash ((\neg A \Rightarrow B) \Rightarrow B)$,

$B_{12} = ((A \Rightarrow B) \Rightarrow ((\neg A \Rightarrow B) \Rightarrow B))$.

Exercise 5.6

Complete the proof $B_1, \ldots, B_{12}$ *of (5.18) by providing comments how each step of the proof was obtained.*

Solution
The proof of (5.18) with comments complementing it is as follows.

$B_1 = (A \Rightarrow B)$,
  hypothesis

$B_2 = (\neg A \Rightarrow B)$,
  hypothesis

$B_3 = ((A \Rightarrow B) \Rightarrow (\neg B \Rightarrow \neg A))$,
  Example 5.5

$B_4 = (\neg B \Rightarrow \neg A),$
$\quad B_1,\ B_3$ and MP

$B_5 = ((\neg A \Rightarrow B) \Rightarrow (\neg B \Rightarrow \neg\neg A))$
$\quad$ Example 5.5 for $A = \neg A,\ B = B$

$B_6 = (\neg B \Rightarrow \neg\neg A),$
$\quad B_2,\ B_5$ and MP

$B_7 = ((\neg B \Rightarrow \neg\neg A) \Rightarrow ((\neg B \Rightarrow \neg A) \Rightarrow B))),$
$\quad$ axiom A3 for $B = B,\ A = \neg A$

$B_8 = ((\neg B \Rightarrow \neg A) \Rightarrow B),$
$\quad B_6,\ B_7$ and MP

$B_9 = B,$
$\quad B_4,\ B_8$ and MP

$B_{10} = (A \Rightarrow B), (\neg A \Rightarrow B) \vdash_{H_2} B,$
$\quad B_1 - B_9$

$B_{11} = (A \Rightarrow B) \vdash ((\neg A \Rightarrow B) \Rightarrow B),$
$\quad$ Deduction Theorem 5.31

$B_{12} = ((A \Rightarrow B) \Rightarrow ((\neg A \Rightarrow B) \Rightarrow B)).$ Deduction Theorem 5.31

Example 5.7

We prove that

$$\vdash_{H_2} ((\neg A \Rightarrow A) \Rightarrow A) \tag{5.19}$$

by constructing its formal proof $B_1,\ B_2,\ B_3$ *as follows. Here are consecutive steps.*

$B_1 = ((\neg A \Rightarrow \neg A) \Rightarrow ((\neg A \Rightarrow A) \Rightarrow A))),$

$B_2 = (\neg A \Rightarrow \neg A),$

$B_3 = ((\neg A \Rightarrow A) \Rightarrow A)).$

Exercise 5.7

Complete the proof $B_1,\ B_2,\ B_3$ *of (5.19) by providing comments how each step of the proof was obtained.*

Solution
The proof of (5.19) with comments complementing it is as follows.

$B_1 = ((\neg A \Rightarrow \neg A) \Rightarrow ((\neg A \Rightarrow A) \Rightarrow A))),$
$\quad$ axiom A3 for $B = A$

$B_2 = (\neg A \Rightarrow \neg A)$,
Lemma 5.2 for $A = \neg A$

$B_3 = ((\neg A \Rightarrow A) \Rightarrow A))$.
$B_1,\ B_2$ and MP

The above Examples 5.10 - 5.7 and the Fact 5.1 provide a proof of the following lemma.

Lemma 5.3

For any formulas $A, B, Cin\mathcal{F}$ of the system H_2 the following holds.

1. $\vdash_{H_2} (A \Rightarrow A)$;
2. $\vdash_{H_2} (\neg\neg B \Rightarrow B)$;
3. $\vdash_{H_2} (B \Rightarrow \neg\neg B)$;
4. $\vdash_{H_2} (\neg A \Rightarrow (A \Rightarrow B))$;
5. $\vdash_{H_2} ((\neg B \Rightarrow \neg A) \Rightarrow (A \Rightarrow B))$;
6. $\vdash_{H_2} ((A \Rightarrow B) \Rightarrow (\neg B \Rightarrow \neg A))$;
7. $\vdash_{H_2} (A \Rightarrow (\neg B \Rightarrow (\neg(A \Rightarrow B)))$;
8. $\vdash_{H_2} ((A \Rightarrow B) \Rightarrow ((\neg A \Rightarrow B) \Rightarrow B))$;
9. $\vdash_{H_2} ((\neg A \Rightarrow A) \Rightarrow A$.

The set of provable formulas from the above Lemma 5.3 includes a set of provable formulas needed, with H_2 axioms to execute two proofs of the Completeness Theorem 5.6 for H_2. These two proofs represent two very different methods of proving Completeness Theorem.

5.2 Completeness Theorem: Proof One

The Proof One of the Completeness Theorem 5.6 for H_2. presented here is similar in its structure to the proof of the Deduction Theorem 5.3 and is due to Kalmar, 1935. It is, as Deduction Theorem was, a *constructive proof.* It means it defines a method how one can use the assumption that a formula A is a tautology in order to *construct* its formal proof. We hence call it a *proof construction method.* It relies heavily on the Deduction Theorem.

Proof One, the first proof of the Completeness Theorem 5.6 presented here is very elegant and simple, but is applicable only to the classical propositional logic. Methods it uses are specific to a propositional language $\mathcal{L}_{\{\neg, \Rightarrow\}}$ and the proof system H_2. Nevertheless, it can be adopted and extended to other classical propositional languages $\mathcal{L}_{\{\neg, \cup, \Rightarrow\}}$, $\mathcal{L}_{\{\neg, \cap, \cup, \Rightarrow\}}$, $\mathcal{L}_{\{\neg, \cap, \cup, \Rightarrow, \Leftrightarrow\}}$, and proof systems based on them. We do so by adding appropriate new logical axioms to the logical axioms of H_2 (section 5.2.1). Such obtained proof systems are called *extentions* of the system H_2. It means that one can think about the system H_2, i.e. an axiomatization given by set $\{A1, A2, A3\}$ of logical axioms of H_2, and its language $\mathcal{L}_{\{\neg, \Rightarrow\}}$ as in a sense, a "minimal one" for classical propositional logic and its languages that contain implication.

Proof One, i.e. the methods of carrying it, can't be extended to the classical predicate logic, not to mention variety of non-classical logics. Hence we present, in the next section5.3 another, more general proof, called Proof Two, that can.

We have already proved the Soundness Theorem 5.5 for H_2, so in order to prove the Completeness Theorem 5.6 we need to prove only the completeness part of the completeness theorem, i.e. the following implication.

For any formula A of H_2,

$$\text{if} \quad \models A, \quad \text{then} \quad \vdash_S A. \tag{5.20}$$

In order to prove (5.20), i.e. to prove that any tautology has a formal proof in H_2, we need first to present one definition and prove one lemma stated below. We write $\vdash\ A$ instead of $\vdash_{H_2} A$, as the system H_2 is fixed.

Definition 5.4

Let A be a formula and $b_1, b_2, ..., b_n$ be all propositional variables that occur in A. Let v be variable assignment $v : VAR \longrightarrow \{T, F\}$. We define, for $A, b_1, b_2, ..., b_n$ and v a corresponding formulas A', $B_1, B_2, ..., B_n$ as follows:

$$A' = \begin{cases} A & \textit{if } v^*(A) = T \\ \neg A & \textit{if } v^*(A) = F \end{cases}$$

$$B_i = \begin{cases} b_i & \textit{if } v(b_i) = T \\ \neg b_i & \textit{if } v(b_i) = F \end{cases}$$

for $i = 1, 2, ..., n$.

Example 5.8

Let A be a formula

$$(a \Rightarrow \neg b) \tag{5.21}$$

and let v be such that

$$v(a) = T, \quad v(b) = F. \tag{5.22}$$

In this case $b_1 = a$, $b_2 = b$, and $v^*(A) = v^*(a \Rightarrow \neg b) = v(a) \Rightarrow \neg v(b) = T \Rightarrow \neg F = T$. The corresponding A', B_1, B_2 are: $A' = A$ (as $v^*(A) = T$), $B_1 = a$ (as $v(a) = T$), $B_2 = \neg b$ (as $v(b) = F$).

Here is a simple exercise.

Exercise 5.8

Let A be a formula $((\neg a \Rightarrow \neg b) \Rightarrow c)$ and let v be such that $v(a) = T$, $v(b) = F, v(c) = F$.

Evaluate A', $B_1, ...B_n$ as defined by the definition 5.4.

Solution
In this case $n = 3$ and $b_1 = a$, $b_2 = b, b_3 = c$, and $v^*(A) = v^*((\neg a \Rightarrow \neg b) \Rightarrow c)$ $=((\neg v(a) \Rightarrow \neg v(b)) \Rightarrow v(c)) = ((\neg T \Rightarrow \neg F) \Rightarrow F) = (T \Rightarrow F) = F$. The corresponding A', B_1, B_2, B_2 are: $A' = \neg A = \neg((\neg a \Rightarrow \neg b) \Rightarrow c)$ (as $v^*(A) = F$), $B_1 = a$ (as $v(a) = T$), $B_2 = \neg b$ (as $v(b) = F$). $B_3 = \neg c$ (as $v(c) = F$).

The lemma stated below describes a method of transforming a semantic notion of a tautology into a syntactic notion of provability. It defines, for any formula A and a variable assignment v a corresponding deducibility relation $\vdash$.

Lemma 5.4 (Main Lemma)

For any formula A and a variable assignment v, if $A^{'}$, B_1 , B_2, ..., B_n are corresponding formulas defined by 5.4, then

$$B_1, B_2, ..., B_n \ \vdash \ A'. \qquad (5.23)$$

Example 5.9 *Let A, v be as defined by (5.21) and (5.22), respectively.*

1. The Lemma 5.4 asserts that $a, \neg b \ \vdash \ (a \Rightarrow \neg b)$.

Let A, v be as defined in Exercise 5.8.

2. The Lemma 5.4 asserts that $a, \neg b, \neg c \ \vdash \ \neg((\neg a \Rightarrow \neg b) \Rightarrow c)$.

Proof of the Main Lemma

The Main Lemma 5.4 states: for any formula A and a variable assignment v, if $A^{'}$, B_1 , B_2, ..., B_n are corresponding formulas defined by Definition 5.4, then

$$B_1, B_2, ..., B_n \ \vdash \ A'.$$

Proof We carry the proof by mathematical induction on the degree of A i.e. a number n of logical connectives in A.

Case: $n = 0$

In the case that $n = 0$ A is atomic and so consists of a single propositional variable, say a. We have to cases to consider, $v^*(A) = T$ or $v^*(A) = F$. Clearly, if $v^*(A) = T$ then we $A' = A = a$, $B_1 = a$, and $a \vdash a$ holds by the Deduction Theorem and 11.15. I.e. $\vdash (a \Rightarrow a)$ holds by **??**). Applying the the Deduction Theorem we get $a \vdash a$.

If $v^*(A) = F$ then we $A' = \neg A = \neg a$, $B_1 = \neg a$, and $\vdash (\neg a \Rightarrow \neg a)$ holds by Lemma 5.3. Applying the the Deduction Theorem we get $\neg a \vdash \neg a$. So the lemma holds for the case $n = 0$.

Now assume that the lemma holds for any A with $j < n$ logical connectives (any A of the degree $j < n$). The goal is to prove that it holds for A with the degree n.
There are several sub-cases to deal with.

Case: A **is** $\neg A_1$

If A is of the form $\neg A_1$ then A_1 has less then n connectives and by the inductive assumption we have the formulas $A_1^{'}$, B_1 , B_2, ..., B_n corresponding to the A_1 and the propositional variables $b_1, b_2, ..., b_n$ in A_1, as defined by the definition 5.4, such that

$$B_1, B_2, ..., B_n \ \vdash \ A_1^{'}. \tag{5.24}$$

Observe, that the formulas A and $\neg A_1$ have the same propositional variables, so the corresponding formulas B_1 , B_2, ..., B_n are the same for both of them. We are going to show that the inductive assumption (5.24) allows us to prove that the lemma holds for A, ie. that

$$B_1, B_2, ..., B_n \ \vdash \ A^{'}.$$

There two cases to consider.

Case: $v^*(A_1) = T$

If $v^*(A_1) = T$ then by definition 5.4 $A_1^{'} = A_1$ and by the inductive assumption (5.24)

$$B_1, B_2, ..., B_n \ \vdash \ A_1. \tag{5.25}$$

In this case $v^*(A) = v^*(\neg A_1) = \neg v^*(T) = F$ and so $A^{'} = \neg A = \neg\neg A_1$. We have by Lemma 5.3, $\vdash \ (A_1 \Rightarrow \neg\neg A_1)$, By the monotonicity, $B_1, B_2, ..., B_n \ \vdash (A_1 \Rightarrow \neg\neg A_1)$. By inductive assumption (5.25) and Modus Ponens we have that also $B_1, B_2, ..., B_n \ \vdash \neg\neg A_1$, that is $B_1, B_2, ..., B_n \ \vdash \neg A$, that is $B_1, B_2, ..., B_n \ \vdash A^{'}$.

Case: $v^*(A_1) = F$

If $v^*(A_1) = F$ then $A_1^{'} = \neg A_1$ and $v^*(A) = T$ so $A^{'} = A$. Therefore the inductive assumption (5.24) $B_1, B_2, ..., B_n \ \vdash \ \neg A_1$, that is $B_1, B_2, ..., B_n \ \vdash \ A^{'}$.

Case: A is $(A_1 \Rightarrow A_2)$

If A is of the form $(A_1 \Rightarrow A_2)$ then A_1 and A_2 have less than n connectives.

$A = A(b_1, \ldots b_n)$ so there are some subsequences $c_1, ..., c_k$ and $d_1, ...d_m$, for $k, m \leq n$, of the sequence $b_1, ..., b_n$ such that $A_1 = A_1(c_1, ..., c_k)$ and $A_2 = A(d_1, ...d_m)$. A_1 and A_2 have less than n connectives and so by the inductive assumption we have appropriate formulas $C_1, ..., C_k$ and $D_1, ...D_m$ such that $C_1, C_2, \ldots, C_k \vdash A_1^{'}$ and $D_1, D_2, \ldots, D_m \vdash A_2^{'}$. The formulas $C_1, C_2, ..., C_k$ and $D_1, D_2, ..., D_m$ are *subsequences* of formulas $B_1, B_2, ..., B_n$ corresponding to the propositional variables in A. Hence by monotonicity we have also that have $B_1, B_2, ..., B_n \vdash A_1^{'}$ and $B_1, B_2, ..., B_n \vdash A_2^{'}$, where $B_1, B_2, ..., B_n$ are formulas corresponding to the propositional variables in A.

Now we have the following sub-cases to consider.

Case: $v^*(A_1) = v^*(A_2) = T$

If $v^*(A_1) = T$ then $A_1^{'}$ is A_1 and if $v^*(A_2) = T$ then $A_2^{'}$ is A_2. We also have $v^*(A_1 \Rightarrow A_2) = T$ and so $A^{'}$ is $(A_1 \Rightarrow A_2)$. By the above and the inductive assumption, therefore, $B_1, B_2, ..., B_n \vdash A_2$ and by Lemma 5.3, i.e. $\vdash (A_2 \Rightarrow (A_1 \Rightarrow A_2))$. By monotonicity and Modus Ponens, that $B_1, B_2, ..., B_n \vdash (A_1 \Rightarrow A_2)$, that is $B_1, B_2, ..., B_n \vdash A^{'}$.

Case: $v^*(A_1) = T, v^*(A_2) = F$

If $v^*(A_1) = T$ then $A_1^{'}$ is A_1 and if $v^*(A_2) = F$ then $A_2^{'}$ is $\neg A_2$. Also we have in this case $v^*(A_1 \Rightarrow A_2) = F$ and so $A^{'}$ is $\neg(A_1 \Rightarrow A_2)$. By the above and the inductive assumption, therefore, $B_1, B_2, ..., B_n \vdash \neg A_2$. By Lemma 5.3, $\vdash (A_1 \Rightarrow (\neg A_2 \Rightarrow \neg(A_1 \Rightarrow A_2)))$. By monotonicity and Modus Ponens twice, we have that $B_1, B_2, ..., B_n \vdash \neg(A_1 \Rightarrow A_2)$, that is $B_1, B_2, ..., B_n \vdash A^{'}$.

Case: $v^*(A_1) = F$

If $v^*(A_1) = F$ then $A_1^{'}$ is $\neg A_1$ and, whatever value v gives A_2, we have $v^*(A_1 \Rightarrow A_2) = T$ and so $A^{'}$ is $(A_1 \Rightarrow A_2)$. Therefore, $B_1, B_2, ..., B_n \vdash \neg A_1$ and by Lemma 5.3, $\vdash (\neg A_1 \Rightarrow (A_1 \Rightarrow A_2))$. By monotonicity and Modus Ponens we get that $B_1, B_2, ..., B_n \vdash (A_1 \Rightarrow A_2)$, that is $B_1, B_2, ..., B_n \vdash A^{'}$.

With that we have covered all cases and, by induction on n, the proof of the lemma is complete.

Proof of the Completeness Theorem

Now we use the Main Lemma 5.4 to prove the completeness part of the Completeness Theorem 5.6, i.e. to prove the implication (5.20):

$$\text{For any formula } A \in \mathcal{F}, \text{ if } \models A, \text{then } \vdash A.$$

Proof Assume that $\models A$. Let $b_1, b_2, ..., b_n$ be all propositional variables that occur in A, i.e. $A = A(b_1, b_2, ..., b_n)$.

Let $v : VAR \to \{T, F\}$ be any variable assignment, and

$$v_A : \quad \{b_1, b_2, ..., b_n\} \quad \to \{T, F\} \tag{5.26}$$

its restriction to the formula A, i.e. $v_A = v|\{b_1, b_2, ..., b_n\}$. Let

$$V_A = \{v_A : \quad v_A : \{b_1, b_2, ..., b_n\} \quad \to \{T, F\}\} \tag{5.27}$$

By the Main Lemma 5.4 and the assumption that $\models A$ any $v \in V_A$ defines formulas B_1 , B_2, ..., B_n such that

$$B_1, B_2, ..., B_n \;\vdash\; A. \tag{5.28}$$

The proof is based on a method of using all $v \in V_A$ to define a process of elimination of all hypothesis $B_1, B_2, ..., B_n$ in (5.28) to finally construct the proof of A in H_2 i.e. to prove that $\vdash\; A$.

Step 1: elimination of B_n.

Observe that by definition 5.4, each B_i is b_i or $\neg b_i$ depending on the choice of $v \in V_A$. In particular $B_n = b_n$ or $B_n = \neg b_n$. We choose two truth assignments $v_1 \neq v_2 \in V_A$ such that

$$v_1|\{b_1, ..., b_{n-1}\} = v_2|\{b_1, ..., b_{n-1}\} \tag{5.29}$$

and $v_1(b_n) = T$ and $v_2(b_n) = F$.

Case 1: $v_1(b_n) = T$, by definition 5.4 $B_n = b_n$. By the property (5.29), assumption that $\models A$, and the Main Lemma 5.4 applied to v_1

$$B_1, B_2, ..., B_{n-1}, b_n \;\vdash\; A.$$

By Deduction Theorem 5.3 we have that

$$B_1, B_2, ..., B_{n-1} \;\vdash\; (b_n \Rightarrow A). \tag{5.30}$$

Case 2: $v_2(b_n) = F$ hence by definition 5.4 $B_n = \neg b_n$. By the property (5.29), assumption that $\models A$, and the Main Lemma 5.4 applied to v_2

$$B_1, B_2, ...B_{n-1}, \neg b_n \;\vdash\; A.$$

By the Deduction Theorem 5.3 we have that

$$B_1, B_2, ..., B_{n-1} \;\vdash\; (\neg b_n \Rightarrow A). \tag{5.31}$$

By Lemma 5.3 of the formula $\vdash ((A \Rightarrow B) \Rightarrow ((\neg A \Rightarrow B) \Rightarrow B))$. Hence for for $A = b_n, B = A$ we have that

$$\vdash ((b_n \Rightarrow A) \Rightarrow ((\neg b_n \Rightarrow A) \Rightarrow A)).$$

By monotonicity we have that

$$B_1, B_2, ..., B_{n-1} \ \vdash \ ((b_n \Rightarrow A) \Rightarrow ((\neg b_n \Rightarrow A) \Rightarrow A)). \tag{5.32}$$

Applying Modus Ponens twice to the above property (5.32) and properties (5.30), (5.31) we get that

$$B_1, B_2, ..., B_{n-1} \ \vdash \ A. \tag{5.33}$$

We have eliminated B_n.

Step 2: elimination of B_{n-1} from (5.33). We repeat the Step 1.

As before we have 2 cases to consider: $B_{n-1} = b_{n-1}$ or $B_{n-1} = \neg b_{n-1}$. We choose two truth assignments $w_1 \neq w_2 \in V_A$ such that

$$w_1|\{b_1, ..., b_{n-2}\} = w_2|\{b_1, ..., b_{n-2}\} = v_1|\{b_1, ..., b_{n-2}\} = v_2|\{b_1, ..., b_{n-2}\} \tag{5.34}$$

and $w_1(b_{n-1}) = T$ and $w_2(b_{n-1}) = F$.

As before we apply Main Lemma, Deduction Theorem, monotonicity, proper substitutions of the formula $((A \Rightarrow B) \Rightarrow ((\neg A \Rightarrow B) \Rightarrow B))$, and Modus Ponens twice and eliminate B_{n-1} just as we eliminated B_n.

After n steps, we finally obtain that

$$\vdash \ A.$$

This **ends** the proof of Completeness Theorem.

Observe that our proof of the fact that $\vdash A$ is a constructive one. Moreover, we have used in it only Main Lemma 5.4 and Deduction Theorem 5.3, and both of them have fully constructive proofs. So we can always reconstruct all steps in proofs which use the Main Lemma 5.4and Deduction Theorem 5.3 back to the original axioms of H_2. The same applies to the proofs that use the formulas proved in H_2 that are stated in Lemma 5.3.

It means that for any $A \in \mathcal{F}$, such that $\models \ A$, the set V_A of all v restricted to A provides us a method of a construction of the formal proof of A in H_2 from its axioms $A1, \ A2, \ A3$ only. .

5.2.1 Examples

Example 5.10

As an example of how the **Proof One** *of the Completeness Theorem works, we consider a following tautology*

$$\models (a \Rightarrow (\neg a \Rightarrow b))$$

and show how to construct its proof, i.e. to show that

$$\vdash (a \Rightarrow (\neg a \Rightarrow b)).$$

We apply the Main Lemma 5.4 to all possible variable assignments $v \in V_A$. We have 4 variable assignments to consider.

Case 1: $v(a) = T, v(b) = T$.
In this case $B_1 = a, B_2 = b$ and, as in all cases, $A^{'} = A$ and by the Main Lemma 5.4

$$a, b \ \vdash \ (a \Rightarrow (\neg a \Rightarrow b)).$$

Case 2: $v(a) = T, v(b) = F$.
In this case $B_1 = a, B_2 = \neg b$ and by the Main Lemma 5.4

$$a, \neg b \ \vdash \ (a \Rightarrow (\neg a \Rightarrow b)).$$

Case 3: $v(a) = F, v(b) = T$.
In this case $B_1 = \neg a, B_2 = b$ and by the Main Lemma 5.4

$$\neg a, b \ \vdash \ (a \Rightarrow (\neg a \Rightarrow b)).$$

Case 4: $v(a) = F, v(b) = F$.
In this case $B_1 = \neg a, B_2 = \neg b$ and by the lemma 5.4

$$\neg a, \neg b \ \vdash \ (a \Rightarrow (\neg a \Rightarrow b)).$$

Applying the Deduction Theorem 5.3 to the cases above we have that

D1 (Cases 1 and 2)

$$a \ \vdash \ (b \Rightarrow (a \Rightarrow (\neg a \Rightarrow b))),$$

$$a \ \vdash \ (\neg b \Rightarrow (a \Rightarrow (\neg a \Rightarrow b))),$$

D2 (Cases 2 and 3)

$$\neg a \ \vdash \ (b \Rightarrow (a \Rightarrow (\neg a \Rightarrow b))),$$

$$\neg a \ \vdash \ (\neg b \Rightarrow (a \Rightarrow (\neg a \Rightarrow b))).$$

By the monotonicity and the proper substitution of formula

$$((A \Rightarrow B) \Rightarrow ((\neg A \Rightarrow B) \Rightarrow B))$$

the provable by Lemma 5.3, we have that

$$a \vdash ((b \Rightarrow (a \Rightarrow (\neg a \Rightarrow b))) \Rightarrow ((\neg b \Rightarrow (a \Rightarrow (\neg a \Rightarrow b))) \Rightarrow (a \Rightarrow (\neg a \Rightarrow b))),$$

$$\neg a \vdash ((b \Rightarrow (a \Rightarrow (\neg a \Rightarrow b))) \Rightarrow ((\neg b \Rightarrow (a \Rightarrow (\neg a \Rightarrow b))) \Rightarrow (a \Rightarrow (\neg a \Rightarrow b))).$$

Applying Modus Ponens twice to **D1**, **D2** and these above, respectively, gives us

$$a \vdash (a \Rightarrow (\neg a \Rightarrow b)) \textit{ and}$$
$$\neg a \vdash (a \Rightarrow (\neg a \Rightarrow b)).$$

Applying the Deduction Theorem 5.3 to the above we obtain

D3 $\vdash (a \Rightarrow (a \Rightarrow (\neg a \Rightarrow b))),$

D4 $\vdash (\neg a \Rightarrow (a \Rightarrow (\neg a \Rightarrow b))).$

We form now an appropriate form of the formula

$$((A \Rightarrow B) \Rightarrow ((\neg A \Rightarrow B) \Rightarrow B)), \tag{5.35}$$

provable by the Lemma 5.3. The appropriate form is

$$\Rightarrow ((\neg a \Rightarrow (a \Rightarrow (\neg a \Rightarrow b))) \Rightarrow (a \Rightarrow (\neg a \Rightarrow b)))). \tag{5.36}$$

We apply Modus Ponens twice to **D3** and **D4** and (5.58) and get finally the proof of $(a \Rightarrow (\neg a \Rightarrow b))$, i.e. we have proved that

$$\vdash (a \Rightarrow (\neg a \Rightarrow b)).$$

Example 5.11

The Proof One of Completeness Theorem defines a method of efficiently combining $v \in V_A$ as defined in (5.27), while constructing the proof of A. Let's consider the following tautology $A = A(a, b, c)$

$$((\neg a \Rightarrow b) \Rightarrow (\neg(\neg a \Rightarrow b) \Rightarrow c).$$

We present bellow all steps of Proof One as applied to A.

By the Main Lemma 5.4 and the assumption that $\models A(a, b, c)$ any $v \in V_A$ defines formulas B_a , B_b, B_c such that

$$B_a, B_b, B_c \vdash A. \tag{5.37}$$

The proof is based on a method of using all $v \in V_A$ (there is 16 of them) to define a process of elimination of all hypothesis B_a, B_b, B_c in (5.37) to construct the proof of A in H_2 i.e. to prove that $\vdash A$.

Step 1: elimination of B_c.

Observe that by definition 5.4, B_c is c or $\neg c$ depending on the choice of $v \in V_A$. We choose two truth assignments $v_1 \neq v_2 \in V_A$ such that

$$v_1|\{a,b\} = v_2|\{a,b\} \tag{5.38}$$

and $v_1(c) = T$ and $v_2(c) = F$.

Case 1: $v_1(c) = T$, by definition 5.4 $B_c = c$. By the property (5.38), assumption that $\models A$, and the Main Lemma 5.4 applied to v_1

$$B_a, B_b, c \vdash A.$$

By Deduction Theorem 5.3 we have that

$$B_a, B_b \vdash (c \Rightarrow A). \tag{5.39}$$

Case 2: $v_2(c) = F$ hence by definition 5.4 $B_c = \neg c$. By the property (5.38), assumption that $\models A$, and the Main Lemma 5.4 applied to v_2

$$B_a, B_b, \neg c \vdash A.$$

By the Deduction Theorem 5.3 we have that

$$B_a, B_b \vdash (\neg c \Rightarrow A). \tag{5.40}$$

By Lemma 5.3, i.e. provability of the formula (5.35) for $A = c, B = A$ we have that

$$\vdash ((c \Rightarrow A) \Rightarrow ((\neg c \Rightarrow A) \Rightarrow A)).$$

By monotonicity we have that

$$B_a, B_b \vdash ((c \Rightarrow A) \Rightarrow ((\neg c \Rightarrow A) \Rightarrow A)). \tag{5.41}$$

Applying Modus Ponens twice to the above property (5.41) and properties (5.39), (5.40) we get that

$$B_a, B_b \vdash A. \tag{5.42}$$

and hence we have eliminated B_c.

Step 2: elimination of B_b from (5.42). We repeat the Step 1.

As before we have 2 cases to consider: $B_b = b$ or $B_b = \neg b$. We choose from V_A two truth assignments $w_1 \neq w_2 \in V_A$ such that

$$w_1|\{a\} = w_2|\{a\} = v_1|\{a\} = v_2|\{a\} \tag{5.43}$$

and $w_1(b) = T$ and $w_2(b) = F$.

Case 1: $w_1(b) = T$, by definition 5.4 $B_b = b$. By the property (5.43), assumption that $\models A$, and the Main Lemma 5.4 applied to w_1

$$B_a, b \vdash A.$$

By Deduction Theorem 5.3 we have that

$$B_a \vdash (b \Rightarrow A). \tag{5.44}$$

Case 2: $w_2(c) = F$ hence by definition 5.4 $B_b = \neg b$. By the property (5.3), assumption that $\models A$, and the Main Lemma 5.4 applied to w_2

$$B_a, \neg b \vdash A.$$

By the Deduction Theorem 5.3 we have that

$$B_a \vdash (\neg b \Rightarrow A). \tag{5.45}$$

By Lemma 5.3, i.e. provability of the formula (5.35) for $A = b, B = A$ we have that

$$\vdash ((b \Rightarrow A) \Rightarrow ((\neg b \Rightarrow A) \Rightarrow A)).$$

By monotonicity we have that

$$B_a \vdash ((b \Rightarrow A) \Rightarrow ((\neg b \Rightarrow A) \Rightarrow A)). \tag{5.46}$$

Applying Modus Ponens twice to the above property (5.46) and properties (5.44), (5.45) we get that

$$B_a \vdash A. \tag{5.47}$$

and hence we have eliminated B_b.

Step 3: elimination of B_a from (5.47). We repeat the Step 2.
As before we have 2 cases to consider: $B_a = a$ or $B_a = \neg a$. We choose from V_A two truth assignments $g_1 \neq g_2 \in V_A$ such that

$$g_1(a) = T \text{ and } g_2(a) = F. \tag{5.48}$$

Case 1: $g_1(a) = T$, by definition 5.4 $B_a = a$. By the property (5.48), assumption that $\models A$, and the Main Lemma 5.4 applied to g_1

$$a \vdash A.$$

By Deduction Theorem 5.3 we have that

$$\vdash (a \Rightarrow A). \tag{5.49}$$

Case 2: $g_2(a) = F$ hence by definition 5.4 $B_a = \neg a$. By the property (5.48), assumption that $\models A$, and the Main Lemma 5.4 applied to g_2

$$\neg a \ \vdash \ A.$$

By the Deduction Theorem 5.3 we have that

$$\vdash \ (\neg a \Rightarrow A). \tag{5.50}$$

By Lemma 5.3, i.e. provability of the formula (5.35) for $A = a, B = A$ we have that

$$\vdash ((a \Rightarrow A) \Rightarrow ((\neg a \Rightarrow A) \Rightarrow A)). \tag{5.51}$$

Applying Modus Ponens twice to the above property (5.51) and properties (5.49), (5.50) we get that

$$\vdash \ A. \tag{5.52}$$

and hence we have eliminated B_a, B_b and B_c and constructed the proof of A.

5.3 Completeness Theorem: Proof Two

The Proof Two is much more complicated then the Proof One. Its strength and importance lies in a fact that the methods it uses can be applied in an extended version to the proof of completeness for classical predicate logic and even many of non-classical propositional and predicate logics. The main point of the proof is a presentation of a general, non- constructive method for proving existence of a counter-model for any non-provable A. The generality of the method makes it possible to adopt it for other cases of predicate and some non-classical logics. We call it a a *counter-model existence method.*

We prove now the completeness part of the Completeness Theorem 5.6 for H_2 by proving that the opposite implication:

$$\text{if} \ \ \nvdash \ A, \ \text{then} \ \ \not\models \ A \tag{5.53}$$

to the implication (5.20):

$$\text{if} \ \ \models A, \text{then} \ \ \vdash A$$

holds hat for all $A \in \mathcal{F}$.

We will show now how one can define of a counter-model for A from the fact that A is not provable. This means that we deduce that a formula A is not a tautology from the fact that it does not have a proof. We hence call it a *a counter-model existence method.*

The definition of the counter-model for any non-provable A is much more general (and less constructive) then in the case of the Proof One in section 5.2. It can be generalized to the case of predicate logic, and many of non-classical logics; propositional and predicate. It is hence a much more general method then the first one and this is the reason we present it here.

We remind that $\not\models\ A$ means that there is a truth assignment $v : VAR \longrightarrow \{T, F\}$, such that $v^*(A) \neq T$, i.e. in classical semantics, such that that $v^*(A) = F$. Such v is called a counter-model for A, hence the proof provides a counter-model construction method.

Since we assume in (8.16) that A does not have a proof in H_2 ($\nvdash\ A$) the method uses this information in order to show that A is not a tautology, i.e. to define v such that $v^*(A) = F$. We also have to prove that all steps in that method are correct. This is done in the following steps.

Step 1: Definition of Δ^*

We use the information $\nvdash\ A$ to define a special set $\Delta^* \subseteq \mathcal{F}$, such that $\neg A \in \Delta^*$.

Step 2: Counter - model definition

We define the truth assignment $v : VAR \longrightarrow \{T, F\}$ as follows:

$$v(a) = \begin{cases} T & \text{if } \Delta^* \vdash a \\ F & \text{if } \Delta^* \vdash \neg a. \end{cases}$$

Step 3: Prove that v is a counter-model

We first prove a more general property, namely we prove that the set Δ^* and v defined in the steps 1 and 2, respectively, are such that for every formula $B \in \mathcal{F}$,

$$v^*(B) = \begin{cases} T & \text{if } \Delta^* \vdash B \\ F & \text{if } \Delta^* \vdash \neg B. \end{cases}$$

Then we use the **Step 1** (definition of Δ^*) to prove that $v^*(A) = F$.

The definition and the properties of the set Δ^*, and hence the **Step 1**, are the most essential for the proof. The other steps have mainly technical character. The main notions involved in the **Step 1** (definition of Δ^*) are: *consistent set, complete set* and a *consistent complete extension of a set.* We are going now to introduce them and to prove some essential facts about them.

Consistent and Inconsistent Sets

There exist two definitions of consistency; semantical and syntactical. The **semantical** one uses definition the notion of a model and says, in plain English:

a set of formulas is consistent if it has a model.

The **syntactical** one uses the notion of provability and says:

a set of formulas is consistent if one can't prove a contradiction from it.

In our Proof Two of the Completeness Theorem we use assumption that a given formula A does not have a proof to deduce that A is not a tautology. We hence use the following syntactical definition of consistency.

Consistent set

We say that a set $\Delta \subseteq \mathcal{F}$ of formulas is **consistent** if and only if **there is no** a formula $A \in \mathcal{F}$ such that

$$\Delta \vdash A \quad \text{and} \quad \Delta \vdash \neg A. \tag{5.54}$$

Inconsistent set

A set $\Delta \subseteq \mathcal{F}$ is inconsistent if and only if **there is** a formula $A \in \mathcal{F}$ such that $\Delta \vdash A$ and $\Delta \vdash \neg A$.

The notion of consistency, as defined above, is characterized by the following lemma.

Lemma 5.5 (Consistency Condition)

For every set $\Delta \subseteq \mathcal{F}$ of formulas, the following conditions are equivalent:

(i) Δ *is* consistent,

(ii) *there is a formula $A \in \mathcal{F}$ such that $\Delta \nvdash A$.*

Proof The implications: **(i)** implies **(ii)** and vice-versa are proved by showing the corresponding opposite implications. I.e. to establish the equivalence of **(i)** and **(ii)**, we first show that **not (ii)** implies **not (i)**, and then that **not (i)** implies **not (ii)**.

Case 1

Assume that **not (ii)**. It means that for all formulas $A \in \mathcal{F}$ we have that $\Delta \vdash A$. In particular it is true for a certain $A = B$ and $A = \neg B$ and hence proves that Δ is inconsistent, i.e. **not (i)** holds.

Case 2

> Assume that **not (i)**, i.e that Δ is inconsistent. Then there is a formula A such that $\Delta \vdash A$ and $\Delta \vdash \neg A$. Let B be any formula. Since $(\neg A \Rightarrow (A \Rightarrow B))$ is provable in H_2 by Lemma 5.3, hence by applying Modus Ponens twice and by detaching from it $\neg A$ first, and A next, we obtain a formal proof of B from the set Δ, so that $\Delta \vdash B$ for any formula B. Thus **not (ii)**.

The inconsistent sets are hence characterized by the following fact.

Lemma 5.6 (Inconsistency Condition)

For every set $\Delta \subseteq \mathcal{F}$ of formulas, the following conditions are equivalent:

(i) Δ *is* inconsistent*,*

(ii) *for all formulas $A \in \mathcal{F}$, $\Delta \vdash A$.*

We remind here the property of the finiteness of the consequence operation.

Lemma 5.7

For every set Δ of formulas and for every formula $A \in \mathcal{F}$, $\Delta \vdash A$ if and only if there is a finite subset $\Delta_0 \subseteq \Delta$ such that $\Delta_0 \vdash A$.

Proof
If $\Delta_0 \vdash A$ for a certain $\Delta_0 \subseteq \Delta$, then by the monotonicity of the consequence, also $\Delta \vdash A$. Assume now that $\Delta \vdash A$ and let $A_1, A_2, ..., A_n$ be a formal proof of A from Δ. Let $\Delta_0 = \{A_1, A_2, ..., A_n\} \cap \Delta$. Obviously, Δ_0 is finite and $A_1, A_2, ..., A_n$ is a formal proof of A from Δ_0.

The following theorem is a simply corollary of the above Lemma 5.7.

Theorem 5.7 (Finite Inconsistency)

If a set Δ is inconsistent, then there is a finite subset $\Delta_0 \subseteq \Delta$ which is inconsistent. It follows therefore from that if every finite subset of a set Δ is consistent, then the set Δ is also consistent.

Proof
If Δ is inconsistent, then for some formula A, $\Delta \vdash A$ and $\Delta \vdash \neg A$. By above Lemma 5.7, there are finite subsets Δ_1 and Δ_2 of Δ such that $\Delta_1 \vdash A$ and $\Delta_2 \vdash \neg A$. By monotonicity, the union $\Delta_1 \cup \Delta_2$ is a finite subset of Δ, such that $\Delta_1 \cup \Delta_2 \vdash A$ and $\Delta_1 \cup \Delta_2 \vdash \neg A$. Hence $\Delta_1 \cup \Delta_2$ is a finite inconsistent subset of Δ. The second implication is the opposite to the one just proved and

hence also holds.

The following lemma links the notion of non-provability and consistency. It will be used as an important step in our proof of the Completeness Theorem.

Lemma 5.8

For any formula $A \in \mathcal{F}$, if $\nvdash A$, then the set $\{\neg A\}$ is consistent.

Proof
If $\{\neg A\}$ is inconsistent, then by the Inconsistency Condition 5.6 we have $\{\neg A\} \vdash A$. This and the Deduction Theorem 5.3 imply $\vdash (\neg A \Rightarrow A)$. Applying the Modus Ponens rule to $\vdash (\neg A \Rightarrow A)$ a formula $((\neg A \Rightarrow A) \Rightarrow A)$, provable by LemmaH2lemma, we get that $\vdash A$, contrary to the assumption of the lemma.

Complete and Incomplete Sets

Another important notion, is that of *a complete* set of formulas. Complete sets, as defined here are sometimes called *maximal*, but we use the first name for them. They are defined as follows.

Complete set

A set Δ of formulas is called complete if **for every** formula $A \in \mathcal{F}$,

$$\Delta \vdash A \;\; or \;\; \Delta \vdash \neg A. \tag{5.55}$$

The complete sets are characterized by the following fact.

Lemma 5.9 (Complete set condition)

For every set $\Delta \subseteq \mathcal{F}$ of formulas, the following conditions are equivalent:

(i) Δ *is* complete*,*

(ii) *for every formula $A \in \mathcal{F}$, if $\Delta \nvdash A$, then the set $\Delta \cup \{A\}$ is inconsistent.*

Proof
We consider two cases. We show that **(i)** implies **(ii)** and vice-versa, that **(ii)** also implies **(i)**.

Case 1

Assume that **(i)** and that for every formula $A \in \mathcal{F}$, $\Delta \nvdash A$, we have to show that in this case $\Delta \cup \{A\}$ is inconsistent. But if $\Delta \nvdash A$, then from the definition of complete set and assumption that Δ is complete set, we get that $\Delta \vdash \neg A$. By the monotonicity of the consequence we have that

$\Delta \cup \{A\} \vdash \neg A$ as well. Since, by formula 11.15 we have $\vdash (A \Rightarrow A)$, by monotonicity $\Delta \vdash (A \Rightarrow A)$ and by Deduction Theorem $\Delta \cup \{A\} \vdash A$. This proves that $\Delta \cup \{A\}$ is inconsistent. Hence **(ii)** holds.

Case 2

Assume that **(ii)**. Let A be any formula. We want to show that the condition: $\Delta \vdash A$ or $\Delta \vdash \neg A$ is satisfied. If $\Delta \vdash \neg A$, then the condition is obviously satisfied.

If, on other hand, $\Delta \nvdash \neg A$, then we are going to show now that it must be , under the assumption of **(ii)**, that $\Delta \vdash A$, i.e. that **(i)** holds.

Assume that $\Delta \nvdash \neg A$, then by **(ii)**, the set $\Delta \cup \{\neg A\}$ is inconsistent. It means, by the Consistency Condition 5.5, that $\Delta \cup \{\neg A\} \vdash A$. By the Deduction Theorem 5.3, this implies that $\Delta \vdash (\neg A \Rightarrow A)$. Since $((\neg A \Rightarrow A) \Rightarrow A)$ is provable in H_2 (Lemma 5.3), by monotonicity $\Delta \vdash ((\neg A \Rightarrow A) \Rightarrow A)$. Detaching $(\neg A \Rightarrow A)$, we obtain that $\Delta \vdash A$, what ends the proof that **(i)** holds.

Incomplete set

A set Δ of formulas is called incomplete if it is not complete, i.e. if **there exists** a formula $A \in \mathcal{F}$ such that

$$\Delta \nvdash A \ \ and \ \ \Delta \nvdash \neg A. \tag{5.56}$$

We get as a direct consequence of the lemma 5.9 the following characterization of incomplete sets.

Lemma 5.10 (Incomplete Set Condition)

For every set $\Delta \subseteq \mathcal{F}$ of formulas, the following conditions are equivalent:

(i) Δ *is* incomplete*,*

(ii) *there is formula $A \in \mathcal{F}$ such that $\Delta \nvdash A$ and the set $\Delta \cup \{A\}$ is consistent.*

Main Lemma: Complete Consistent Extension

Now we are going to prove a lemma that is essential to the construction of the special set Δ^* mentioned in the **Step 1** of the proof of the Completeness Theorem, and hence to the proof of the theorem itself. Let's first introduce one more notion.

Extensions

A set Δ^* of formulas is called an **extension** of a set Δ of formulas if the following condition holds.

$$\{A \in \mathcal{F} : \ \ \Delta \ \vdash \ A\} \subseteq \{A \in \mathcal{F} : \ \ \Delta^* \ \vdash \ A\}. \tag{5.57}$$

In this case we say also that Δ **extends** to the set of formulas Δ^*.

The Main Lemma states as follows.

Lemma 5.11 (Complete Consistent Extension)

Every consistent set Δ of formulas can be extended to a complete consistent set Δ^* of formulas.

Proof
Assume that the lemma does not hold, i.e. that there is a consistent set Δ, such that all its consistent extensions are not complete. In particular, as Δ is an consistent extension of itself, we have that Δ is not complete.

The proof consists of a construction of a particular set Δ^* and proving that it forms a complete consistent extension of Δ, contrary to the assumption that all its consistent extensions are not complete.

Construction of Δ^*.

As we know, the set $\mathcal{F}$ of all formulas is enumerable. They can hence be put in an infinite sequence

$$A_1, A_2, ..., A_n, ... \tag{5.58}$$

such that every formula of $\mathcal{F}$ occurs in that sequence exactly once.

We define now, as the first step in the construction of Δ^*, an infinite sequence $\{\Delta_n\}_{n\in N}$ of consistent subsets of formulas together with a sequence $\{B\}_{n\in N}$ of formulas as follows.

Initial Step

In this step we define the sets Δ_1, Δ_2 and the formula B_1. We prove that Δ_1 and Δ_2 are consistent, incomplete extensions of Δ.

We take, as the first set, the set Δ, i.e. we define

$$\Delta_1 = \Delta. \tag{5.59}$$

Since, by assumption, the set Δ, and hence also Δ_1 is not complete, it follows from the Incomplete Set Condition Lemma 5.10, that there is a formula $B \in \mathcal{F}$ such that $\Delta_1 \ \not\vdash \ B$, then and the set $\Delta_1 \cup \{B\}$ is consistent.

Let

$$B_1$$

be the first formula with this property in the sequence (5.58) of all formulas; we then define

$$\Delta_2 = \Delta_1 \cup \{B_1\}. \tag{5.60}$$

The set Δ_2 is consistent and $\Delta_1 = \Delta \subseteq \Delta_2$, so by the monotonicity, Δ_2 is a consistent extension of Δ. Hence Δ_2 cannot be complete.

Inductive Step

Suppose that we have defined a sequence

$$\Delta_1, \Delta_2, ..., \Delta_n$$

of incomplete, consistent extensions of Δ, and a sequence

$$B_1, B_2, ...B_{n-1}$$

of formulas, for $n \geq 2$.

Since Δ_n is incomplete, it follows from the Incomplete Set Condition Lemma 5.10, that there is a formula $B \in \mathcal{F}$ such that $\Delta_n \not\vdash B$ and the set $\Delta_n \cup \{B\}$ is consistent.

Let

$$B_n$$

be the first formula with this property in the sequence (5.58) of all formulas.

We then define

$$\Delta_{n+1} = \Delta_n \cup \{B_n\}. \tag{5.61}$$

By the definition, $\Delta \subseteq \Delta_n \subseteq \Delta_{n+1}$ and the set Δ_{n+1} is consistent. Hence Δ_{n+1} is an incomplete consistent extension of Δ.

By the principle of mathematical induction we have defined an infinite sequence

$$\Delta = \Delta_1 \subseteq \Delta_2 \subseteq ..., \subseteq \Delta_n \subseteq \Delta_{n+1} \subseteq .... \tag{5.62}$$

such that for all $n \in N$, Δ_n is consistent, and moreover, it is an incomplete consistent extension of Δ.

Moreover, we have also defined a sequence

$$B_1, B_2, ..., B_n, ... \tag{5.63}$$

of formulas, such that for all $n \in N$, $\Delta_n \not\vdash B_n$, and the set $\Delta_n \cup \{B_n\}$ is consistent.

Observe that $B_n \in \Delta_{n+1}$ for all $n \geq 1$.

Definition of Δ^*

Now we are ready to define Δ^*, i.e. we define:

$$\Delta^* = \bigcup_{n\in N} \Delta_n. \tag{5.64}$$

To complete the proof our theorem we have now to prove that Δ^* is a complete consistent extension of Δ. Obviously, by the definition, Δ^* is an extension of Δ. Now we prove (by contradiction) the following.

Fact 5.5

The set Δ^* is consistent.

Proof
Assume that Δ^* is inconsistent. By the Finite Inconsistency Theorem 5.7 there is a finite subset Δ_0 of Δ^* that is inconsistent. By Definition 5.64 have that

$$\Delta_0 = \{C_1, ..., C_n\} \subseteq \bigcup_{n\in N} \Delta_n.$$

By the definition, $C_i \in \Delta_{k_i}$ for certain Δ_{k_i} in the sequence (5.62) and $1 \leq i \geq n$. Hence $\Delta_0 \subseteq \Delta_m$ for $m = max\{k_1, k_2, ..k_n\}$. But all sets of the sequence (5.62) are consistent. This contradicts the fact that Δ_m is inconsistent, as it contains an inconsistent subset Δ_0. Hence Δ^* must be consistent.

Fact 5.6

The set Δ^* is complete.

Proof
Assume that Δ^* is not complete. By the Incomplete Set Condition Lemma 5.10, there is a formula $B \in \mathcal{F}$ such that $\Delta^* \not\vdash B$ and the set $\Delta^* \cup \{B\}$ is consistent.

But, by definition (5.64) of Δ^*, the above condition means that for every $n \in N$, $\Delta_n \not\vdash B$ holds and the set $\Delta_n \cup \{B\}$ is consistent.

Since the formula B is one of the formulas of the sequence (5.58) and it would have to be one of the formulas of the sequence (5.63), i.e. $B = B_j$ for certain j. Since $B_j \in \Delta_{j+1}$, it proves that $B \in \Delta^* = \bigcup_{n\in N}$. But this means that $\Delta^* \vdash B$, contrary to the assumption. This proves that Δ^* is a complete consistent extension of Δ and **ends** the proof out our lemma.

Now we are ready to prove the completeness theorem for the system H_2.

Proof of the Completeness Theorem

As by assumption our system H_2 is sound, we have to prove only the Completeness part of the Completeness Theorem 5.6, i.e we have to show the implication

$$\text{if} \quad \models A, \quad \text{then} \quad \vdash A$$

for any formula A. We prove it by proving the logically equivalent opposite implication

$$\text{if} \quad \nvdash A, \quad \text{then} \quad \not\models A.$$

We remind that $\not\models A$ means that there is a variable assignment $v : VAR \longrightarrow \{T, F\}$, such that $v^*(A) \neq T$. In classical case it means that $v^*(A) = F$, i.e. that there is a variable assignment that falsifies A. Such v is also called a **counter-model** for A.

Assume that A doesn't have a proof in S, we want to define a **counter-model** for A. But if $\nvdash A$, then by the lemma 5.8, the set $\{\neg A\}$ is consistent. By the Main Lemma 5.11 there is a complete, consistent extension of the set $\{\neg A\}$, i.e. there is a set set Δ^* such that $\{\neg A\} \subseteq \Delta^*$, i.e.

$$\neg A \in \Delta^*. \tag{5.65}$$

Since Δ^* is a consistent, complete set, it satisfies the following form of the consistency condition 5.54, which says that for any A, $\Delta^* \nvdash A$ or $\Delta^* \nvdash \neg A$. It also satisfies the completeness condition (5.55), which says that for any A, $\Delta^* \vdash A$ or $\Delta^* \vdash \neg A$. This means that for any A, exactly one of the following conditions is satisfied: $\Delta^* \vdash A$, $\Delta^* \vdash \neg A$. In particular, for every propositional variable $a \in VAR$ exactly one of the following conditions is satisfied: $\Delta^* \vdash a$, $\Delta^* \vdash \neg a$. This justifies the correctness of the following definition.

Definition of v

We define the variable assignment

$$v : VAR \longrightarrow \{T, F\} \tag{5.66}$$

as follows:

$$v(a) = \begin{cases} T & \text{if } \Delta^* \vdash a \\ F & \text{if } \Delta^* \vdash \neg a. \end{cases}$$

We show, as a separate lemma below, that such defined variable assignment v has the following property.

Lemma 5.12 (Property of v)

Let v be the variable assignment defined by (5.66) and v^* its extension to the set $\mathcal{F}$ of all formulas. Then for every formula $B \in \mathcal{F}$,

$$v^*(B) = \begin{cases} T & \textit{if } \Delta^* \vdash B \\ F & \textit{if } \Delta^* \vdash \neg B. \end{cases} \tag{5.67}$$

Given the above property (5.67) of v (still to be proven), we prove that the v is in fact, a counter model for any formula A, such that $\nvdash A$ as follows. Let A be such that $\nvdash A$. By (5.65), $\neg A \in \Delta^*$ and obviously, $\Delta^* \vdash \neg A$. Hence, by the property (5.67) of v, $v^*(A) = F$, what proves that v is a counter-model for A and hence ends the proof of the completeness theorem. In order to really complete the proof we still have to write a proof of the Lemma 5.12.

Proof of the Lemma 5.12
The proof is conducted by the induction on the degree of the formula A.

If A is a propositional variable, then the lemma is true holds by (5.66), i.e. by the definition of v.

If A is not a propositional variable, then A is of the form $\neg C$ or $(C \Rightarrow D)$, for certain formulas C, D. By the inductive assumption the lemma, i.e. the property (5.67) holds for the formulas C and D.

Case $A = \neg C$**.** We have to consider two possibilities: $\Delta^* \vdash A$ and $\Delta^* \vdash \neg A$.

Assume $\Delta^* \vdash A$. It means that $\Delta^* \vdash \neg C$. Then from the fact that Δ^* is *consistent* it must be that $\Delta^* \nvdash C$. This means, by the inductive assumption, that $v^*(C) = F$, and accordingly

$$v^*(A) = v^*(\neg C) = \neg v^*(C) = \neg F = T.$$

Assume now that $\Delta^* \vdash \neg A$. Then from the fact that Δ^* is *consistent* it must be that $\Delta^* \nvdash A$. I.e. $\Delta^* \nvdash \neg C$. If so, then $\Delta^* \vdash C$, as the set Δ^* is *complete*. Hence by the inductive assumption, that $v^*(C) = T$, and accordingly

$$v^*(A) = v^*(\neg C) = \neg v^*(C) = \neg T = F.$$

Thus A satisfies the property (5.67).

Case $A = (C \Rightarrow D)$**.** As in the previous case, we assume that the lemma, i.e. the property (5.67) holds for the formulas C, D and we consider two possibilities: $\Delta^* \vdash A$ and $\Delta^* \vdash \neg A$.

Assume $\Delta^* \vdash A$. It means that $\Delta^* \vdash (C \Rightarrow D)$. If at the same time $\Delta^* \nvdash C$, then $v^*(C) = F$, and accordingly

$$v^*(A) = v^*(C \Rightarrow D) = v^*(C) \Rightarrow v^*(D) = F \Rightarrow v^*(D) = T.$$

If at the same time $\Delta^* \vdash C$, then, since $\Delta^* \vdash (C \Rightarrow D)$, we infer, by Modus Ponens, that $\Delta^* \vdash D$. If so, then

$$v^*(C) = v^*(D) = T,$$

and accordingly

$$v^*(A) = v^*(C \Rightarrow D) = v^*(C) \Rightarrow v^*(D) = T \Rightarrow T = T.$$

Thus, if $\Delta^* \vdash A$, then $v^*(A) = T$.

Assume now, as before, that $\Delta^* \vdash \neg A$. Then from the fact that Δ^* is *consistent* it must be that $\Delta^* \nvdash A$, i.e.,

$$\Delta^* \nvdash (C \Rightarrow D).$$

It follows from this that

$$\Delta^* \nvdash D,$$

for if $\Delta^* \vdash D$, then, as $(D \Rightarrow (C \Rightarrow D))$ is provable (Lemma 5.4), by monotonicity

$$\Delta^* \vdash (D \Rightarrow (C \Rightarrow D)).$$

Applying Modus Ponens we obtain $\Delta^* \vdash (C \Rightarrow D)$, which is contrary to the assumption.

Also we must have

$$\Delta^* \vdash C,$$

for otherwise, by the fact that Δ^* we would have

$$\Delta^* \vdash \neg C.$$

But this is impossible, since the formula $(\neg C \Rightarrow (C \Rightarrow D))$ is provable provable (Lemma 5.4) and by monotonicity

$$\Delta^* \vdash (\neg C \Rightarrow (C \Rightarrow D)).$$

Applying Modus Ponens we would get $\Delta^* \vdash (C \Rightarrow D)$, which is contrary to the assumption. This **ends** the proof of the Lemma 5.12 and the **Proof Two** of the Completeness Theorem 5.6.

5.4 Some Other Axiomatizations

We present here some of most known, and historically important axiomatizations of classical propositional logic, i.e. the following Hilbert proof systems.

Łukasiewicz (1929)

$$\text{Ł} = (\ \mathcal{L}_{\{\neg,\ \Rightarrow\}},\ \ \mathcal{F},\ \ A1, A2, A3,\ \ MP\ ), \tag{5.68}$$

where

A1 $((\neg A \Rightarrow A) \Rightarrow A)$,

A2 $(A \Rightarrow (\neg A \Rightarrow B))$,

A3 $((A \Rightarrow B) \Rightarrow ((B \Rightarrow C) \Rightarrow (A \Rightarrow C))))$
for any $A, B, C \in \mathcal{F}$.

2. Hilbert and Ackermann (1928)

$$HA = (\ \mathcal{L}_{\{\neg, \cup\}},\ \ \mathcal{F},\ \ A1 - A4,\ \ MP\ ), \tag{5.69}$$

where

A1 $(\neg(A \cup A) \cup A)$,

A2 $(\neg A \cup (A \cup B))$,

A3 $(\neg(A \cup B) \cup (B \cup A))$,

A4 $(\neg(\neg B \cup C) \cup (\neg(A \cup B) \cup (A \cup C)))$,

for any $A, B, C \in \mathcal{F}$.

Modus Ponens rule in the language $\mathcal{L}_{\{\neg, \cup\}}$ has a form

$$(MP)\ \ \frac{A\ ;\ \ (\neg A \cup B)}{B}.$$

Observe that also the Deduction Theorem is now formulated as follow.

Theorem 5.8 (Deduction Theorem for HA)

For any subset Γ of the set of formulas $\mathcal{F}$ of HA and for any formulas $A, B \in \mathcal{F}$,

$$\Gamma,\ A \vdash_{HA} B\ \ if\ and\ only\ if\ \ \Gamma \vdash_{HA} (\neg A \cup B).$$

In particular,

$$A \vdash_{HA} B\ \ if\ and\ only\ if\ \ \vdash_{HA} (\neg A \cup B).$$

2. Hilbert (1928)

$$H = (\ \mathcal{L}_{\{\neg, \cup, \cap, \Rightarrow\}},\ \ \mathcal{F},\ \ A1 - A15,\ \ MP\ ), \tag{5.70}$$

where

A1 $(A \Rightarrow A)$,

A2 $(A \Rightarrow (B \Rightarrow A))$,

A3 $((A \Rightarrow B) \Rightarrow ((B \Rightarrow C) \Rightarrow (A \Rightarrow C)))$,

A4 $((A \Rightarrow (A \Rightarrow B)) \Rightarrow (A \Rightarrow B))$, A5 $((A \Rightarrow (B \Rightarrow C)) \Rightarrow (B \Rightarrow (A \Rightarrow C)))$,

A6 $((A \Rightarrow B) \Rightarrow ((B \Rightarrow C) \Rightarrow (A \Rightarrow C)))$,

A7 $((A \cap B) \Rightarrow A)$,

A8 $((A \cap B) \Rightarrow B)$,

A9 $((A \Rightarrow B) \Rightarrow ((A \Rightarrow C) \Rightarrow (A \Rightarrow (B \cap C)))$,

A10 $(A \Rightarrow (A \cup B))$,

A11 $(B \Rightarrow (A \cup B))$,

A12 $((A \Rightarrow C) \Rightarrow ((B \Rightarrow C) \Rightarrow ((A \cup B) \Rightarrow C)))$,

A13 $((A \Rightarrow B) \Rightarrow ((A \Rightarrow \neg B) \Rightarrow \neg A))$,

A14 $(\neg A \Rightarrow (A \Rightarrow B))$,

A15 $(A \cup \neg A)$,

for any $A, B, C \in \mathcal{F}$.

Kleene (1952)

$$K = (\ \mathcal{L}_{\{\neg,\cup,\cap,\Rightarrow\}},\quad \mathcal{F},\quad A1 - A10,\quad MP\ ), \tag{5.71}$$

where

A1 $(A \Rightarrow (B \Rightarrow A))$,

A2 $((A \Rightarrow (B \Rightarrow C)) \Rightarrow (B \Rightarrow (A \Rightarrow C)))$,

A3 $((A \cap B) \Rightarrow A)$,

A4 $((A \cap B) \Rightarrow B)$,

A5 $(A \Rightarrow (B \Rightarrow (A \cap B)))$,

A6 $(A \Rightarrow (A \cup B))$,

A7 $(B \Rightarrow (A \cup B))$,

A8 $((A \Rightarrow C) \Rightarrow ((B \Rightarrow C) \Rightarrow ((A \cup B) \Rightarrow C)))$,

A9 $((A \Rightarrow B) \Rightarrow ((A \Rightarrow \neg B) \Rightarrow \neg A))$,

A10 $(\neg\neg A \Rightarrow A)$

for any $A, B, C \in \mathcal{F}$.

Rasiowa-Sikorski (1950)

$$RS = (\ \mathcal{L}_{\{\neg,\cup,\cap,\Rightarrow\}},\quad \mathcal{F},\quad A1 - A12,\quad MP\ ), \tag{5.72}$$

where

A1 $((A \Rightarrow B) \Rightarrow ((B \Rightarrow C) \Rightarrow (A \Rightarrow C)))$,

A2 $(A \Rightarrow (A \cup B))$,

A3 $(B \Rightarrow (A \cup B))$,

A4 $((A \Rightarrow C) \Rightarrow ((B \Rightarrow C) \Rightarrow ((A \cup B) \Rightarrow C)))$,

A5 $((A \cap B) \Rightarrow A)$,

A6 $((A \cap B) \Rightarrow B)$,

A7 $((C \Rightarrow A) \Rightarrow ((C \Rightarrow B) \Rightarrow (C \Rightarrow (A \cap B)))$,

A8 $((A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \cap B) \Rightarrow C))$,

A9 $(((A \cap B) \Rightarrow C) \Rightarrow (A \Rightarrow (B \Rightarrow C))$,

A10 $(A \cap \neg A) \Rightarrow B)$,

A11 $((A \Rightarrow (A \cap \neg A)) \Rightarrow \neg A)$,

A12 $(A \cup \neg A)$,

for any $A, B, C \in \mathcal{F}$.

Here is the shortest axiomatization for the language $\mathcal{L}_{\{\neg, \Rightarrow\}}$. It contains just one axiom.

Meredith (1953)

$$Ł = (\ \mathcal{L}_{\{\neg, \Rightarrow\}},\ \ \mathcal{F},\ \ A1\ \ MP\ ), \tag{5.73}$$

where

A1 $((((((A \Rightarrow B) \Rightarrow (\neg C \Rightarrow \neg D)) \Rightarrow C) \Rightarrow E)) \Rightarrow ((E \Rightarrow A) \Rightarrow (D \Rightarrow A)))$.

We have proved in chapter **??** that

$$\mathcal{L}_{\{\neg, \cup, \cap, \Rightarrow\}} \equiv \mathcal{L}_{\{\uparrow\}} \equiv \mathcal{L}_{\{\downarrow\}}.$$

Here is another axiomatization that uses only one axiom,

Nicod (1917)

$$N = (\ \mathcal{L}_{\{\uparrow\}},\ \ \mathcal{F},\ \ A1,\ \ (r)\ ), \tag{5.74}$$

where

A1 $(((A \uparrow (B \uparrow C)) \uparrow ((D \uparrow (D \uparrow D)) \uparrow ((E \uparrow B) \uparrow ((A \uparrow E) \uparrow (A \uparrow E))))))$.
The rule of inference is (r) is expressed in the language $\mathcal{L}_{\{\uparrow\}}$ as

$$\frac{A \uparrow (B \uparrow C)}{A}.$$

5.5 Exercises

Here are few exercises designed to help the readers with understanding the notions of completeness, monotonicity of the consequence operation, the role of the deduction theorem and importance of some basic tautologies.
Let S be any Hilbert proof system

$$S = (\mathcal{L}_{\{\cap,\cup,\Rightarrow,\neg\}},\ \mathcal{F},\ LA,\ (MP)\frac{A, (A \Rightarrow B)}{B}) \tag{5.75}$$

with its set LA of logical axioms such that S is **complete** under classical semantics.

Let $X \subseteq \mathcal{F}$ be any subset of the set $\mathcal{F}$ of formulas of the language $\mathcal{L}_{\{\cap,\cup,\Rightarrow,\neg\}}$ of S. We define, as we did in chapter 4, a set $Cn(X)$ of all **consequences** of the set X as

$$Cn(X) = \{A \in \mathcal{F} : \quad X \vdash_S A\}. \tag{5.76}$$

Plainly speaking, the set $Cn(X)$ of all **consequences** of the set X is the set of all formulas that can be proved in S from the set $(LA \cup X)$.

All exercises 5.9 - 5.13 concern the system S defined by (5.75).

Exercise 5.9

*1. Prove that for any subsets X, Y of the set $\mathcal{F}$ of formulas the following **monotonicity property** holds.*

$$\textit{If } X \subseteq Y, \textit{ then } Cn(X) \subseteq Cn(Y). \tag{5.77}$$

2. Do we need the **completeness** *of S to prove that the monotonicity property holds for S?*

Solution
1. Let $A \in \mathcal{F}$ be any formula such that $A \in Cn(X)$. By (5.76), we have that $X \vdash_S A$. This means that A has a formal proof from the set $X \cup LA$. But $X \subseteq Y$, hence this proof is also a proof from $Y \cup LA$, i.e . $Y \vdash_S A$, and hence $A \in Cn(Y)$. This proves that $Cn(X) \subseteq Cn(Y)$.

2. No, we do not need the completeness of S for the monotonicity property to hold. We have used only the definition of a formal proof from the hypothesis X and the definition of the consequence operation.

Exercise 5.10

Prove that for any set $X \subseteq \mathcal{F}$, the set $\mathbf{T} \subseteq \mathcal{F}$ of all propositional classical tautologies of the language $\mathcal{L}_{\{\cap,\cup,\Rightarrow,\neg\}}$ of the system S is a subset of $Cn(X)$, i.e. prove that

$$\mathbf{T} \subseteq Cn(X). \tag{5.78}$$

2. Do we need the **completeness** *of S to prove that the property (5.78) holds for S?*

Solution

1. The proof system S is complete, so by the completeness theorem we have that

$$\mathbf{T} = \{\in \mathcal{F} : \ \vdash_S A\}. \tag{5.79}$$

By definition (5.76) of the consequence,

$$\{A \in \mathcal{F} : \ \vdash_S A\} = Cn(\emptyset)$$

and hence $Cn(\emptyset) = \mathbf{T}$. But $\emptyset \subseteq X$ for any set X, so by monotonicity property (5.77),

$$\mathbf{T} \subseteq Cn(X).$$

2. Yes, the completeness (5.79) of S in the main property used. The next one is the monotonicity property (5.77).

Exercise 5.11

Prove that for any formulas $A, B \in \mathcal{F}$, and for any set $X \subseteq \mathcal{F}$,

$$(A \cap B) \in Cn(X) \ \textit{ if and only if } \ A \in Cn(X) \ \textit{and} \ B \in Cn(X). \tag{5.80}$$

List all properties essential to the proof.

1. **Proof of the implication:**

$$\text{if } \ (A \cap B) \in Cn(X), \ \text{then } A \in Cn(X) \ \text{ and } B \in Cn(X).$$

Assume $(A \cap B) \in Cn(X)$, i.e.

$$X \vdash_S (A \cap B). \tag{5.81}$$

From monotonicity property proved in exercise 5.9, completeness of S, and the fact that

$$\models ((A \cap B) \Rightarrow A) \text{ and } \ \models ((A \cap B) \Rightarrow B) \tag{5.82}$$

we get that

$$X \vdash_S ((A \cap B) \Rightarrow A), \text{ and } \ X \vdash_S ((A \cap B) \Rightarrow B). \tag{5.83}$$

By the assumption (5.81) we have that $X \vdash_S (A \cap B)$, by (8.3), $X \vdash_S ((A \cap B) \Rightarrow A)$, and so we get $X \vdash_S A$ by Modus Ponens.

Similarly, $X \vdash_S (A \cap B)$, by the assumption (5.81), $X \vdash_S ((A \cap B) \Rightarrow B)$ by by (**??**), and so we get $X \vdash_S B$ by MP. This proves that $A \in Cn(X)$ and $B \in Cn(X)$ and **ends** the proof of the implication 1.

2. **Proof of the implication:**

$$\text{if } \ A \in Cn(X) \text{ and } B \in Cn(X), \text{ then } (A \cap B) \in Cn(X).$$

Assume now that $A \in Cn(X)$ and $B \in Cn(X)$, i.e.

$$X \vdash_S A, \text{ and } X \vdash_S B. \tag{5.84}$$

By the monotonicity property, completeness of S, and a tautology $(A \Rightarrow (B \Rightarrow (A \cap B)))$, we get that

$$X \vdash_S (A \Rightarrow (B \Rightarrow (A \cap B))). \tag{5.85}$$

By the assumption (5.84) we have that $X \vdash_S A$, $X \vdash_S B$, by (8.17), $X \vdash_S (A \Rightarrow (B \Rightarrow (A \cap B)))$, so we get $X \vdash_S (B \Rightarrow (A \cap B))$ by Modus Ponens. Applying Modus Ponens again we obtain $X \vdash_S (A \cap B)$. This proves that $(A \cap B) \in Cn(X)$ and ends the proof and the implication 2, and the completes the proof of (5.80).

Exercise 5.12

Let S be the proof system (5.75). Prove that the Deduction Theorem holds for S, i.e. prove the following.

For any subset Γ of the set of formulas $\mathcal{F}$ of S and for any formulas $A, B \in \mathcal{F}$,

$$\Gamma,\ A \vdash_S B \quad \textit{if and only if} \ \ \Gamma \vdash_S (A \Rightarrow B). \tag{5.86}$$

Solution
The formulas $A1 = (A \Rightarrow (B \Rightarrow A))$ and $A2 = ((A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C)))$ are basic propositional tautologies. By the completeness of S we have that

$$\vdash_S (A \Rightarrow (B \Rightarrow A)) \text{ and } \vdash_S ((A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C))). \tag{5.87}$$

The formulas $A1, A2$ are axioms of the Hilbert system H_1 defined by (5.1). By (5.87) both axioms $A1, A2$ of H_1 are provable in S. These axioms were sufficient for the proof of the Deduction Theorem 5.3 for H_1 and its proof now can be repeated for the system S.

Exercise 5.13

Prove that for any $A, B \in \mathcal{F}$,

$$Cn(\{A, B\}) = Cn(\{(A \cap B)\})$$

1. **Proof of the inclusion:** $Cn(\{A, B\}) \subseteq Cn(\{(A \cap B)\})$.

Assume $C \in Cn(\{A, B\})$, i.e. $\{A, B\} \vdash_S C$, what we usually write as $A,\ B \vdash_S C$. Observe that by exercise 8.7 the Deduction Theorem (theorem 5.3) holds for S. We apply Deduction Theorem to the assumption $A,\ B \vdash_S C$ twice we get that the assumption is equivalent to

$$\vdash_S (A \Rightarrow (B \Rightarrow C)). \tag{5.88}$$

We use completeness of S, the fact that the formula $(((A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \cap B) \Rightarrow C)))$ is a tautology, and by monotonicity and get that

$$\vdash_S (((A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \cap B) \Rightarrow C))). \tag{5.89}$$

Applying Modus Ponens to the assumption (5.88) and (8.22) we get $\vdash_S ((A \cap B) \Rightarrow C)$. This is equivalent to $(A \cap B) \vdash_S C$ by Deduction Theorem. We have proved that $C \in Cn(\{(A \cap B)\})$.

2. **Proof of the inclusion:** $Cn(\{(A \cap B)\}) \subseteq Cn(\{A, B\})\})$.

Assume that $C \in Cn(\{(A \cap B)\})$, i.e. $(A \cap B) \vdash_S C$. By Deduction Theorem,

$$\vdash_S ((A \cap B) \Rightarrow C). \tag{5.90}$$

We want to prove that $C \in Cn(\{A, B\})$. This is equivalent, by the Deduction Theorem applied twice to proving that

$$\vdash_S (A \Rightarrow (B \Rightarrow C)).$$

The proof as similar to the previous case. We use completeness of S, the fact that the formula $(((A \cap B) \Rightarrow C) \Rightarrow (A \Rightarrow (B \Rightarrow C)))$ is a tautology and by monotonicity and get that

$$\vdash_S (((A \cap B) \Rightarrow C) \Rightarrow (A \Rightarrow (B \Rightarrow C)) \tag{5.91}$$

Applying Modus Ponens to the assumption (5.88) and (8.22) we get $\vdash_S (A \Rightarrow (B \Rightarrow C))$ what **ends** the proof.

5.6 Homework Problems

For the formulas A_i and corresponding truth assignments v find formulas $B_1, ..B_k, A_i^{'}$ as described by the Main Lemma 5.4, i.e. such that

$$B_1, ...B_k \vdash A_i^{'}.$$

1. $A_1 = ((\neg(b \Rightarrow a) \Rightarrow \neg a) \Rightarrow ((\neg b \Rightarrow (a \Rightarrow \neg c)) \Rightarrow c))$
   $v(a) = T, v(b) = F, v(c) = T$.

2. $A_2 = ((a \Rightarrow (c \Rightarrow (\neg b \Rightarrow c))) \Rightarrow ((\neg d \Rightarrow (a \Rightarrow (\neg a \Rightarrow b))) \Rightarrow (a \Rightarrow (\neg a \Rightarrow b))))$
   $v(a) = F, v(b) = F, v(c) = T, v(d) = F$

3. $A_3 = (\neg b \Rightarrow (c \Rightarrow (\neg a \Rightarrow b)))$
   $v(a) = F, v(b) = F, v(c) = T$

4. $A_4 = (\neg a_1 \Rightarrow (a_2 \Rightarrow (\neg a_3 \Rightarrow a_1)))$
   $v(a_1) = F, v(a_2) = F, v(a_3) = T$

4. $A_5 = ((b \Rightarrow (a_1 \Rightarrow (\neg c \Rightarrow b))) \Rightarrow ((\neg b \Rightarrow (a_2 \Rightarrow (\neg a_1 \Rightarrow b))) \Rightarrow (c \Rightarrow (\neg a \Rightarrow b)))$
   $v(a) = F, v(b) = T, v(c) = F, v(a_1) = T, v(a_2) = F$

For any of the formulas listed below construct their formal proofs, as described in the Proof One of the Completeness Theorem. Follow example 5.10, or example 5.11.

1. $A_1 = (\neg\neg b \Rightarrow b)$
2. $A_2 = ((a \Rightarrow b) \Rightarrow (\neg b \Rightarrow \neg a))$
3. $A_3 = (\neg(a \Rightarrow b) \Rightarrow \neg(\neg b \Rightarrow \neg a))$
4. $A_4 = (\neg(\neg(a \Rightarrow \neg b) \Rightarrow \neg c) \Rightarrow \neg(b \Rightarrow \neg c))$
5. $A_5 = ((a \Rightarrow (b \Rightarrow \neg a)) \Rightarrow (\neg(b \Rightarrow \neg a) \Rightarrow \neg a))$.

   Read carefully proofs of Deduction Theorem 5.3 and Completeness Theorem 5.6 and write careful answers to the following problems.

6. List all formulas that have to be provable in H_2, axioms included, that are are needed for the proof of Deduction Theorem 5.3. Write down each part of the proof that uses them.
7. List all formulas that have to be provable in H_2, axioms included, that are needed for the proof of Main Lemma 5.4.
8. List all formulas that have to be provable in H_2, axioms included, that are included in the Proof of Completeness Theorem part of the Proof One.
9. List all formulas that have to be provable in H_2, axioms included, that are needed to carry all of the Proof One of Completeness Theorem **??**.

10. We proved the Completeness Theorem for the proof system H_2 based on the language $\mathcal{L}_{\{\neg,\Rightarrow\}}$. Extend the H_2 proof system to a proof system S_1 based on a language $\mathcal{L}_{\{\neg,\Rightarrow,\cup\}}$ by adding new logical axioms, as we did in a case of H_1 and H_2 systems. The added logical axioms must be such that they allow to adopt the Proof One to S_1, i.e. such that it is a complete proof system with respect to classical semantics.

11. Repeat the same for the language $\mathcal{L}_{\{\neg,\Rightarrow,\cap\}}$. Call resulting proof system S_2.

12. Repeat the same for the language $\mathcal{L}_{\{\neg,\Rightarrow,\cap,\cup\}}$, i.e. extends systems S_1 or S_2 to a complete proof system S_3 based on the language $\mathcal{L}_{\{\neg,\Rightarrow,\cap,\cup\}}$.

13. Prove Completeness Theorem for the system S_3 from the previous problem.

Completeness Proof Two Problems

1. List all formulas that have to be provable in H_2, axioms included, that are are needed for the Proof Two if the Completeness Theorem 5.6.

2. We proved the Completeness Theorem 5.6 for the proof system H_2 based on the language $\mathcal{L}_{\{\neg,\Rightarrow\}}$. Extend the H_2 proof system to a proof system S_1 based on a language $\mathcal{L}_{\{\neg,\Rightarrow,\cup\}}$ by adding new logical axioms, as we did in a case of H_1 and H_2 systems. The added logical axioms must be such that they allow to adopt the Proof Two to S_1, i.e. such that it is a complete proof system with respect to classical semantics.

3. Extend the H_2 proof system to a proof system based on a language $\mathcal{L}_{\{\neg,\Rightarrow,\cap\}}$ by adding new logical axioms. Call resulting proof system S_2. The added logical axioms must be such that they allow to adopt the Proof Two to S_2, i.e. such that it is a complete proof system with respect to classical semantics.

4. Repeat the same for the language $\mathcal{L}_{\{\neg,\Rightarrow,\cap,\cup\}}$, i.e. extends systems S_1 or S_2 to a complete proof system S_3 based on the language $\mathcal{L}_{\{\neg,\Rightarrow,\cap,\cup\}}$.

5. Conduct appropriate version of Proof Two of the Completeness Theorem 5.6 for the system S_3 from the previous problem.

Axiomatizations Problems

1. Let HA be Hilbert and Ackermann proof system (5.69). We use abbreviation $(A \Rightarrow B)$ for $(\not A \cup B)$.

   (i) Prove $\vdash_{HA} (A \Rightarrow A)$, for any $A \in \mathcal{F}$.

   (ii) Prove $\vdash_{HA} (A \Rightarrow (B \Rightarrow A))$, for any $A, B \in \mathcal{F}$.

(iii) Prove $\vdash_{HA} ((A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C)))$, for any $A, B, C \in \mathcal{F}$.

(iv) Prove $(A \Rightarrow B), (B \Rightarrow C) \vdash_{HA} (A \Rightarrow C)$, for any $A, B, C \in \mathcal{F}$

(v) Prove Deduction Theorem 5.8.

(vi) Prove $\vdash_{HA} A$ if and only if $\models A$, for any $A \in \mathcal{F}$.

2. Let H be Hilbert proof system (**??**).

   (i) Prove $\vdash_{HA} ((A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C)))$, for any $A, B, C \in \mathcal{F}$.

   (ii) Prove Deduction Theorem for H.

   (ii) Prove Completeness Theorem for H.

3. Let K be Kleene proof system (5.71).

   (i) Prove $\vdash_K (A \Rightarrow A)$, for any $A \in \mathcal{F}$.

   (ii) Prove the following.

   For any subset Γ of the set of formulas $\mathcal{F}$ of K and for any formulas $A, B \in \mathcal{F}$, $\Gamma,\ A \vdash_K B\ \ if\ and\ only\ if\ \ \Gamma \vdash_K ((A \Rightarrow B))$.

Completeness General Problems

1. Let RS be Rasiowa-Sikorski proof system (5.72).

   The set $\mathcal{F}$ of formulas of $\mathcal{L}$ determines an abstract algebra

$$F \;=\; (\ \mathcal{F},\ \cup,\ \cap,\ \Rightarrow,\ \neg\ ), \tag{5.92}$$

   where by performing an operation on a formula (two formulas) means writing the formula having this operation as a main connective. For example $\cap(A, B) = (A \cup B)$. We define an binary relations $\leq$ and $\approx$ in the algebra F of formulas of $\mathcal{L}$ as follows. For any $A, B \in \mathcal{F}$,

$$A \leq B\ \ if\ and\ \ only\ if\ \ \vdash_{RS} (A \Rightarrow B), \tag{5.93}$$

$$A \approx B\ \ if\ and\ \ only\ if\ \ \vdash_{RS} (A \Rightarrow B)\ \ and\ \ \vdash\ (B \Rightarrow A). \tag{5.94}$$

   (i) Prove that the relation $\leq$ defined by (5.93) is a quasi-ordering in $\mathcal{F}$.

   (ii) Prove that the relation $\approx$ defined by (5.93) is an equivalence relation in $\mathcal{F}$. The equivalence class containing a formula A is denoted by $\|A\|$.

   (iii) The quasi ordering $\leq$ in $\mathcal{F}$ as defined by (5.93) induces a relation $\leq$ in $\mathcal{F}/\approx$ defined as follows:

$$\|A\| \leq \|B\|\ \text{ if and only if }\ A \leq B, \text{ i.e.}$$

$$\|A\| \leq \|B\|\ \ if\ and\ only\ if\ \ \vdash_{RS} (A \Rightarrow B). \tag{5.95}$$

Prove that the relation (5.95) is an order relation in $\mathcal{F}/\approx$

(iv) Prove that the relation $\approx$ defined by (5.94) is a congruence in the algebra $\mathcal{F}$ of formulas defined by (8.5).

2. The algebra $LT = (\ \mathcal{F}/\approx,\ \cup,\ \cap,\ \Rightarrow,\ \neg)$, where the operations $\cup,\ \cap,\ \Rightarrow$ and $\neg$ are determined by the congruence relation (5.94) i.e.

$$\|A\| \cup \|B\| = \|(A \cup B)\|,$$

$$\|A\| \cap \|B\| = \|(A \cap B)\|,$$

$$\|A\| \Rightarrow \|B\| = \|(A \Rightarrow B)\|,$$

$$\neg\|A\| = \|\neg A\|,$$

   is called a Lindenbaum-Tarski algebra of RS.

   Prove that the Lindenbaum-Tarski algebra of RS as defined by (5.72) is a Boolean algebra. The unit element is the greatest element in $(\mathcal{F}/\approx, \leq)$, where the order relation $\leq$ is defined by (5.95).

3. Formulate and prove the Deduction Theorem for Hilbert and Ackermann system (5.69).

4. Formulate and prove the Deduction Theorem for Lukasiewicz system (5.68).

5. Formulate and prove the Deduction Theorem Kleene system (5.71).

6. Formulate and prove the Deduction Theorem Rasiowa-Sikorski system (5.72)

7. Let HS be any Hilbert proof system based on a language $\mathcal{L}_{HS}$. Prove that if HS is complete under classical semantic, them the Deduction Theorem appropriately expressed in the language $\mathcal{L}_{HS}$ holds for HS.

Chapter 6

Automated Proof Systems Completeness of Classical Propositional Logic

6.1 Gentzen Style Proof System RS

Hilbert style systems are easy to define and admit different proofs of the Completeness Theorem but they are difficult to use. By humans, not mentioning computers. Their emphasis is on logical axioms, keeping the rules of inference, with obligatory Modus Ponens, at a minimum.

Gentzen style proof systems reverse this situation by emphasizing the importance of inference rules, reducing the role of logical axioms to an absolute minimum. They may be less intuitive then the Hilbert-style systems, but they allow us to define effective automatic procedures for proof search, what was impossible in a case of the Hilbert style systems. For this reason they are also called *automated proof systems.* They serve as formal models of computing systems that automate the reasoning process. Building computing systems means providing an algorithmic description to a formal proof system so that it can be implemented on a computer to prove theorems in an efficient manner.

The first proof systems of this style was invented by G. Gentzen in 1934, hence the name. His proof systems for classical and intuitionistic predicate logics introduced special expressions built of formulas called *sequents.* Hence the Gentzen style systems using sequents as basic expressions are often called sequent systems, or Gentzen sequent systems, or simply Gentzen formalizations.

We present here (section 6.5) two Gentzen systems **GL** and **G** for classical propositional logic and prove their completeness. We also present a propositional version of Gentzen original system **LK** and discuss a proof of Gentzen Hauptsatz for it. Hauptsatz is literally rendered as the *main theorem* and is known as *cut-elimination theorem.* We prove the equivalency of the cut-free propositional **LK** and the complete system **G**. The Gentzen original formalization for intuitionistic propositional logic **LI** is discussed and presented in chapter 7. The classical and intuitionistic predicate versions are discussed in chapter 9.

The other historically important automated proof system is due to Rasiowa and Sikorski (1960). Their proof systems for classical propositional and predicate logic use as basic expressions sequences of formulas, less complicated then Gentzen sequents. As they were inspired Gentzen systems we call them, as we call many others similarly inspired, Gentzen style proof system, or Gentzen style formalization. The Rasiowa and Sikorski proof system is simpler and easier to understand then the Gentzen sequent systems. Hence their system **RS** is the first to be presented here in section 6.1.

Historical importance and lasting influence of Rasiowa and Sikorski work lays in the fact that they were first to use the proof searching capacity of their proof system to define a constructive method of proving the completeness theorem for both propositional and predicate classical logic. We introduce and explain in detail their method and use it prove the completeness of the **RS** in section 6.3. We also introduce and discuss two other **RS** style system **RS1** and **RS2** in in section 6.4. We also generalize the **RS** completeness proof method to the Gentzen sequent systems and prove the completeness of **GL** and **G** systems in section 6.5.1. The completeness proof for proof system **RSQ** for classical predicate logic is presented in chapter 9.

We present here a propositional version of the original Rasiowa and Sikorski (1960) Gentzen style proof system for classical logic. We call it **RS system** for **R**asiowa-**S**ikorski. The **RS** system extends naturally to predicate logic **QRS** system which is presented in chapter 8. Both systems admit a constructive proof of Completeness Theorem. We prove completeness of **RS** in section 6.3. We define components and semantics of the system **RS** as follows.

Components of the proof system RS

Language $\mathcal{L}$

Let $\mathcal{F}$ denote a set of formulas of $\mathcal{L} = \mathcal{L}_{\{\neg, \Rightarrow, \cup, \cap\}}$. The rules of inference of our system **RS** operate on *finite sequences of formulas*, i.e. elements of $\mathcal{F}^*$, unlike on plain formulas $\mathcal{F}$ in Hilbert style formalizations.

Expressions $\mathcal{E}$

We adopt as the set of expressions $\mathcal{E}$ of **RS** the set $\mathcal{F}^*$, i.e. $\mathcal{E} = \mathcal{F}^*$. We will denote the expressions of **RS**, i.e. the finite sequences of formulas by Γ, Δ, Σ, with indices if necessary.

Semantic Link

The intuitive meaning of a sequence $\Gamma \in \mathcal{F}^*$ is that the truth assignment v makes it true if and only if it makes the formula of the form of the disjunction of all formulas of Γ true. As we know, the disjunction in classical logic is associative and commutative, i.e., for any formulas $A, B, C \in \mathcal{F}$, the formulas $(A \cup (B \cup C))$, $((A \cup B) \cup C)$, $(A \cup (C \cup B))$, $((B \cup A) \cup C)$, $(C \cup (B \cup A))$, $(C \cup (A \cup B))$, $((C \cup A) \cup B)$, etc... are logically equivalent. In particular we write

$$\delta_{\{A,B,C\}} = A \cup B \cup C$$

to denote any disjunction of formulas A, B, C.

In a general case, for any sequence $\Gamma \in \mathcal{F}^*$, if Γ is of a form

$$A_1, A_2, ..., A_n \tag{6.1}$$

then by δ_Γ we will understand any disjunction of all formulas of Γ. We write it informally

$$\delta_\Gamma = A_1 \cup A_2 \cup ... \cup A_n.$$

Formal Semantics for RS

Let $v : VAR \longrightarrow \{T, F\}$ be a truth assignment, v^* its extension to the set of formulas $\mathcal{F}$. We formally extend v to the set $|calE$ of expressions of **RS** , i.e. to the set $\mathcal{F}^*$ of all finite sequences of $\mathcal{F}$ as follows. For any sequence $\Gamma \in \mathcal{F}^*$, if Γ is the sequence (6.1), then we define:

$$v^*(\Gamma) = v^*(\delta_\Gamma). \tag{6.2}$$

Model

A sequence Γ is said to be **satisfiable** if there is a truth assignment $v : VAR \longrightarrow \{T, F\}$ such that $v^*(\Gamma) = T$. Such a truth assignment is called a **model** for Γ. We denote it as

$$v \models \Gamma. \tag{6.3}$$

Counter- Model

A sequence Γ is said to be **falsifiable** if there is a truth assignment v, such that $v^*(\Gamma) = F$. Such a truth assignment is] called a **counter-model** for Γ. We write it symbolically as

$$v \not\models \Gamma. \tag{6.4}$$

Tautology

The sequence Γ is said to be a **tautology** if $v^*(\Gamma) = T$ for all truth assignments $v : VAR \longrightarrow \{T, F\}$. We write it as

$$\models \Gamma. \tag{6.5}$$

Exercise 6.1

Let Γ be a sequence $a, (b \cap a), \neg b, (b \Rightarrow a)$.

1. Show that the truth assignment $v : VAR \longrightarrow \{T, F\}$, such that $v(a) = F$ and $v(b) = T$ falsifies Γ, i.e. $v \not\models \Gamma$.

2. Let Γ be a sequence $a, (\neg b \cap a), \neg b, (a \cup b)$ and let v be a truth assignment for which $v(a) = T$. Prove that $v \models \Gamma$.

3. Let Γ be a sequence $a, (\neg b \cap a), \neg b, (a \cup b)$. Prove that $\models \Gamma$.

Solution

1. Γ is the sequence $a, (b \cap a), \neg b, (b \Rightarrow a)$. We eveluate $v^*(\Gamma) = v^*(\delta_\Gamma) = v^*(a) \cup v^*(b \cap a) \cup v^*(\neg b) \cup v^*(b \Rightarrow a) = F \cup (F \cap T) \cup F \cup (T \Rightarrow F) = F \cup F \cup F \cup F = F$. By (6.4) we proved $v \not\models \Gamma$.

2. Let Γ be a sequence $a, (\neg b \cap a), \neg b, (a \cup b)$. We eveluate $v^*(\Gamma) = v^*(\delta_\Gamma) = v^*(a) \cup v^*(\neg b \cap a) \cup v^*(\neg b) \cup v^*(a \cup b) = T \cup v^*(\neg b \cap a) \cup v^*(\neg b) \cup v^*(a \cup b) = T$. By (6.3) we proved $v \models \Gamma$.

3. Assume now that Γ is *falsifiable* i.e. that we have a truth assignment v for which $v^*(\Gamma) = v^*(\delta_\Gamma) = v^*(a) \cup v^*(\neg b \cap a) \cup v^*(\neg b) \cup v^*(a \cup b) = F$ This is possible only when (in short-hand notation)

$$a \cup (\neg b \cap a) \cup \neg b \cup a \cup b = F,$$

what is impossible as $(\neg b \cup b) = T$ for all v. This contradiction proves that Γ that (6.5) holds and Γ is a **tautology**.

In order to define the axioms LA and the set of rules of inference of **RS** we need to introduce some definitions.

Literals

We form a special subset $\mathcal{F}' \subseteq \mathcal{F}$ of formulas, called a set of all *literals*, which is defined as follows.

$$LT = VAR \cup \{\neg a : a \in VAR\}. \tag{6.6}$$

The variables are called **positive literals** and the elements of the second set of the above union (6.6) are called **negative literals**. I.e propositional variables are called positive literals and the negation of a variable is called a negative literal, a variable or a negation of propositional variable is called a literal.

Indecomposable formulas and sequences

Literals are also called the indecomposable formulas. Now we form *finite sequences* out of formulas (and, as a special case, out of literals). We need to distinguish the sequences formed out of literals from the sequences formed out of other formulas, so we adopt the following notation.

We denote by

$$\Gamma^{'},\ \Delta^{'},\ \Sigma^{'},\ldots \quad \text{with indices if necessary,} \tag{6.7}$$

elements of $LT^* \subseteq \mathcal{F}^*$, i.e. $\Gamma^{'},\ \Delta^{'},\ \Sigma^{'}$ are finite sequences (empty included) formed out of **literals**. We call them **indecomposable sequences**.

We denote by

$$\Gamma,\ \Delta,\ \Sigma,\ldots \quad \text{with indices if necessary,} \tag{6.8}$$

the elements of $\mathcal{F}^*$, i.e. $\Gamma,\ \Delta,\ \Sigma$ denote finite sequences (empty included) formed out of elements of $\mathcal{F}$.

Logical Axioms LA

As the *logical axiom* of **RS** we adopt any sequence of literals which contains any propositional variable and its negation, i.e any sequence of the form

$$\Gamma^{'}_1, a, \Gamma^{'}_2, \neg a, \Gamma^{'}_3 \tag{6.9}$$

or of the form

$$\Gamma^{'}_1, \neg a, \Gamma^{'}_2, a, \Gamma^{'}_3 \tag{6.10}$$

for any variable $a \in VAR$ and any sequences $\Gamma^{'}_1, \Gamma^{'}_2, \Gamma^{'}_3 \in LT^*$ of literals.

Semantic Link
Consider axiom (6.9). Directly from the extension of the notion of tautology to **bf RS** (6.5), we have that for any truth assignments $v : VAR \longrightarrow \{T, F\}$, $v^*(\Gamma^{'}_1, \neg a, \Gamma^{'}_2, a, \Gamma^{'}_3) = v^*(\Gamma^{'}_1) \cup v^*(\neg a) \cup v^*(a) \cup v^*(\Gamma^{'}_2, \Gamma^{'}_3) = v^*(\Gamma^{'}_1) \cup T \cup v^*(\Gamma^{'}_2, \Gamma^{'}_3) = T$ The same applies to the axiom (6.10) We have thus proved the following.

Fact 6.1

Logical axioms of **RS** *are tautologies.*

Rules of inference $\mathcal{R}$

All rules of inference of **RS** are of the form

$$\frac{\Gamma_1}{\Gamma} \quad or \quad \frac{\Gamma_1 \ ; \ \Gamma_2}{\Gamma},$$

where $\Gamma_1,\ \Gamma_2,\ \Gamma \in \mathcal{F}^*$, i.e. $\Gamma_1,\ \Gamma_2,\ \Gamma$ are any finite sequences (**??**) of formulas. The sequences Γ_1, Γ_2 are called **premisses** and Γ is called a **conclusion** of the rule of inference.

Each rule of inference of **RS** introduces a new logical connective, or a negation of a logical connective. We denote a rule of inerence that introduces the logical connective $\circ$ in the conclusion sequent Γ by $(\circ)$. The notation $(\neg \circ)$ means that the negation of the logical connective $\circ$ is introduced in the conclusion sequence Γ. As our language contains the connectives: $\cap, \cup, \Rightarrow$ and $\neg$, so we we are going to define the following seven inference rules:

$$(\cup),\ (\neg \cup),\ (\cap),\ (\neg \cap),\ (\Rightarrow),\ (\neg \ \Rightarrow),\ \text{and}\ (\neg \ \neg). \tag{6.11}$$

We define formally the inference rules of **RS** as follows.

Disjunction rules

$$(\cup)\ \frac{\Gamma^{'}, A, B, \Delta}{\Gamma', (A \cup B), \Delta}, \qquad (\neg \cup)\ \frac{\Gamma^{'}, \neg A, \Delta \quad : \quad \Gamma^{'}, \neg B, \Delta}{\Gamma', \neg(A \cup B), \Delta}$$

Conjunction rules

$$(\cap)\ \frac{\Gamma^{'}, A, \Delta \quad ; \quad \Gamma^{'}, B, \Delta}{\Gamma', (A \cap B), \Delta}, \qquad (\neg \cap)\ \frac{\Gamma^{'}, \neg A, \neg B, \Delta}{\Gamma', \neg(A \cap B), \Delta}$$

Implication rules

$$(\Rightarrow)\ \frac{\Gamma^{'}, \neg A, B, \Delta}{\Gamma', (A \Rightarrow B), \Delta}, \qquad (\neg \ \Rightarrow)\ \frac{\Gamma^{'}, A, \Delta \quad : \quad \Gamma^{'}, \neg B, \Delta}{\Gamma', \neg(A \Rightarrow B), \Delta}$$

Negation rule

$$(\neg \ \neg)\ \frac{\Gamma^{'}, A, \Delta}{\Gamma', \neg\neg A, \Delta}$$

where $\Gamma^{'} \in LT^*, \Delta \in \mathcal{F}^*, A, B \in \mathcal{F}$.

The Proof System RS

Formally we define the proof system **RS** as follows.

$$\mathbf{RS} = (\mathcal{L}_{\{\neg,\Rightarrow,\cup,\cap\}},\ \mathcal{E},\ \ LA,\ \ \mathcal{R}), \tag{6.12}$$

where $\mathcal{E} = \{\Gamma :\ \ \Gamma \in \mathcal{F}^*\}$, LA contains logical axioms of the system defined by the schemas (6.9) and (6.10), $\mathcal{R}$ is the set of rules of inference:

$$\mathcal{R} = \{(\cup),\ (\neg\cup), (\cap), (\neg\cap), (\Rightarrow), (\neg \Rightarrow), (\neg\neg)\}$$

defined by (6.11).

By a **formal proof** of a sequence Γ in the proof system **RS** we understand any sequence

$$\Gamma_1,\ \Gamma_2, \ldots.\ \Gamma_n \tag{6.13}$$

of sequences of formulas (elements of $\mathcal{F}^*$, such that

$\Gamma_1 \in LA$, $\Gamma_n = \Gamma$,

and for all i $(1 \leq i \leq n)$ $\Gamma_i \in AL$, or Γ_i is a conclusion of one of the inference rules of **RS** with all its premisses placed in the sequence $\Gamma_1\Gamma_2....\Gamma_{i-1}$.

As the proof system under consideration is fixed, we will write, as usual,

$$\vdash \Gamma$$

instead of $\vdash_{\mathbf{RS}} \Gamma$ to denote that Γ has a formal proof in **RS**.

As the proofs in **RS** are sequences (definition of the formal proof) of sequences of formulas (definition of **RS**) we will not use "," to separate the steps of the proof, and write the formal proof as $\Gamma_1;\ \Gamma_2;\ \ldots.\ \Gamma_n$.

We write, however, the formal proofs in **RS** in a form of trees rather then in a form of sequences, ie. in a form of a tree, where *leafs* of the tree are axioms, *nodes* are sequences such that each sequence on the tree follows from the ones immediately preceding it by one of the rules. The *root* is a theorem. We picture, and write our tree-proofs with the node on the top, and leafs on the very bottom, instead of more common way, where the leafs are on the top and root is on the bottom of the tree. We adopt hence the following definition.

Definition 6.1 (Proof Tree)

By a proof tree, or **RS***-proof of* Γ *we understand a tree* $\mathbf{T}_\Gamma$ *of sequences satisfying the following conditions:*

1. The topmost sequence, i.e the root of $\mathbf{T}_\Gamma$ *is* Γ*,*

2. all leafs are axioms,

3. the nodes are sequences such that each sequence on the tree follows from the ones immediately preceding it by one of the rules.

We picture, and write our proof trees with the node on the top, and leafs on the very bottom, instead of more common way, where the leafs are on the top and root is on the bottom of the tree.

In particular cases we write our proof trees indicating additionally the name of the inference rule used at each step of the proof. For example, if the tree-proof of a given formula A from *axioms* was obtained by the subsequent use of the rules $(\cap), (\cup), (\cup), (\cap), (\cup), (\neg\neg)$, and $(\Rightarrow)$, we represent it as the following proof tree:

$$\begin{array}{c}
A\ (conclusion\ of\ (\Rightarrow)) \\
\mid (\Rightarrow) \\
conclusion\ of\ (\neg\neg) \\
\mid (\neg\neg) \\
conclusion\ of\ (\cup) \\
\mid (\cup) \\
conclusion\ of\ (\cap) \\
\bigwedge(\cap)
\end{array}$$

$$\begin{array}{cc}
conclusion\ of\ (\cap) & conclusion\ of\ (\cup) \\
\mid (\cup) & \mid (\cup) \\
axiom & conclusion\ of\ (\cap) \\
 & \bigwedge(\cap) \\
 & axiom \qquad axiom
\end{array}$$

The proof trees are often called **derivation trees** and we will use this notion as well. Remark that the proof trees don't represent a different *definition* of a formal proof. Trees represent a certain *visualization* of the proofs and any formal proof in any system can be represented in a tree form.

Example 6.1

Here is a proof tree in **RS** *of the de Morgan law* $(\neg(a \cap b) \Rightarrow (\neg a \cup \neg b))$.

$$\begin{array}{c}
(\neg(a \cap b) \Rightarrow (\neg a \cup \neg b)) \\
\mid (\Rightarrow) \\
\neg\neg(a \cap b), (\neg a \cup \neg b) \\
\mid (\neg\ \neg) \\
(a \cap b), (\neg a \cup \neg b) \\
\bigwedge(\cap)
\end{array}$$

$$a, (\neg a \cup \neg b) \qquad\qquad\qquad b, (\neg a \cup \neg b)$$

$$\mid (\cup) \qquad\qquad\qquad\qquad\qquad \mid (\cup)$$

$$a, \neg a, \neg b \qquad\qquad\qquad\qquad b, \neg a, \neg b$$

To obtain a "linear " formal proof of $(\neg(a \cap b) \Rightarrow (\neg a \cup \neg b))$ we just write down the tree as a sequence, starting from the leafs and going up (from left to right) to the root. The formal proof (with comments) thus obtained is:

$$a, \neg a, \neg b \quad (axiom)$$

$$b, \neg a, \neg b \quad (axiom)$$

$$a, (\neg a \cup \neg b) \quad (rule\ (\cup))$$

$$b, (\neg a \cup \neg b) \quad (rule\ (\cup))$$

$$(a \cap b), (\neg a \cup \neg b) \quad (rule(\cap))$$

$$\neg\neg(a \cap b), (\neg a \cup \neg b) \quad (rule\ (\neg\neg))$$

$$(\neg(a \cap b) \Rightarrow (\neg a \cup \neg b)) \quad (rule\ (\Rightarrow)).$$

6.2 Search for Proofs and Decomposition Trees

The main advantage of the Gentzen style proof systems lies not in a way we generate proofs in them, but in the way we can *search* for proofs in them. That such proof searches happens to be deterministic and automatic. Before we describe a general proof search procedure for **RS** let us look at few simple examples. Consider now a formula A of the form of another de Morgan law

$$(\neg(a \cup b) \Rightarrow (\neg a \cap \neg b)).$$

Obviously it should have a proof in **RS** as we want it to be, and will prove later to be complete. The search for the proof consists of building a certain tree. We call it a **decomposition tree**, to be defined formally later. We proceed as follows.

Observe that the *main connective* of A is $\Rightarrow$. So, if A *had* a proof in **RS** it would have come from the *only possible rule* used in its last step, namely the rule $(\Rightarrow)$ applied to its premiss, namely a sequence $\neg\neg(a \cup b), (\neg a \cap \neg b)$. So the last step in the proof of A would look as follows.

$$(\neg(a \cup b) \Rightarrow (\neg a \cap \neg b))$$

$$\mid (\Rightarrow)$$

$$\neg\neg(a \cup b), (\neg a \cap \neg b)$$

Now, if the sequence $\neg\neg(a \cup b), (\neg a \cap \neg b)$ (and hence also the formula A) had a proof in **RS** its *only step* at this stage would have been the application of the rule $(\neg\neg)$ to a sequence $(a \cup b), (\neg a \cap \neg b)$. So, if A had a proof, its last two steps would have been:

$$(\neg(a \cup b) \Rightarrow (\neg a \cap \neg b))$$

$$\mid (\Rightarrow)$$

$$\neg\neg(a \cup b), (\neg a \cap \neg b)$$

$$\mid (\neg\neg)$$

$$(a \cup b), (\neg a \cap \neg b)$$

Again, if the sequence $(a \cup b), (\neg a \cap \neg b)$ had a proof in **RS** its *only step* at this stage would have been the application of the rule $(\cup)$ to a sequence $a, b, (\neg a \cap \neg b)$. So, if A had a proof, its last three steps would have been as follows.

$$(\neg(a \cup b) \Rightarrow (\neg a \cap \neg b))$$

$$\mid (\Rightarrow)$$

$$\neg\neg(a \cup b), (\neg a \cap \neg b)$$

$$\mid (\neg\ \neg)$$

$$(a \cup b), (\neg a \cap \neg b)$$

$$\mid (\cup)$$

$$a, b, (\neg a \cap \neg b)$$

Now, if the sequence $a, b, (\neg a \cap \neg b)$ had a proof in **RS** its *only step* at this stage would have been the application of the rule $(\cap)$ to the sequences $a, b, \neg a$ and $a, b, \neg b$ as its left and right premisses, respectively. Both sequences are axioms and the following tree is a proof of A in **RS**.

$$(\neg(a \cup b) \Rightarrow (\neg a \cap \neg b))$$

$$\mid (\Rightarrow)$$

$$\neg\neg(a \cup b), (\neg a \cap \neg b)$$

$$\mid (\neg\neg)$$

$$(a \cup b), (\neg a \cap \neg b)$$

$$\mid (\cup)$$

$$a, b, (\neg a \cap \neg b)$$

$$\bigwedge (\cap)$$

$$a, b, \neg a \qquad\qquad a, b, \neg b$$

From the above proof tree of A we construct, if we want, its formal proof, written in a vertical manner, by writing the two axioms, which form the two premisses of the rule $(\cap)$ one above the other. All other sequences remain the same. I.e. the following sequence of elements of $\mathcal{F}^*$ is a **formal proof** of $(\neg(a \cup b) \Rightarrow (\neg a \cap \neg b))$ in **RS**.

$$a, b, \neg b$$

$$a, b, \neg a$$

$$a, b, (\neg a \cap \neg b)$$

$$(a \cup b), (\neg a \cap \neg b)$$

$$\neg\neg(a \cup b), (\neg a \cap \neg b)$$

$$(\neg(a \cup b) \Rightarrow (\neg a \cap \neg b))$$

Consider now a formula A of the form

$$(((a \Rightarrow b) \cap \neg c) \cup (a \Rightarrow c)).$$

Observe that the *main connective* of A is $\cup$. So, if A *had* a proof in **RS** it would have come from the *only possible rule* used in its last step, namely the rule $(\cup)$ applied to a sequence $((a \Rightarrow b) \cap \neg c), (a \Rightarrow c)$. So the last step in the proof of A would have been:

$$(((a \Rightarrow b) \cap \neg c) \cup (a \Rightarrow c))$$

$$\mid (\cup)$$

$$((a \Rightarrow b) \cap \neg c), (a \Rightarrow c)$$

Now, if the sequence $((a \Rightarrow b) \cap \neg c), (a \Rightarrow c)$ (and hence also had our formula) had a proof in **RS** its *only step* at this stage would have been the application of the rule $(\cap)$ to the sequences $(a \Rightarrow b), (a \Rightarrow c)$ and $\neg c, (a \Rightarrow c)$ as its left and right premisses, respectively. So, if A had a proof, its last two steps would have been:

$$(((a \Rightarrow b) \cap \neg c) \cup (a \Rightarrow c))$$

$$\mid (\cup)$$

$$((a \Rightarrow b) \cap \neg c),\ (a \Rightarrow c)$$

$$\bigwedge (\cap)$$

$$(a \Rightarrow b), (a \Rightarrow c) \qquad\qquad \neg c, (a \Rightarrow c)$$

Now, if the sequences $(a \Rightarrow b), (a \Rightarrow c)$ and $\neg c, (a \Rightarrow c)$ had proofs in **RS**, then their last, and the only steps would have been the the separate application of the rule $(\Rightarrow)$ to the sequences $\neg a, b, (a \Rightarrow c)$ and $\neg c, \neg a, c$, respectively. The sequence $\neg c, \neg a, c$ is an axiom, so we stop the search on this branch. The sequence $\neg a, b, (a \Rightarrow c)$ is not an axiom, so the search continues. In this case we can go one step further: if $\neg a, b, (a \Rightarrow c)$ had a proof it would have been only by the application of the rule $(\Rightarrow)$ to a sequence $\neg a, b, \neg a,\ c$ which is not an axiom and the **search ends**. The tree generated by this search is called a **decomposition tree** and is the following.

$$(((a \Rightarrow b) \cap \neg c) \cup (a \Rightarrow c))$$

$$\mid (\cup)$$

$$((a \Rightarrow b) \cap \neg c), (a \Rightarrow c)$$

$$\bigwedge(\cap)$$

$$\begin{array}{cc} (a \Rightarrow b), (a \Rightarrow c) & \neg c,\ (a \Rightarrow c) \\ \mid (\Rightarrow) & \mid (\Rightarrow) \\ \neg a, b,\ (a \Rightarrow c) & \neg c, \neg a, c \\ \mid (\Rightarrow) & \\ \neg a, b, \neg a, c & \end{array}$$

The tree generated by this search contains a **non-axiom leaf**, so by definition, it **is not a proof**.

Decomposition Rules and Trees

The process of searching for the proof of a formula A in **RS** consists of building a certain tree, called a **decomposition tree** whose root is the formula A, nodes correspond to sequences which are conclusions of certain rules (and those rules are well defined at each step by the way the node is built), and leafs are axioms or are sequences of a non- axiom literals. We prove that each formula A generates its *unique, finite* decomposition tree, $\mathbf{T}_A$ such that if all its leafs are axioms, the tree constitutes the proof of A in **RS**. If there is a leaf of $\mathbf{T}_A$ that *is not an axiom*, the tree is not a proof, moreover, the proof of A *does not exist.*

Before we give a proper definition of the proof search procedure by building a decomposition tree we list few important observations about the structure of the rules of the system **RS**.

Introduction of Connectives

The rules of **RS** are defined in such a way that each of them *introduces* a new logical connective, or a negation of a connective to a sequence in its domain (rules $(\cup), (\Rightarrow), (\cap)$) or a negation of a new logical connective (rules $(\neg\cup), (\neg\cap), (\neg\Rightarrow), (\neg\neg)$).

The rule $(\cup)$ introduces a new connective $\cup$ to a sequence Γ', A, B, Δ and it becomes, after the application of the rule, a sequence $\Gamma', (A\cup B), \Delta$. Hence a name for this rule is $(\cup)$.

The rule $(\neg\cup)$ introduces a negation of a connective, $\neg\cup$ by combining sequences $\Gamma', \neg A, \Delta$ and $\Gamma', \neg B, \Delta$ into one sequence (conclusion of the rule) $\Gamma', \neg(A\cup B), \Delta$. Hence a name for this rule is $(\neg\cup)$.

The same applies to all remaining rules of **RS**, hence their names say which connective, or the negation of which connective has been introduced by the particular rule.

Decomposition Rules

Building a proof search decomposition tree consists of using the inference rules in an inverse order; we transform them into rules that transform a conclusion into its premisses. We call such rules the **decomposition rules**. Here are all of **RS** decomposition rules.

Disjunction decomposition rules

$$(\cup)\ \frac{\Gamma', (A\cup B), \Delta}{\Gamma', A, B, \Delta}, \qquad (\neg\cup)\ \frac{\Gamma', \neg(A\cup B), \Delta}{\Gamma', \neg A, \Delta \quad : \quad \Gamma', \neg B, \Delta}$$

Conjunction decomposition rules

$$(\cap)\ \frac{\Gamma', (A\cap B), \Delta}{\Gamma', A, \Delta \quad ; \quad \Gamma', B, \Delta}, \qquad (\neg\cap)\ \frac{\Gamma', \neg(A\cap B), \Delta}{\Gamma', \neg A, \neg B, \Delta}$$

Implication decomposition rules

$$(\Rightarrow)\ \frac{\Gamma', (A\Rightarrow B), \Delta}{\Gamma', \neg A, B, \Delta}, \qquad (\neg\Rightarrow)\ \frac{\Gamma', \neg(A\Rightarrow B), \Delta}{\Gamma', A, \Delta \quad : \quad \Gamma', \neg B, \Delta}$$

Negation decomposition rule

$$(\neg\neg)\ \frac{\Gamma', \neg\neg A, \Delta}{\Gamma', A, \Delta}$$

where $\Gamma' \in LT^*,\ \Delta \in \mathcal{F}^*,\ A,\ B \in \mathcal{F}$.

We write the **decomposition rules** in a **visual tree form** as follows.

Tree Decomposition Rules

$(\cup)$ **rule**

$$\begin{array}{c}\Gamma^{'}, (A \cup B), \Delta \\ \mid (\cup) \\ \Gamma^{'}, A,\ B,\ \Delta\end{array}$$

$(\neg \cup)$ **rule**

$$\begin{array}{c}\Gamma^{'},\ \neg(A \cup B),\ \Delta \\ \bigwedge(\neg \cup) \\ \Gamma^{'},\ \neg A,\ \Delta \qquad \Gamma^{'},\ \neg B,\ \Delta\end{array}$$

$(\cap)$ **rule:**

$$\begin{array}{c}\Gamma^{'},\ (A \cap B),\ \Delta \\ \bigwedge(\cap) \\ \Gamma^{'},\ A,\ \Delta \qquad \Gamma^{'},\ B,\ \Delta\end{array}$$

$(\neg \cap)$ **rule:**

$$\begin{array}{c}\Gamma^{'}, \neg(A \cap B), \Delta \\ \mid (\neg \cap) \\ \Gamma^{'}, \neg A, \neg B, \Delta\end{array}$$

$(\Rightarrow)$ **rule:**

$$\begin{array}{c}\Gamma^{'},\ (A \Rightarrow B),\ \Delta \\ \mid (\cup) \\ \Gamma^{'}, \neg A, B, \Delta\end{array}$$

$(\neg \Rightarrow)$ **rule:**

$$\begin{array}{c} \Gamma^{'},\ \neg(A \Rightarrow B),\ \Delta \\ \bigwedge(\neg\ \Rightarrow) \\ \\ \Gamma^{'},\ A,\ \Delta \qquad\qquad \Gamma^{'},\ \neg B,\ \Delta \end{array}$$

$(\neg\ \neg)$ **rule:**

$$\begin{array}{c} \Gamma^{'},\ \neg\neg A,\ \Delta \\ \mid (\neg\ \neg) \\ \Gamma^{'},\ A,\ \Delta \end{array}$$

Observe that we use the same names for the inference and decomposition rules, as once the we have built the decomposition tree (with use of the decomposition rules) with all leaves being axioms, it constitutes a proof of A in **RS** with branches labeled by the proper inference rules.

Now we still need to introduce few useful definitions and observations.

Definition 6.2

1. A sequence Γ *is* **indecomposable** *if and only if* $\Gamma \in LT^{*}$.

2. A formula A *is* **decomposable** *if and only if* $A \in \mathcal{F} - LT$.

3. A sequence Γ *is* **decomposable** *if and only if it contains a decomposable formula.*

Directly from the definition 6.8 we have three simple, but important observations.

Fact 6.2

1. For any decomposable sequence Γ*, i.e. for any* $\Gamma \notin LT^{*}$ *there is* **exactly one** *decomposition rule that can be applied to it. This rule is determined by the first decomposable formula in* Γ*, and by the main connective of that formula.*

2. If the main connective of the first decomposable formula is $\cup, \cap$*, or* $\Rightarrow$*, then the decomposition rule determined by it is* $(\cup), (\cap)$*, or* $(\Rightarrow)$*, respectively.*

3. If the main connective of the first decomposable formula is $\neg$*, then the decomposition rule determined by it is determined by the second connective of the formula. If the second connective is* $\cup, \cap, \neg$*, or* $\Rightarrow$*, then corresponding decomposition rule is* $(\neg\cup), (\neg\cap), (\neg\neg)$ *and* $(\neg \Rightarrow)$.

Directly from the Fact 6.2 we we have the following lemma.

Lemma 6.1 (Unique Decomposition)

For any sequence $\Gamma \in \mathcal{F}^*$,

$\Gamma \in LT^*$ *or* Γ *is in the domain of only one of the* **RS** *Decomposition Rules.*

Now we define formally, for any formula $A \in \mathcal{F}$ and $\Gamma \in \mathcal{F}^*$ their decompositions trees. The decomposition tree for for the formula A is a particular case (one element sequence) of the tree for a sequence Γ.

Definition 6.3 (Decomposition Tree $\mathbf{T}_A$)

For each formula $A \in \mathcal{F}$*, its decomposition tree* $\mathbf{T}_A$ *is a tree build as follows.*

Step 1. The formula A *is the* **root** *of* $\mathbf{T}_A$ *and for any node* Γ *of the tree we follow the steps below.*

Step 2. If Γ *is* **indecomposable***, then* Γ *becomes a* **leaf** *of the tree.*

Step 3. If Γ *is* **decomposable***, then we traverse* Γ *from left to right to identify the first decomposable formula* B *and identify the* **decomposition rule** *determined by the main connective of* B*. In case of a one premisses rule we put is premise as a leaf; in case of a two premisses rule we put its left and right premisses as the left and right leaves respectively.*

Step 4. We **repeat** *Step 2 and Step 3 until we obtain only leaves.*

We now prove the following Decomposition Tree Theorem 6.1. This Theorem provides a crucial step in the proof of the Completeness Theorem for **RS**.

Theorem 6.1 (Decomposition Tree)

For any sequence $\Gamma \in \mathcal{F}^*$ *the following conditions hold.*

1. $\mathbf{T}_\Gamma$ *is finite and unique.*

2. $\mathbf{T}_\Gamma$ *is a proof of* Γ *in* **RS** *if and only if all its leafs are axioms.*

3. $\nvdash_{\mathbf{RS}}$ *if and only if* $\mathbf{T}_\Gamma$ *has a non- axiom leaf.*

Proof
The tree $\mathbf{T}_\Gamma$ is unique by the Unique Decomposition Lemma 6.1. It is finite because there is a finite number of logical connectives in Γ and all decomposition rules diminish the number of connectives. If the tree has a non- axiom leaf it is not a proof by definition. By the its uniqueness it also means that the **proof does not exist**.

Exercise 6.2

Construct a decomposition tree $\mathbf{T}_A$ *of the following formula A.*

$$A = ((a \cup b) \Rightarrow \neg a) \cup (\neg a \Rightarrow \neg c))$$

Solution
The formula A forms a one element decomposable sequence. The first decomposition rule used is determined by its main connective. We put a box around it, to make it more visible. The first and only rule applied is $(\cup)$ and we can write the first segment of our *decomposition tree* $\mathbf{T}_A$:

$$\mathbf{T}_A$$

$$((a \cup b) \Rightarrow \neg a)\boxed{\cup}(\neg a \Rightarrow \neg c))$$

$$\mid (\cup)$$

$$((a \cup b) \Rightarrow \neg a), (\neg a \Rightarrow \neg c)$$

Now we decompose the sequence $((a \cup b) \Rightarrow \neg a), (\neg a \Rightarrow \neg c))$. It is a decomposable sequence with the first, decomposable formula $((a \cup b) \Rightarrow \neg a)$. The next step of the construction of our decomposition tree is determined by its main connective $\Rightarrow$ (we put the box around it), hence the only rule determined by the sequence is $(\Rightarrow)$. The second stage of the decomposition tree is now as follows.

$$\mathbf{T}_A$$

$$((a \cup b) \Rightarrow \neg a)\boxed{\cup}(\neg a \Rightarrow \neg c))$$

$$\mid (\cup)$$

$$((a \cup b)\boxed{\Rightarrow}\neg a), (\neg a \Rightarrow \neg c)$$

$$\mid (\Rightarrow)$$

$$\neg(a \cup b), \neg a, (\neg a \Rightarrow \neg c)$$

The next sequence to decompose is the sequence $\neg(a \cup b), \neg a, (\neg a \Rightarrow \neg c)$.
The first decomposable formula is $\neg(a \cup b)$. Its main connective is $\neg$, so determine the appropriate decomposition rule we have to examine next connective, which is $\cup$.

The rule determine by this stage of decomposition is $(\neg \cup)$ and now the next stage of the decomposition tree $\mathbf{T}_A$ is as follows.

$$\mathbf{T}_A$$

$$((a \cup b) \Rightarrow \neg a)\boxed{\cup}(\neg a \Rightarrow \neg c))$$

$$\mid (\cup)$$

$$((a \cup b)\boxed{\Rightarrow}\neg a), (\neg a \Rightarrow \neg c)$$

$$\mid (\Rightarrow)$$

$$\boxed{\neg}(a\boxed{\cup}b), \neg a, (\neg a \Rightarrow \neg c)$$

$$\bigwedge(\neg\cup)$$

$$\neg a, \neg a, (\neg a \Rightarrow \neg c) \qquad\qquad \neg b, \neg a, (\neg a \Rightarrow \neg c)$$

Now we have two decomposable sequences: $\neg a, \neg a, (\neg a \Rightarrow \neg c)$ and $\neg b, \neg a, (\neg a \Rightarrow \neg c)$. They both happen to have the same first decomposable formula $(\neg a \Rightarrow \neg c)$. We decompose it simultenously and obtain the following:

$$\mathbf{T}_A$$

$$((a \cup b) \Rightarrow \neg a)\boxed{\cup}(\neg a \Rightarrow \neg c))$$

$$\mid (\cup)$$

$$((a \cup b)\boxed{\Rightarrow}\neg a), (\neg a \Rightarrow \neg c)$$

$$\mid (\Rightarrow)$$

$$\boxed{\neg}(a\boxed{\cup}b), \neg a, (\neg a \Rightarrow \neg c)$$

$$\bigwedge(\neg\cup)$$

$$\neg a, \neg a, (\neg a\boxed{\Rightarrow}\neg c) \qquad\qquad \neg b, \neg a, (\neg a\boxed{\Rightarrow}\neg c)$$

$$\mid (\Rightarrow) \qquad\qquad\qquad\qquad \mid (\Rightarrow)$$

$$\neg a, \neg a, \neg\neg a, \neg c \qquad\qquad \neg b, \neg a, \neg\neg a, \neg c$$

It is easy to see that we need only one more step to complete the process of constructing the unique decomposition tree of $\mathbf{T}_A$, namely, by decomposing the sequences: $\neg a, \neg a, \neg\neg a, \neg c$ and $\neg b, \neg a, \neg\neg a, \neg c$.

The complete decomposition tree $\mathbf{T}_A$ is:

$$\mathbf{T}_A$$

$$((a \cup b) \Rightarrow \neg a)\boxed{\cup}(\neg a \Rightarrow \neg c))$$

$$\mid (\cup)$$

$$((a \cup b)\boxed{\Rightarrow}\neg a), (\neg a \Rightarrow \neg c)$$

$$\mid (\Rightarrow)$$

$$\boxed{\neg}(a\boxed{\cup}b), \neg a, (\neg a \Rightarrow \neg c)$$

$$\bigwedge(\neg\cup)$$

$$\neg a, \neg a, (\neg a\boxed{\Rightarrow}\neg c) \qquad \neg b, \neg a, (\neg a\boxed{\Rightarrow}\neg c)$$

$$\mid (\Rightarrow) \qquad \mid (\Rightarrow)$$

$$\neg a, \neg a, \boxed{\neg\neg}a, \neg c \qquad \neg b, \neg a, \boxed{\neg\neg}a, \neg c$$

$$\mid (\neg\neg) \qquad \mid (\neg\neg)$$

$$\neg a, \neg a, a, \neg c \qquad \neg b, \neg a, a, \neg c$$

All leafs are axioms, the tree represents a proof of A in **RS**

Exercise 6.3

Prove that the formula $A = (((a \Rightarrow b) \cap \neg c) \cup (a \Rightarrow c))$ *is not provable in* **RS**, *i.e.*

$$\nvdash_{\mathbf{RS}} (((a \Rightarrow b) \cap \neg c) \cup (a \Rightarrow c)).$$

Solution
We construct the formula A decomposition tree as follows.

$$\mathbf{T}_A$$

$$(((a \Rightarrow b) \cap \neg c) \cup (a \Rightarrow c))$$

$$\mid (\cup)$$

$$((a \Rightarrow b) \cap \neg c), (a \Rightarrow c)$$

$$\bigwedge(\cap)$$

$$(a \Rightarrow b), (a \Rightarrow c) \qquad \neg c, (a \Rightarrow c)$$

$$\mid (\Rightarrow) \qquad \mid (\Rightarrow)$$

$$\neg a, b, (a \Rightarrow c) \qquad \neg c, \neg a, c$$

$$\mid (\Rightarrow)$$

$$\neg a, b, \neg a, c$$

The above tree $\mathbf{T}_A$ is unique by the Theorem 6.1 and represents the only possible search for proof of the formula $A = ((a \Rightarrow b) \cap \neg c) \cup (a \Rightarrow c))$ in **RS**. It has a non-axiom leaf, hence by Theorem 6.1 the proof of A in **RS** does not exists.

6.3 Strong Soundness and Completeness

Our main goal is to prove the Completeness Theorem for **RS**. The proof of completeness presented here is due to Rasiowa and Sikorski, as is the proof system **RS**. Their proof, and the proof system was inverted for the classical predicate logic and was published in 1961. We present their predicate logic proof system **QRS** together with the proof of its completeness in chapter 10. Both completeness proofs, for propositional **RS** and predicate **QRS** proof systems, are constructive as they are based on a direct construction of a counter model for any unprovable formula. The construction of a counter model for a formula A uses directly its decomposition tree $\mathbf{T}_A$. We call such constructed model a *counter model determined* by the tree $\mathbf{T}_A$. Both proofs relay heavily of the notion of a *strong soundness.* We define it now, adopting Chapter 4 general definition to our semantics.

Definition 6.4 (Strongly Sound Rules)

Given a proof system $S = (\mathcal{L}, \mathcal{E}, LA, \mathcal{R})$ An inference rule $r \in \mathcal{R}$ of the form

$$(r) \qquad \frac{P_1 \;\; ; \;\; P_2 \;\; ; \;\; .... \;\; ; \;\; P_m}{C}$$

is **strongly sound** *(undef classical semantics) if the following condition holds for all $v : VAR \longrightarrow \{T, F\}$*

$$v \;\models \{P_1, P_2, .P_m\} \quad \textit{if and only if} \quad v \;\models\; C. \tag{6.14}$$

We say it less formally that a rule (r) is **strongly sound** *if the conjunction of its premisses is logically equivalent with the conclusion, i.e.*

$$P_1 \cap P_2 \cap \;\ldots\; \cap P_m \equiv C. \tag{6.15}$$

Definition 6.5 (Strongly Sound S)

A proof system $S = (\mathcal{L}, \mathcal{E}, LA, \mathcal{R})$ is strongly sound (undef classical semantics) if and only if all logical axioms LA are tautologies and all its rules of inference $r \in \mathcal{R}$ are **strongly sound***.*

Theorem 6.2 (Strong Soundness)

The proof system **RS** *(6.18) is* **strongly sound***.*

Proof
The logical axioms (6.9), (6.10) are tautologies by Fact **??**. We prove as an example the **strong soundness** of two of inference rules: $(\cup)$ and $(\neg\cup)$. Proofs for all other rules follow the same patterns and are left as an exercise. By definition 6.4 of strong soundness we have to show the condition (8.77). Written

formally it says that we have to show that that if $P_1,\ P_2$ are premisses of a given rule and C is its conclusion, then for all truth assignments $v: VAR \longrightarrow \{T, F\}$, $v^*(P_1) = v^*(C)$ in case of one premiss rule, and $v^*(P_1) \cap v^*(P_2) = v^*(C)$, in case of a two premisses rule. Consider the rule $(\cup)$.

$$(\cup)\ \ \frac{\Gamma^{'}, A, B, \Delta}{\Gamma', (A \cup B), \Delta}.$$

By the definition:

$v^*(\Gamma^{'}, A, B, \Delta) = v^*(\delta_{\{\Gamma', A, B, \Delta\}}) = v^*(\Gamma^{'}) \cup v^*(A) \cup v^*(B) \cup v^*(\Delta) = v^*(\Gamma^{'}) \cup v^*(A \cup B) \cup v^*(\Delta) = v^*(\delta_{\{\Gamma', (A\cup B), \Delta\}}) = v^*(\Gamma^{'}, (A \cup B), \Delta)$.

Consider the rule $(\neg\cup)$.

$$(\neg\cup)\ \ \frac{\Gamma^{'}, \neg A, \Delta \quad : \quad \Gamma^{'}, \neg B, \Delta}{\Gamma', \neg(A \cup B), \Delta}.$$

By the definition:

$v^*(\Gamma^{'}, \neg A, \Delta) \cap v^*(\Gamma^{'}, \neg B, \Delta) = (v^*(\Gamma^{'}) \cup v^*(\neg A) \cup v^*(\Delta)) \cap (v^*(\Gamma^{'}) \cup v^*(\neg B) \cup v^*(\Delta)) = (v^*(\Gamma^{'}, \Delta) \cup v^*(\neg A)) \cap (v^*(\Gamma^{'}, \Delta) \cup v^*(\neg B)) =$ by distributivity $= (v^*(\Gamma^{'}, \Delta) \cup (v^*(\neg A) \cap v^*(\neg B)) = v^*(\Gamma^{'}) \cup v^*(\Delta) \cup (v^*(\neg A \cap \neg B)) =$ by the logical equivalence of $(\neg A \cap \neg B)$ and $\neg(A \cup B) = v^*(\delta_{\{\Gamma', \neg(A\cup B), \Delta\}} = v^*(\Gamma^{'}, \neg(A \cup B), \Delta))$.

Observe that the strong soundness implies soundness (not only by name!), hence we have also proved the following.

Theorem 6.3 (Soundness for RS)

For any $\Gamma \in \mathcal{F}^*$,
if $\vdash_{\boldsymbol{RS}} \Gamma$, *then* $\models \Gamma$. *In particular, for any* $A \in \mathcal{F}$, *if* $\vdash_{\boldsymbol{RS}} A$, *then* $\models A$.

We have just proved (Theorem 6.2) that all the rules of inference of **RS** of are strongly sound, i.e. $C \equiv P$ and $C \equiv P_1 \cap P_2$. The strong soundness of the rules means that if *at least* one of premisses of a rule is *false*, so is its conclusion. Hence given a formula A, such that its $\mathbf{T}_A$ has a branch ending with a non-axiom leaf. By Strong Soundness Theorem 6.2, any v that make this non-axiom leaf false also falsifies all sequences on that branch, and hence falsifies the formula A. This means that any v, such that it falsifies a non-axiom leaf is a **counter-model** for A. We have hence proved the following.

Theorem 6.4 (Counter Model)

Given a formula $A \in \mathcal{F}$ such that its decomposition tree $\mathbf{T}_A$ contains a **non-axiom** *leaf L_A. Any truth assignment v that falsifies the non-axiom leaf L_A is a counter model for A. We call it a* **counter-model** *for A* **determined** *by the decomposition tree $\mathbf{T}_A$.*

Here is a simple example explaining how the construction of a counter-model determined by the decomposition tree of a works. Consider a tree

$$
\begin{array}{c}
\mathbf{T}_A \\[6pt]
(((a \Rightarrow b) \cap \neg c) \cup (a \Rightarrow c)) \\[6pt]
\mid (\cup) \\[6pt]
((a \Rightarrow b) \cap \neg c), (a \Rightarrow c) \\[6pt]
\bigwedge(\cap)
\end{array}
$$

$$
\begin{array}{ccc}
(a \Rightarrow b), (a \Rightarrow c) & \qquad\qquad & \neg c, (a \Rightarrow c) \\
\mid (\Rightarrow) & & \mid (\Rightarrow) \\
\neg a, b, (a \Rightarrow c) & & \neg c, \neg a, c \\
\mid (\Rightarrow) & & \\
\neg a, b, \neg a, c & &
\end{array}
$$

The tree $\mathbf{T}_A$ has a non-axiom leaf $L_A: \ \neg a, \ b, \neg a, \ c$. The truth assignment $v: \ VAR \longrightarrow \{T, F\}$ that falsifies the leaf $\neg a, \ b, \neg a, \ c$ must be such that

$v^*(\neg a, b, \neg a, c) = v^*(\neg a) \cup v^*(b) \cup v^*(\neg a) \cup v^*(c) = \neg v(a) \cup v(b) \cup \neg v(a) \cup v(c) = F$, i.e. v must be such that $\neg v(a) \cup v(b) \cup \neg v(a) \cup v(c) = F$. We hence get that $v(a) = T, \ \ v(b) = F, \quad v(c) = F$. By the Counter Model Theorem 6.4, the truth assignment v determined by the non-axiom leaf also falsifies the formula A, i.e. we proved that v is a counter model for A and

$$\not\models (((a \Rightarrow b) \cap \neg c) \cup (a \Rightarrow c)).$$

The Counter Model Theorem 6.4, says that the logical value **F** determined by the evaluation a non-axiom leaf "climbs" the decomposition tree. We picture it as follows.

$$
\begin{array}{c}
\mathbf{T}_A \\[6pt]
(((a \Rightarrow b) \cap \neg c) \cup (a \Rightarrow c)) = \mathbf{F} \\[6pt]
\mid (\cup) \\[6pt]
((a \Rightarrow b) \cap \neg c), (a \Rightarrow c) = \mathbf{F} \\[6pt]
\bigwedge(\cap)
\end{array}
$$

$$\begin{array}{ccc}
(a \Rightarrow b), (a \Rightarrow c) = \mathbf{F} & \qquad & \neg c, (a \Rightarrow c) \\
\mid (\Rightarrow) & & \mid (\Rightarrow) \\
\neg a, b, (a \Rightarrow c) = \mathbf{F} & & \neg c, \neg a, c \\
\mid (\Rightarrow) & & \textit{axiom} \\
\neg a, b, \neg a, c = \mathbf{F} & &
\end{array}$$

Observe that the same counter model construction applies to any other non-axiom leaf of $\mathbf{T}_A$, if exists. The other non-axiom leaf of $\mathbf{T}_A$ defines another evaluation of the non- axiom leaf to $\mathbf{F}$ that also "climbs the tree" and hence defines another counter- model for a formula A. By Counter Model 6.4 all possible restricted counter-models for A are those determined by its all non-axioms leaves.

In our case the tree $\mathbf{T}_A$ has only one non-axiom leaf, and hence the formula $(((a \Rightarrow b) \cap \neg c) \cup (a \Rightarrow c))$ only only one restricted counter model.

Our main goal is to prove the Completeness Theorem for **RS**. We prove first the Completeness Theorem for formulas $A \in \mathcal{F}$ and then we generalize it to any sequences $\Gamma \in \mathcal{F}^*$.

Theorem 6.5 (Completeness Theorem)

For any formula $A \in \mathcal{F}$,
1. $\vdash_{\mathbf{RS}} A$ *if and only if* $\models A$, *and for any* $\Gamma \in \mathcal{F}^*$,
2. $\vdash_{\mathbf{RS}} \Gamma$ *if and only if* $\models \Gamma$.

Proof
Case 1. We have already proved the Soundness Theorem 6.3, so we need to prove only the completeness part of it, namely to prove the implication:

$$\text{if} \quad \models A, \quad \text{then} \ \vdash_{\mathbf{RS}} A. \tag{6.16}$$

We prove instead of the opposite implication:

$$\text{if} \quad \not\vdash_{\mathbf{RS}} A \ \text{ then } \ \not\models A. \tag{6.17}$$

Assume that A is any formula is such that $\not\vdash_{\mathbf{RS}} A$. By the Decomposition Tree Theorem 6.1 the tree $\mathbf{T}_A$ contains a non-axiom leaf L_A. We use the non-axiom leaf L_A **to define** a truth assignment $v : VAR \longrightarrow \{T, F\}$ which **falsifies** it as follows:

$$v(a) = \begin{cases} F & \text{if a appears in } L_A \\ T & \text{if } \neg a \text{ appears in } L_A \\ \text{any value} & \text{if a does not appear in } L_A \end{cases}$$

By the Counter Model Theorem 6.4we have that v also **falsifies** the formula A. We proved that

$$\not\models A$$

and it ends the proof of the case 1.

Case 2. Assume that $\Gamma \in \mathcal{F}^*$ is any sequence such that $\nvdash_{\mathbf{RS}} \Gamma$. But obviously, $\vdash_{\mathbf{RS}} \Gamma$ if and only if $\vdash_{\mathbf{RS}} \delta_\Gamma$, where δ_Γ is any disjunction of all formulas of Γ. So $\nvdash_{\mathbf{RS}} \Gamma$ if and only if $\nvdash_{\mathbf{RS}} \delta_\Gamma$ and by already proven Case 1, $\not\models\ \delta_\Gamma$ what is obviously equivalent to $\not\models\ \Gamma$. This ends the proof of Case 2 and Completeness Theorem.

6.4 Proof Systems RS1 and RS2

We present here a two modifications of the system **RS** as an exercise of importance of paying close attention to the syntax. Proof systems might be, as all presented here **RS** type systems are, semantically identical, nevertheless they are very different proof systems.

Language of **RS1** is the same as the language of **RS**, i.e.

$$\mathcal{L} = \mathcal{L}_{\{\neg, \Rightarrow, \cup, \cap\}}.$$

Rules of inference of **RS1** operate as rules of **RS** on **finite sequences** of formulas and we adopt

$$\mathcal{E} = \mathcal{F}^*$$

as the set of expressions of **RS1** . We denote them, as before, by $\Gamma, \Delta, \Sigma \ldots$, with indices if necessary.

The proof system **RS1** contains seven inference rules, denoted by the same symbols as the rules of **RS**, namely $(\cup)$, $(\neg\cup)$, $(\cap)$, $(\neg\cap)$, $(\Rightarrow)$, $(\neg \Rightarrow)$, $(\neg\neg)$.

The inference rules of **RS1** are quite similar to the rules of **RS** Look at them carefully to see where lies the difference.

Reminder Any propositional variable, or a negation of propositional variable is called a **literal**. The set $LT = VAR \cup \{\neg a : \quad a \in VAR\}$ is called a set of all propositional literals. The variables are called positive literals. Negations of variables are called negative literals. We denote, as before, by $\Gamma^{'}$, $\Delta^{'}$, $\Sigma^{'}, \ldots$ finite sequences (empty included) formed out of literals. We adopt all logical axiom of **RS** as the axioms of **RS1**, i.e. logical axioms LA of **RS1** are:

$$\Gamma^{'}_1,\ a,\ \Gamma^{'}_2,\ \neg a,\ \Gamma^{'}_3,$$

$$\Gamma^{'}_1,\ \neg a,\ \Gamma^{'}_2,\ a,\ \Gamma^{'}_3$$

where $a \in VAR$ is any propositional variable.

We define the inference rules of **RS1** as follows.

Disjunction rules

$$(\cup)\ \frac{\Gamma,\ A,\ B,\ \Delta^{'}}{\Gamma,\ (A\cup B),\ \Delta^{'}},\qquad (\neg\cup)\ \frac{\Gamma,\ \neg A,\ \Delta^{'}\quad :\quad \Gamma,\ \neg B,\ \Delta^{'}}{\Gamma,\ \neg\ (A\cup B),\ \Delta^{'}},$$

Conjunction rules

$$(\cap)\ \frac{\Gamma,\ A,\ \Delta^{'}\quad ;\quad \Gamma,\ B,\ \Delta^{'}}{\Gamma,\ (A\cap B),\ \Delta^{'}},\qquad (\neg\cap)\ \frac{\Gamma,\ \neg A,\ \neg B,\ \Delta^{'}}{\Gamma^{'},\neg(A\cap B),\Delta},$$

Implication rules

$$(\Rightarrow)\ \frac{\Gamma,\ \neg A,\ B,\Delta^{'}}{\Gamma,\ (A\Rightarrow B),\ \Delta^{'}},\qquad (\neg\ \Rightarrow)\ \frac{\Gamma,\ A,\ \Delta^{'}\quad :\quad \Gamma,\ \neg B,\ \Delta^{'}}{\Gamma,\ \neg\ (A\Rightarrow B),\ \Delta^{'}},$$

Negation rule

$$(\neg\ \neg)\ \frac{\Gamma,\ A,\ \Delta^{'}}{\Gamma,\ \neg\ \neg\ A,\ \Delta^{'}}$$

where $\Gamma\in\mathcal{F}^*,\ \Delta^{'}\in LT^*,\ A,\ B\ \in\mathcal{F}$.

Proof System RS1

Formally we define the proof system **RS1** as follows.

$$\mathbf{RS1}=(\mathcal{L}_{\{\neg,\Rightarrow,\cup,\cap\}},\ \mathcal{E},\ \ LA,\ \ \mathcal{R}),\tag{6.18}$$

where $\mathcal{E}=\{\Gamma:\ \ \Gamma\in\mathcal{F}^*\}$, LA is the set logical axioms and $\mathcal{R}$ is the set of rules of inference defined above.

Exercise 6.4

Construct a proof in **RS1** *of a formula*

$$A=(\neg(a\cap b)\Rightarrow(\neg a\cup\neg b)).$$

Solution
The decomposition tree below is a **proof** of A in **RS1** as all its leaves are axioms.

$$\mathbf{T}_A$$

$$(\neg(a\cap b)\Rightarrow(\neg a\cup\neg b))$$

$$\mid(\Rightarrow)$$

$$(\neg\neg(a\cap b),(\neg a\cup\neg b)$$

$$\mid (\cup)$$

$$\neg\neg(a \cap b), \neg a, \neg b$$

$$\mid (\neg\neg)$$

$$(a \cap b), \neg a, \neg b$$

$$\bigwedge(\cap)$$

$$a, \neg a, \neg b \qquad\qquad\qquad b, \neg a, \neg b$$

Exercise 6.5

Prove that **RS1** *is strongly sound.*

Solution
Observe that the system **RS1** is obtained from **RS** by changing the sequence $\Gamma^{'}$ into Γ and the sequence Δ into $\Delta^{'}$ in all of the rules of inference of **RS**. These changes do not influence the essence of proof of **strong soundness** of the rules of **RS**. One has just to replace the sequence $\Gamma^{'}$ by Γ and the sequence Δ by $\Delta^{'}$ in the proof of strong soundness of each rule of **RS** to obtain a corresponding proof of strong soundness of corresponding rule of **RS1**. We do it, for example for the rule $(\cup)$ of **RS1**. Consider the rule $(\cup)$ of **RS1**:

$$(\cup) \quad \frac{\Gamma,\ A,\ B,\ \Delta^{'}}{\Gamma,\ (A \cup B),\ \Delta^{'}}.$$

We evaluate:

$$v^{*}(\Gamma, A, B, \Delta^{'}) = v^{*}(\delta_{\{\Gamma, A, B, \Delta^{'}\}}) = v^{*}(\Gamma) \cup v^{*}(A) \cup v^{*}(B) \cup v^{*}(\Delta^{'})$$

$$= v^{*}(\Gamma) \cup\ v^{*}(A \cup B) \cup v^{*}(\Delta^{'}) = v^{*}(\delta_{\{\Gamma, (A \cup B), \Delta^{'}\}}) = v^{*}(\Gamma, (A \cup B), \Delta^{'}).$$

Exercise 6.6

Define in your own words, for any formula $A \in \mathcal{F}$ the decomposition tree $\mathbf{T}_A$ in **RS1***.*

Solution
The definition of the decomposition tree $\mathbf{T}_A$ is again, it its essence similar to the one for **RS** except for the changes which reflect the differences in the corresponding rules of inference. We follow now the following steps.
Step 1 Decompose A using a rule defined by its main connective.
Step 2 Traverse resulting sequence Γ on the new node of the tree from **right** to **left** and find the first decomposable formula.
Step 3 Repeat **Step 1** and **Step 2** until there is no more decomposable formulas. **End** of tree construction.

Exercise 6.7

Prove the following **Completeness Theorem** *for* **RS1**.

Theorem 6.6

For any formula $A \in \mathcal{F}$,
1. $\vdash_{\mathbf{RS1}} A$ if and only if $\models A$, and for any $\Gamma \in \mathcal{F}^*$,
2. $\vdash_{\mathbf{RS1}} \Gamma$ if and only if $\models \Gamma$.

Solution Part 1.
Observe that directly from the definition of the uniqueness of the decomposition tree $\mathbf{T}_A$ we have that the following holds.

Fact 6.3

The decomposition tree $\mathbf{T}_A$ is a **proof** *if and only if all leaves are axioms and the proof does not exist otherwise, i.e. we have that $\not\vdash_{\mathbf{RS1}} A$ if and only if there is a non- axiom leaf on $\mathbf{T}_A$.*

The Fact 6.3 together with strong soundness of the rules of inference of **RS1** justify the correctness of construction of a counter-model generated by a the a non- axiom leaf and hence the correctness of the following **proof** of the Completeness Theorem.

We prove, as we did in case of **RS** the implication

$$\text{if} \quad \not\vdash_{\mathbf{RS1}} A \ \text{ then } \ \not\models A.$$

Assume that A is any formula such that $\not\vdash_{\mathbf{RS1}} A$. By the Fact 6.3 the decomposition tree $\mathbf{T}_A$ contains a non-axiom leaf L_A. We use the non-axiom leaf L_A and **define** a truth assignment v which falsifies A, as follows:

$$v(a) = \begin{cases} F & \text{if a appears in } L_A \\ T & \text{if } \neg a \text{ appears in } L_A \\ \text{any value} & \text{if a does not appear in } L_A. \end{cases}$$

This proves, by the strong soundness of **RS1**, that $\not\models A$.

The proof of Part 2. is identical to the proof in **RS** case.

$$\textbf{Proof System RS2} \tag{6.19}$$

System **RS2** is a proof system obtained from **RS** by changing the sequences Γ' into Γ in **all of the rules** of inference of **RS**. The **logical axioms** LA remain the same. Observe that now the decomposition tree may not be unique

Exercise 6.8

Construct ***two*** *decomposition trees in* **RS2** *of the formula*

$$A = (\neg(\neg a \Rightarrow (a \cap \neg b)) \Rightarrow (\neg a \cap (\neg a \cup \neg b))).$$

Solution
Here are two out of many more decomposition trees.

$$\mathbf{T1}_A$$

$$(\neg(\neg a => (a \cap \neg b)) => (\neg a \cap (\neg a \cup \neg b)))$$
$$\mid (\Rightarrow)$$
$$\neg\neg(\neg a => (a \cap \neg b)), (\neg a \cap (\neg a \cup \neg b))$$
$$\mid (\neg\neg)$$
$$(\neg a => (a \cap \neg b)), (\neg a \cap (\neg a \cup \neg b))$$
$$\mid (\Rightarrow)$$
$$\neg\neg a, (a \cap \neg b), (\neg a \cap (\neg a \cup \neg b))$$
$$\mid (\neg\neg)$$
$$a, (a \cap \neg b), (\neg a \cap (\neg a \cup \neg b))$$
$$\bigwedge (\cap)$$

$$\begin{array}{cc} a, a, (\neg a \cap (\neg a \cup \neg b)) & a, \neg b, (\neg a \cap (\neg a \cup \neg b)) \\ \bigwedge (\cap) & \bigwedge (\cap) \end{array}$$

$$\begin{array}{cccc} a, a.\neg a, (\neg a \cup \neg b) & a, a, (\neg a \cup \neg b) & a, \neg b, \neg a & a, \neg b, (\neg a \cup \neg b) \\ \mid (\cup) & \mid (\cup) & \textit{axiom} & \mid (\cup) \\ a, a.\neg a, \neg a, \neg b & a, a, \neg a, \neg b & & a, \neg b, \neg a, \neg b \\ \textit{axiom} & \textit{axiom} & & \textit{axiom} \end{array}$$

The other tree is:

$$\mathbf{T2}_A$$

$$(\neg(\neg a => (a \cap \neg b)) => (\neg a \cap (\neg a \cup \neg b)))$$
$$\mid (\Rightarrow)$$
$$\neg\neg(\neg a => (a \cap \neg b)), (\neg a \cap (\neg a \cup \neg b))$$
$$\mid (\neg\neg)$$
$$(\neg a => (a \cap \neg b)), (\neg a \cap (\neg a \cup \neg b))$$
$$\bigwedge (\cap)$$

$$(\neg a => (a \cap \neg b)), \neg a$$
$$\mid (\Rightarrow)$$
$$(\neg\neg a, (a \cap \neg b)), \neg a$$
$$\mid (\neg\neg)$$
$$a, (a \cap \neg b), \neg a$$
$$\bigwedge(\cap)$$
$$a, a, \neg a \qquad\qquad a, \neg b, \neg a$$
$$\mathit{axiom} \qquad\qquad \mathit{axiom}$$

$$(\neg a => (a \cap \neg b)), (\neg a \cup \neg b)$$
$$\mid (\cup)$$
$$(\neg a => (a \cap \neg b)), \neg a, \neg b$$
$$\mid (\Rightarrow)$$
$$(\neg\neg a, (a \cap \neg b), \neg a, \neg b$$
$$\mid (\neg\neg)$$
$$a, (a \cap \neg b), \neg a, \neg b$$
$$\bigwedge(\cap)$$
$$a, a, \neg a, \neg b \qquad a, \neg b, \neg a, \neg b$$
$$\mathit{axiom} \qquad\qquad \mathit{axiom}$$

Exercise 6.9

Explain why the system **RS2** *is* **strongly sound**. *You can use the strong soundness of the system* **RS**.

Solution
The only one difference between **RS** and **RS2** is that in **RS2** each inference rule has at the beginning a sequence of any formulas, not only of literals, as in **RS**. So there are many ways to apply the *decomposition rules* while constructing the decomposition tree, but it does not affect strong soundness, since for all rules of **RS2** premisses and conclusions are still logically equivalent as they were in **RS** .

Consider, for example, **RS2** rule

$$(\cup)\ \ \frac{\Gamma, A, B, \Delta}{\Gamma, (A \cup B), \Delta}.$$

We evaluate $v^*(\Gamma, A, B, \Delta) = v^*(\Gamma) \cup v^*(A) \cup v^*(B) \cup v^*(\Delta) = v^*(\Gamma) \cup v^*(A \cup B) \cup v^*(\Delta) = v^*(\Gamma, (A \cup B), \Delta)$. Similarly, as in **RS**, we show all other rules of **RS2** to be strongly sound, thus **RS2** is sound.

Exercise 6.10

Define shortly, in your own words, for any formula A, its decomposition tree $\mathbf{T}_A$ *in* **RS2**. *Justify why your definition is correct. Show that in* **RS2** *the decomposition tree for some formula A may not be unique.*

Solution
Given a formula A. The decomposition tree $\mathbf{T}_A$ can be defined as follows. It has

A as a **root**. For each **node**, if there is a rule of **RS2** which conclusion has the same form as node sequence, i.e. there is a decomposition rule to be applied, then the node has children that are premises of the rule. If the **node** consists only of literals (i.e. there is no decomposition rule to be applied), then it does not have any children. The last statement define a termination condition for the tree $\mathbf{T}_A$.

This definition defines correctly the decomposition tree $\mathbf{T}_A$ as it identifies and uses appropriate decomposition rules. Since all rules of inference of **RS2** have a sequence Γ instead of Γ' as it was in **RS**, the choice of the decomposition rule for a node may not unique. For example consider a node $(a => b),\ (b \cup a)$. Γ in the **RS2** rules may be a sequence of formulas, not only literals, so for the node $(a => b),\ (b \cup a)$ we can choose as a decomposition rule either $(=>)$ or $(\cup)$. This leads to a non-unique tree.

Exercise 6.11

Prove the following Completeness Theorem for **RS2**.

Theorem 6.7

For any formula $A \in \mathcal{F}$,
1. $\vdash_{\mathbf{RS2}} A$ *if and only if* $\models A$, *and for any* $\Gamma \in \mathcal{F}^*$,
2. $\vdash_{\mathbf{RS2}} \Gamma$ *if and only if* $\models \Gamma$.

Solution
We need to prove the completeness part only, as the Soundness has been already proved, i.e. we have to prove the implication (Part 1): for any formula A,

$$\text{if} \quad \not\vdash_{RS2} A \text{ then } \quad \not\models A.$$

Assume $\not\vdash_{RS2} A$. Then **every** decomposition tree of A has at least one non-axiom leaf. Otherwise, there would exist a tree with all axiom leaves and it would be a proof for A. Let $\mathcal{T}_A$ be a set of all decomposition trees of A. We choose an arbitrary $T_A \in \mathcal{T}_A$ with at least one non-axiom leaf L_A. We use the non-axiom leaf L_A to define a truth assignment v which falsifies A, as follows:

$$v(a) = \begin{cases} F & \text{if a appears in } L_A \\ T & \text{if } \neg a \text{ appears in } L_A \\ \text{any value} & \text{if a does not appear in } L_A \end{cases}$$

The value for a sequence that corresponds to the leaf in is F. Since, because of the strong soundness F "climbs" the tree, we found a counter-model for A. This proves that $\not\models A$. Part 2. proof is identical to the proof in **RS** case.

Exercise 6.12

Write a procedure $TREE_A$ such that for any formula A of **RS2** *it produces its* **unique** *decomposition tree.*

Solution
Here is the procedure.

```
Procedure $TREE_A$(Formula A, Tree T)
{
    $B = ChoseLeftMostFormula(A)$ // Choose the left most formula that is not a literal
    $c = MainConnective(B)$ // Find the main connective of B
    $R = FindRule(c)$// Find the rule which conclusion that has this connective
    $P = Premises(R)$// Get the premises for this rule
    $AddToTree(A, P)$// add premises as children of A to the tree
    For all p in P // go through all premises
        $TREE_A(p, T)$ // build subtrees for each premiss
}
```

Exercise 6.13

Prove ***completeness*** *of your Procedure $TREE_A$.*

Solution
Procedure $TREE_A$ provides a unique tree, since it always chooses the most left indecomposable formula for a choice of a decomposition rule and there is only one such rule. This procedure is equivalent to **RS** system, since with the decomposition rules of **RS** the most left decomposable formula is always chosen. The proof **RS** system is complete, thus this Procedure $TREE_A$ is **complete**.

6.5 Gentzen Sequent Systems GL, G, LK

Gentzen proof systems **GL** and **G** for the classical propositional logic presented here are inspired by and all are versions of the original (1934) Gentzen system **LK**. Their axioms, the rules of inference of the proof system considered here operate, as the original Gentzen system **LK**, on expressions called by Gentzen *sequents*, hence the name Gentzen sequent proof systems, called also Gentzen sequent calculus, or sequents calculus. The original system **LK** is presented and discussed in detail in section 6.7.

6.5.1 Gentzen Sequent Systems GL and G

The system **GL** presented here is the most similar in its structure to the system **RS** (6.18) and hence is the first to be considered. It admits a constructive proof of the Completeness Theorem that is very similar to the proof of the Completeness Theorem for the system **RS**.

Language of GL

We adopt a propositional language $\mathcal{L} = \mathcal{L}_{\{\cup,\cap,\Rightarrow,\neg\}}$ with the set of formulas denoted by $\mathcal{F}$ and we add a new symbol $\longrightarrow$ called a Gentzen arrow, to it. It means we consider formally a new language $\mathcal{L}_1 = \mathcal{L}_{\{\cup,\cap,\Rightarrow,\neg\}} \cup \{\longrightarrow\}$. As the next step we build expressions called *sequents* out of $\mathcal{L}_1$. The sequents are built out of finite sequences (empty included) of formulas, i.e. elements of $\mathcal{F}^*$ of $\mathcal{L}_{\{\cup,\cap,\Rightarrow,\neg\}}$, and the additional sign $\longrightarrow$.

We denote , as in the **RS** system, the finite sequences of formulas of of $\mathcal{L}_{\{\cup,\cap,\Rightarrow,\neg\}}$ by Greek capital letters

$$\Gamma, \Delta, \Sigma, \ \ \ldots,$$

with indices if necessary. We define a sequent as follows.

Definition 6.6 (Sequent)

For any $\Gamma, \Delta \in \mathcal{F}^*$*, the expression*

$$\Gamma \longrightarrow \Delta$$

is called a **sequent***.* Γ *is called the* **antecedent** *of the sequent,* Δ *is called the* **succedent***, and each formula in* Γ *and* Δ *is called a* **sequent-formula***.*

Intuitively, a sequent $A_1, ..., A_n \longrightarrow B_1, ..., B_m$ (where $n, m \geq 1$) means: *if* $A_1 \cap ... \cap A_n$ *then* $B_1 \cup ... \cup B_m$. The sequent $A_1, ..., A_n \longrightarrow$ (where $n \geq 1$) means *that* $A_1 \cap ... \cap A_n$ *yields a contradiction.* The sequent $\longrightarrow B_1, ..., B_m$ (where $m \geq 1$) means *that* $B_1 \cup ... \cup B_m$ *is true.* The empty sequent $\longrightarrow$ means *a contradiction.*

Given non empty sequences Γ, Δ, we denote by σ_Γ any **conjunction** of all formulas of Γ, and by δ_Δ any **disjunction** of all formulas of Δ. The intuitive semantics for a sequent $\Gamma \longrightarrow \Delta$ (where Γ, Δ are nonempty) is hence that it is logically equivalent to the formula $(\sigma_\Gamma \Rightarrow \delta_\Delta)$, i.e.

$$\Gamma \longrightarrow \Delta \ \equiv \ (\sigma_\Gamma \Rightarrow \delta_\Delta).$$

Formal semantics

Formally, let $v : VAR \longrightarrow \{T, F\}$ be a truth assignment, v^* its extension to the set of formulas $\mathcal{F}$. We extend v^* to the set

$$SQ = \{ \ \Gamma \longrightarrow \Delta : \ \Gamma, \Delta \in \mathcal{F}^* \ \} \tag{6.20}$$

of all sequents as follows.

Definition 6.7 *For any sequent* $\Gamma \longrightarrow \Delta \in SQ$,

$$v^*(\Gamma \longrightarrow \Delta) \ = \ v^*(\sigma_\Gamma) \Rightarrow v^*(\delta_\Delta).$$

In the case when $\Gamma = \emptyset$ we define: $v^*(\longrightarrow \Delta) \ = \ T \ \Rightarrow v^*(\delta_\Delta)$. In the case $\Delta = \emptyset$ we define $v^*(\Gamma \longrightarrow \ ) \ = \ v^*(\sigma_\Gamma) \Rightarrow \ F$.

Model

The sequent $\Gamma \ \longrightarrow \ \Delta$ is *satisfiable* if there is a truth assignment $v : VAR \longrightarrow \{T, F\}$ such that $v^*(\Gamma \longrightarrow \Delta) = T$. Such a truth assignment is called a *model* for $\Gamma \longrightarrow \Delta$. We write

$$v \models \ \Gamma \ \longrightarrow \ \Delta.$$

Counter- model

The sequent $\Gamma \ \longrightarrow \ \Delta$ is *falsifiable* if there is a truth assignment v, such that $v^*(\Gamma \ \longrightarrow \ \Delta) = F$. In this case v is called a *counter-model* for $\Gamma \ \longrightarrow \ \Delta$ and we write it as

$$v \not\models \ \Gamma \ \longrightarrow \ \Delta.$$

Tautology

The sequent $\Gamma \ \longrightarrow \ \Delta$ is a *tautology* if $v^*(\Gamma \ \longrightarrow \ \Delta) = T$ for all truth assignments $v : VAR \longrightarrow \{T, F\}$ and we write

$$\models \ \Gamma \ \longrightarrow \ \Delta.$$

Example 6.2

Let $\Gamma \longrightarrow \ \Delta$ *be a sequent*

$$a, (b \cap a) \ \longrightarrow \ \neg b, (b \Rightarrow a).$$

Any truth assignment v*, such that* $v(a) = T$ *and* $v(b) = T$ *is a model for* $\Gamma \longrightarrow \ \Delta$*, i.e.*

$$\models \ a, (b \cap a) \ \longrightarrow \ \neg b, (b \Rightarrow a).$$

We verify it by performing the following computation.

$v^*(a, (b \cap a) \ \longrightarrow \ \neg b, (b \Rightarrow a)) \ = \ v^*(\sigma_{\{a,(b \cap a)\}}) \Rightarrow v^*(\delta_{\{\neg b,(b \Rightarrow a)\}}) \ = \ v(a) \cap (v(b) \cap v(a)) \Rightarrow \neg v(b) \cup (v(b) \Rightarrow v(a)) \ = \ T \cap TcapT \Rightarrow \neg T \cup (T \Rightarrow T) \ = \ T \Rightarrow (F \cup T) \ = \ T \Rightarrow T \ = \ T.$

Observe that the only v for which $v^*(\Gamma) = v^*(a, (b \cap a) = T$ is the above $v(a) = T$ and $v(b) = T$ that is a model for $\Gamma \longrightarrow \ \Delta$. Hence it is impossible to find v which would falsify it, what proves that $\Gamma \longrightarrow \ \Delta$ is a tautology, i.e.

$$\models \ a, (b \cap a) \ \longrightarrow \ \neg b, (b \Rightarrow a).$$

The Proof System GL

The rules of inference of **GL** are of the form:

$$\frac{P_1}{C} \quad or \quad \frac{P_1 \ ; \ P_2}{C},$$

where P_1, P_2 and C are sequents. P_1, P_2 are called premisses and C is called the conclusion of the rule of inference. Each rule of inference introduces a new logical connective to the antecedent or to the succedent of the conclusion sequent. We denote the rule that introduces the logical connective $\circ$ to the antecedent of the conclusion sequent P by $(\circ \rightarrow)$. The notation $(\rightarrow \circ)$ means that the logical connective is introduced to the succedent of the conclusion sequent P.

As our language contains the connectives: $\cap, \cup, \Rightarrow$ and $\neg$, we are going to adopt the following inference rules: $(\cap \rightarrow)$ and $(\rightarrow \cap)$, $(\cup \rightarrow)$ and $(\rightarrow \cup)$, $(\Rightarrow\rightarrow)$ and $(\rightarrow\Rightarrow)$, and finally, $(\neg \rightarrow)$ and $(\rightarrow \neg)$.

Definition 6.8

Finite sequences formed out of **positive literals** *i.e. out of propositional variables are called* **indecomposable**. *We denote them as before by*

$$\Gamma^{'}, \ \Delta^{'}, \ \ldots$$

with indices, if necessary.

A **sequent** *is* **indecomposable** *if it is formed out of indecomposable sequences, i.e. is of the form*

$$\Gamma^{'} \ \longrightarrow \ \Delta^{'}$$

for any $\Gamma^{'}, \Delta^{'} \in VAR^*$.

Remark that now the symbols $\Gamma^{'}, \ \Delta^{'}, \ \ldots$ denote sequences of **variables** (positive literals), and not sequences of literals as in (**RS**.

Axioms of GL

As the axioms of **GL** we adopt any indecomposable sequent sequent which contains a positive literal a (variable) that appears on both sides of the sequent arrow $\longrightarrow$, i.e any sequent of the form

$$\Gamma'_1, \ a, \ \Gamma'_2 \ \longrightarrow \ \Delta'_1, \ a, \ \Delta'_2, \tag{6.21}$$

for any $a \in VAR$ and any sequences $\Gamma'_1, \Gamma'_2, \Delta'_1, \Delta'_2 \in VAR^*$.

Semantic Link

Consider axiom (6.21). Directly from the Definition 6.7 of semantics for **bf GL** we evaluate (in shorthand notation), for any truth assignments $v : VAR \longrightarrow \{T, F\}$, the following (in shorthand notation).

$$v^*(\Gamma'_1,\ a,\ \Gamma'_2 \ \longrightarrow\ \Delta'_1, a, \Delta'_2) =$$

$$(\sigma_{\Gamma'_1} \cap\ a\ \cap \sigma_{\Gamma'_2})\ \Rightarrow\ (\delta_{\Delta'_1} \cup a \cup \delta_{\Delta'_2}) = T.$$

The evaluation is correct because $\models (((A \cap\ a) \cap B) \Rightarrow (C \cup a) \cup D)))$. We have thus proved the following.

Fact 6.4

Logical axioms of **GL** *are tautologies.*

Inference Rules of GL (6.22)

We adopt the following rules of inference.

Conjunction rules

$$(\cap \rightarrow)\ \frac{\Gamma', A, B, \Gamma \ \longrightarrow\ \Delta'}{\Gamma', (A \cap B), \Gamma \ \longrightarrow\ \Delta'}, \qquad (\rightarrow \cap)\ \frac{\Gamma \ \longrightarrow\ \Delta, A, \Delta' \ ;\ \Gamma \ \longrightarrow\ \Delta, B, \Delta'}{\Gamma \ \longrightarrow\ \Delta, (A \cap B), \Delta'},$$

Disjunction rules

$$(\rightarrow \cup)\ \frac{\Gamma \ \longrightarrow\ \Delta, A, B, \Delta'}{\Gamma \ \longrightarrow\ \Delta, (A \cup B), \Delta'}, \qquad (\cup \rightarrow)\ \frac{\Gamma', A, \Gamma \ \longrightarrow\ \Delta' \ ;\ \Gamma', B, \Gamma \ \longrightarrow\ \Delta'}{\Gamma', (A \cup B), \Gamma \ \longrightarrow\ \Delta'},$$

Implication rules

$$(\rightarrow\Rightarrow)\ \frac{\Gamma', A, \Gamma \ \longrightarrow\ \Delta, B, \Delta'}{\Gamma', \Gamma \ \longrightarrow\ \Delta, (A \Rightarrow B), \Delta'},$$

$$(\Rightarrow\rightarrow)\ \frac{\Gamma', \Gamma \ \longrightarrow\ \Delta, A, \Delta' \ ;\ \Gamma', B, \Gamma \ \longrightarrow\ \Delta, \Delta'}{\Gamma', (A \Rightarrow B), \Gamma \ \longrightarrow\ \Delta, \Delta'},$$

Negation rules

$$(\neg \rightarrow)\ \frac{\Gamma', \Gamma \ \longrightarrow\ \Delta, A, \Delta'}{\Gamma', \neg A, \Gamma \ \longrightarrow\ \Delta, \Delta'}, \qquad (\rightarrow \neg)\ \frac{\Gamma', A, \Gamma \ \longrightarrow\ \Delta, \Delta'}{\Gamma', \Gamma \ \longrightarrow\ \Delta, \neg A, \Delta'}.$$

Formally we define:

$$\mathbf{GL} = (\mathcal{L},\ SQ,\ LA,\ (\cup), (\neg\cup), (\cap), (\neg\cap), (\Rightarrow), (\neg \Rightarrow), (\neg\neg)), \qquad (6.23)$$

where $SQ = \{\ \Gamma \longrightarrow \Delta :\ \ \Gamma, \Delta \in \mathcal{F}^*\ \},\ (\cup), (\neg\cup), (\cap), (\neg\cap), (\Rightarrow), (\neg\Rightarrow), (\neg\neg)$ are the inference rules defined above and AL are the logical axioms of the system defined by the schema (6.21).

We define the notion of a bf formal proof in **GL** as in any proof system, i.e., by a formal proof of a sequent $\Gamma \longrightarrow \Delta$ in the proof system **GL** we understand any sequence

$$\Gamma_1 \longrightarrow \Delta_1,\ \Gamma_2 \longrightarrow \Delta_2,\ ....,\ \Gamma_n \longrightarrow \Delta_n$$

of sequents, such that $\Gamma_1 \longrightarrow \Delta_1 \in AL$, $\Gamma_n \longrightarrow \Delta_n = \Gamma \longrightarrow \Delta$, and for all i $(1 < i \leq n)$ $\Gamma_i \longrightarrow \Delta_i \in AL$, or $\Gamma_i \longrightarrow \Delta_i$ is a conclusion of one of the inference rules of **GL** with all its premisses placed in the sequence $\Gamma_1 \longrightarrow \Delta_1,\ ....\Gamma_{i-1} \longrightarrow \Delta_{i-1}$.

We write, as usual, $\vdash_{\mathbf{GL}} \Gamma \longrightarrow \Delta$ to denote that $\Gamma \longrightarrow \Delta$ has a formal proof in **GL**, or we write simply $\vdash \Gamma \longrightarrow \Delta$ when the system **GL** is fixed.

We say that a formula $A \in \mathcal{F}$, has a proof in **GL** and denote it by $\vdash_{\mathbf{GL}} A$ if the sequent $\longrightarrow A$ has a proof in **GL**, i.e. we define:

$$\vdash_{\mathbf{GL}} A \text{ if and only if } \vdash_{\mathbf{GL}} \ \longrightarrow\ A. \tag{6.24}$$

We write, however, the formal proofs in **GL** in a form of proof trees rather then in a form of sequences of sequents.

Proof trees

A proof tree $\Gamma \longrightarrow \Delta$ is a tree $\mathbf{T}_{\Gamma\longrightarrow\Delta}$ satisfying the following conditions:

1. The topmost sequent, i.e the **root** of $\mathbf{T}_{\Gamma\longrightarrow\Delta}$ is $\Gamma \longrightarrow \Delta$.

2. All **leaves** are axioms.

3. The **nodes** are sequents such that each sequent on the tree follows from the ones immediately preceding it by one of the rules.

We picture, and write our proof-trees, as we did in case of **RS** type systems, with the node on the top, and leafs on the very bottom, instead of more common way, where the leaves are on the top and root is on the bottom of the tree. We also write the proof- trees indicating additionally the name of the inference rule used at each step of the proof.

Here is a tree- proof of the de Morgan law $(\neg(a \cap b) \Rightarrow (\neg a \cup \neg b))$.

$$\longrightarrow\ (\neg(a \cap b) \Rightarrow (\neg a \cup \neg b))$$

$$\mid (\rightarrow\Rightarrow)$$

$$\begin{array}{c}
\neg(a \cap b) \ \longrightarrow \ (\neg a \cup \neg b) \\
\mid (\rightarrow \cup) \\
\neg(a \cap b) \ \longrightarrow \ \neg a, \neg b \\
\mid (\rightarrow \neg) \\
b, \neg(a \cap b) \ \longrightarrow \ \neg a \\
\mid (\rightarrow \neg) \\
b, a, \neg(a \cap b) \ \longrightarrow \\
\mid (\neg \rightarrow) \\
b, a \longrightarrow \ (a \cap b) \\
\bigwedge(\rightarrow \cap)
\end{array}$$

$$\begin{array}{ccc}
b, a \longrightarrow \ a & \qquad\qquad & b, a \longrightarrow \ b
\end{array}$$

Remark 6.1

The proof search in **GL** *(to be defined by the decomposition tree) results are not always unique; one formula (sequent) can have many proofs.*

Here is another proof in **GL** of the de Morgan Law.

$$\begin{array}{c}
\longrightarrow \ (\neg(a \cap b) \Rightarrow (\neg a \cup \neg b)) \\
\mid (\rightarrow \Rightarrow) \\
\neg(a \cap b) \ \longrightarrow \ (\neg a \cup \neg b) \\
\mid (\rightarrow \cup) \\
\neg(a \cap b) \ \longrightarrow \ \neg a, \neg b \\
\mid (\rightarrow \neg) \\
b, \neg(a \cap b) \ \longrightarrow \ \neg a \\
\mid (\neg \longrightarrow) \\
b \ \longrightarrow \ \neg a, (a \cap b) \\
\bigwedge(\rightarrow \cap)
\end{array}$$

$$\begin{array}{ccc}
b \longrightarrow \ \neg a, a & \qquad\qquad & b \longrightarrow \ \neg a, b \\
\mid (\longrightarrow \neg) & & \mid (\longrightarrow \neg) \\
b, a \longrightarrow \ a & & b, a \longrightarrow \ b
\end{array}$$

The process of searching for proofs of a formula A in **GL** consists, as in the **RS** type systems, of building decomposition trees. Their construction is similar to the one defined for **RS** type systems and is described intuitiively as follows.

We take a **root** of a decomposition tree T_A a sequent $\longrightarrow A$. For each **node**, if there is a rule of **GL** which conclusion has the same form as the node sequent, then the node has children that are premises of the rule. If the node consists only of an indecomposable sequent (built out of variables only), then it does not have any children. This is a termination condition for the decomposition tree.

We prove that each formula A generates a *finite set* $\mathcal{T}_A$ of decomposition trees, such that the following holds. If there exist a tree $T_A \in \mathcal{T}_A$ whose all leaves are axioms, then tree T_A constitutes a proof of A in **GL**. If all trees in $\mathcal{T}_A$ have at least one non-axiom leaf, the proof of A does not exist.

The first step in formally defining a notion of a decomposition tree consists of transforming the inference rules of **GL**, as we did in the case of the **RS** type systems, into corresponding *decomposition rules.*

Decomposition rules of GL

Building a proof search decomposition tree consists of using the inference rules in an inverse order; we transform the inference rules into decomposition rules by reversing the role of conclusion and its premisses. We call such rules the decomposition rules. Here are all of **GL** decomposition rules.

Conjunction decomposition rules

$$(\cap \rightarrow)\ \frac{\Gamma', (A \cap B), \Gamma \ \longrightarrow\ \Delta'}{\Gamma', A, B, \Gamma \ \longrightarrow\ \Delta'}, \qquad (\rightarrow \cap)\ \frac{\Gamma \ \longrightarrow\ \Delta, (A \cap B), \Delta'}{\Gamma \ \longrightarrow\ \Delta, A, \Delta' \ ; \ \Gamma \ \longrightarrow\ \Delta, B, \Delta'},$$

Disjunction decomposition rules

$$(\rightarrow \cup)\ \frac{\Gamma \ \longrightarrow\ \Delta, (A \cup B), \Delta'}{\Gamma \ \longrightarrow\ \Delta, A, B, \Delta'}, \qquad (\cup \rightarrow)\ \frac{\Gamma', (A \cup B), \Gamma \ \longrightarrow\ \Delta'}{\Gamma', A, \Gamma \ \longrightarrow\ \Delta' \ ; \ \Gamma', B, \Gamma \ \longrightarrow\ \Delta'},$$

Implication decomposition rules

$$(\rightarrow\Rightarrow)\ \frac{\Gamma', \Gamma \ \longrightarrow\ \Delta, (A \Rightarrow B), \Delta'}{\Gamma', A, \Gamma \ \longrightarrow\ \Delta, B, \Delta'},$$

$$(\Rightarrow\rightarrow)\ \frac{\Gamma', (A \Rightarrow B), \Gamma \ \longrightarrow\ \Delta, \Delta'}{\Gamma', \Gamma \ \longrightarrow\ \Delta, A, \Delta' \ ; \ \Gamma', B, \Gamma \ \longrightarrow\ \Delta, \Delta'},$$

Negation decomposition rules

$$(\neg \rightarrow)\ \frac{\Gamma^{'}, \neg A, \Gamma \ \longrightarrow\ \Delta, \Delta^{'}}{\Gamma', \Gamma \ \longrightarrow\ \Delta, A, \Delta'}, \qquad (\rightarrow \neg)\ \frac{\Gamma^{'}, \Gamma \ \longrightarrow\ \Delta, \neg A, \Delta^{'}}{\Gamma', A, \Gamma \ \longrightarrow\ \Delta, \Delta'}.$$

We write the decomposition rules in a visual tree form as follows.

$(\rightarrow \cup)$ **rule**

$$\begin{array}{c} \Gamma \ \longrightarrow\ \Delta, (A \cup B), \Delta^{'} \\ \mid (\rightarrow \cup) \\ \Gamma \ \longrightarrow\ \Delta, A, B, \Delta^{'} \end{array}$$

$(\cup \rightarrow)$ **rule**

$$\begin{array}{c} \Gamma^{'}, (A \cup B), \Gamma \ \longrightarrow\ \Delta^{'} \\ \bigwedge(\cup \rightarrow) \\ \Gamma^{'}, A, \Gamma \ \longrightarrow\ \Delta^{'} \qquad \Gamma^{'}, B, \Gamma \ \longrightarrow\ \Delta^{'} \end{array}$$

$(\rightarrow \cap)$ **rule**

$$\begin{array}{c} \Gamma \ \longrightarrow\ \Delta, (A \cap B), \Delta^{'} \\ \bigwedge(\rightarrow \cap) \\ \Gamma \ \longrightarrow\ \Delta, A, \Delta^{'} \qquad \Gamma \ \rightarrow\ \Delta, B, \Delta^{'} \end{array}$$

$(\cap \rightarrow)$ **rule**

$$\begin{array}{c} \Gamma^{'}, (A \cap B), \Gamma \ \longrightarrow\ \Delta^{'} \\ \mid (\cap \rightarrow) \\ \Gamma^{'}, A, B, \Gamma \ \longrightarrow\ \Delta^{'} \end{array}$$

$(\rightarrow\Rightarrow)$ **rule**

$$\begin{array}{c} \Gamma^{'}, \Gamma \ \longrightarrow\ \Delta, (A \Rightarrow B), \Delta^{'} \\ \mid (\rightarrow\Rightarrow) \\ \Gamma^{'}, A, \Gamma \ \longrightarrow\ \Delta, B, \Delta^{'} \end{array}$$

$(\Rightarrow\rightarrow)$ **rule**

$$\Gamma^{'}, (A \Rightarrow B), \Gamma \longrightarrow \Delta, \Delta^{'}$$

$$\bigwedge(\Rightarrow\rightarrow)$$

$$\Gamma^{'}, \Gamma \longrightarrow \Delta, A, \Delta^{'} \qquad \Gamma^{'}, B, \Gamma \longrightarrow \Delta, \Delta^{'}$$

$(\neg \rightarrow)$ **rule**

$$\Gamma^{'}, \neg A, \Gamma \longrightarrow \Delta, \Delta^{'}$$

$$\mid (\neg \rightarrow)$$

$$\Gamma^{'}, \Gamma \longrightarrow \Delta, A, \Delta^{'}$$

$(\longrightarrow \neg)$ **rule**

$$\Gamma^{'}, \Gamma \longrightarrow \Delta, \neg A, \Delta^{'}$$

$$\mid (\neg \rightarrow)$$

$$\Gamma^{'}, A, \Gamma \longrightarrow \Delta, \Delta^{'}$$

Observe that we use the same names for the inference and decomposition rules, as once the we have built a decomposition tree (with use of the decomposition rules) with all leaves being axioms, it constitutes a proof of A in **GL** with branches labeled by the proper inference rules.

We have already defined (definition 6.8) indecomposable sequence as any sequence $\Gamma^{'} \longrightarrow \Delta^{'}$ when $\Gamma^{'}, \Delta^{'} \in VAR^{*}$. In particular, a formula that is not a positive literal (propositional variable) is called a **decomposable formula**, and a sequent $\Gamma \longrightarrow \Delta$ where either Γ or Δ contains a decomposable formula is called a **decomposable sequent**.

By inspecting the domain of the rules we can see that at most two rules could apply for any given decomposable sequent $\Gamma \longrightarrow \Delta$.

For any decomposable sequent, at most two decomposition rules can be applied to it. This rule is determined by the first decomposable formula in Γ when we traverse it from left to right, and by the main connective of that formula, or by the first decomposable formula in Δ when we traverse it from the right to left, and by the main connective of that formula. We hence are now ready to define a decomposition tree.

Decomposition Tree $\mathbf{T}_{\rightarrow A}$

For each formula $A \in \mathcal{F}$, a decomposition tree $\mathbf{T}_{\rightarrow A}$ is a tree build as follows.

Step 1. The sequent $\longrightarrow A$ is the **root** of $\mathbf{T}_{\rightarrow A}$ and for any node $\Gamma \longrightarrow \Delta$ of the tree we follow the steps below.

Step 2. If $\Gamma \longrightarrow \Delta$ is indecomposable, then $\Gamma \longrightarrow \Delta$ becomes a **leaf** of the tree.

Step 3. If $\Gamma \longrightarrow \Delta$ is decomposable, then we pick a decomposition rule that applies by matching the sequent of the current node with the domain of the decomposition rule. To do so we proceed as follows.

1. We traverse Γ from left to right to find the first decomposable formula. Its main connective $\circ$ identifies a possible decomposition rule $(\circ \longrightarrow)$. Then we check if this decomposition rule applies. If it does we put its conclusions (conclusion) as leaves (leaf).

2. We traverse Δ from right to left to find the first decomposable formula. Its main connective $\circ$ identifies a possible decomposition rule $(\longrightarrow\ \circ)$. Then we check if this decomposition rule applies. If it does we put its conclusions (conclusion) as leaves (leaf). 3. If 1. and 2. applies we choose one of the rules.

Step 4. We repeat steps 2 and 3 until we obtain only leaves.

Observation 6.1

The decomposable $\Gamma \longrightarrow \Delta$ is always in the domain in one of the decomposition rules $(\circ \longrightarrow)$, $(\longrightarrow\ \circ)$, or in the domain of both. Hence the tree $\mathbf{T}_{\rightarrow A}$ may not be unique and all possible choices of 3. give all possible decomposition trees.

We generalize the definition of $\mathbf{T}_{\rightarrow A}$ to the decomposition tree $\mathbf{T}_{\Sigma \rightarrow \Lambda}$ of any sequent $\Sigma \longrightarrow \Lambda \in SQ$ as follows.

Decomposition Tree $\mathbf{T}_{\Sigma \longrightarrow \Lambda}$

Step 1. The sequent $\Sigma \longrightarrow \Lambda$ is the **root** of $\mathbf{T}_{\Sigma \longrightarrow \Lambda}$, and for any node $\Gamma \longrightarrow \Delta$ of the tree we follow the steps below.

Step 2. If $\Gamma \longrightarrow \Delta$ is indecomposable, then $\Gamma \longrightarrow \Delta$ becomes a **leaf** of the tree. **Step 3.** and **Step 4.** are the same as in the above definition of the tree $\mathbf{T}_{\rightarrow A}$.

Exercise 6.14

Prove, by constructing a proper **decomposition tree** *that*

$$\vdash_{\mathbf{GL}}((\neg a \Rightarrow b) \Rightarrow (\neg b \Rightarrow a)).$$

Solution
By definition,we have that

$\vdash_{\mathbf{GL}}((\neg a \Rightarrow b) \Rightarrow (\neg b \Rightarrow a))$ if and only if $\vdash_{\mathbf{GL}} \longrightarrow ((\neg a \Rightarrow b) \Rightarrow (\neg b \Rightarrow a))$.

We construct a decomposition tree as follows.

$$\mathbf{T}_{\rightarrow A}$$

$$\longrightarrow ((\neg a \Rightarrow b) \Rightarrow (\neg b \Rightarrow a))$$

$$\mid (\rightarrow\Rightarrow)$$

$$(\neg a \Rightarrow b) \longrightarrow (\neg b \Rightarrow a)$$

$$\mid (\rightarrow\Rightarrow)$$

$$\neg b, (\neg a \Rightarrow b) \longrightarrow a$$

$$\mid (\rightarrow \neg)$$

$$(\neg a \Rightarrow b) \longrightarrow b, a$$

$$\bigwedge(\Rightarrow\longrightarrow)$$

$$\begin{array}{ccc}
\longrightarrow\ \neg a, b, a & \qquad\qquad & b \longrightarrow b, a \\
\mid (\rightarrow \neg) & & axiom \\
a\ \longrightarrow\ b, a & & \\
axiom & &
\end{array}$$

All leaves of the tree are axioms, hence it constitutes a **proof** in **GL**.

Exercise 6.15

Prove, by constructing proper **decomposition trees** *that*

$$\nvdash_{\mathbf{GL}}((a \Rightarrow b) \Rightarrow (\neg b \Rightarrow a)).$$

Solution
Observe that for some formulas A, their decomposition tree $\mathbf{T}_{\rightarrow A}$ in **GL** may not be unique. Hence we have to construct all possible decomposition trees to see that none of them is a proof, i.e. to see that each of them has a non axiom leaf. We construct the decomposition trees for $\longrightarrow A$ as follows.

$$\mathbf{T1}_{\rightarrow A}$$

$$\longrightarrow ((a \Rightarrow b) \Rightarrow (\neg b \Rightarrow a))$$

(one choice)

$$\mid (\longrightarrow\Rightarrow)$$

$$(a \Rightarrow b) \;\longrightarrow\; (\neg b \Rightarrow a)$$

(first of two choices)

$$\mid (\longrightarrow\Rightarrow)$$

$$\neg b, (a \Rightarrow b) \;\longrightarrow\; a$$

(one choice)

$$\mid (\neg \longrightarrow)$$

$$(a \Rightarrow b) \;\longrightarrow\; b, a$$

(one choice)

$$\bigwedge(\Rightarrow\longrightarrow)$$

$$\begin{array}{cc} \longrightarrow\; a, b, a & b \longrightarrow b, a \\ non - axiom & axiom \end{array}$$

The tree contains a non- axiom leaf, hence it is **not a proof**. We have one more tree to construct.

$$\mathbf{T2}_{\rightarrow A}$$

$$\longrightarrow ((a \Rightarrow b) \Rightarrow (\neg b \Rightarrow a))$$

$$\mid (\longrightarrow\Rightarrow)$$

$$(a \Rightarrow b) \;\longrightarrow\; (\neg b \Rightarrow a)$$

$(second\ choice)$

$$\bigwedge(\Rightarrow\longrightarrow)$$

$$\begin{array}{cc} \longrightarrow\; (\neg b \Rightarrow a), a & b \longrightarrow\; (\neg b \Rightarrow a) \\ \mid (\longrightarrow\Rightarrow) & \mid (\longrightarrow\Rightarrow) \\ \neg b \longrightarrow\; a, a & b, \neg b \longrightarrow\; a \\ \mid (\neg \longrightarrow) & \mid (\neg \longrightarrow) \\ \longrightarrow\; b, a, a & b \;\longrightarrow\; a, b \\ non - axiom & axiom \end{array}$$

All possible trees end with a non-axiom leaf. It proves that

$$\nvdash_{\mathbf{GL}} ((a \Rightarrow b) \Rightarrow (\neg b \Rightarrow a)).$$

Exercise 6.16

Does the tree below constitute a proof in **GL***?*

$$\mathbf{T}_{\rightarrow A}$$

$$\longrightarrow \neg\neg((\neg a \Rightarrow b) \Rightarrow (\neg b \Rightarrow a))$$

$$\mid (\rightarrow \neg)$$

$$\neg((\neg a \Rightarrow b) \Rightarrow (\neg b \Rightarrow a)) \longrightarrow$$

$$\mid (\neg \rightarrow)$$

$$\longrightarrow ((\neg a \Rightarrow b) \Rightarrow (\neg b \Rightarrow a))$$

$$\mid (\rightarrow\Rightarrow)$$

$$(\neg a \Rightarrow b) \longrightarrow (\neg b \Rightarrow a)$$

$$\mid (\rightarrow\Rightarrow)$$

$$(\neg a \Rightarrow b), \neg b \longrightarrow a$$

$$\mid (\neg \rightarrow)$$

$$(\neg a \Rightarrow b) \longrightarrow b, a$$

$$\bigwedge (\Rightarrow\rightarrow)$$

$$\longrightarrow \neg a, b, a \qquad\qquad b \longrightarrow b, a$$

$$\mid (\rightarrow \neg) \qquad\qquad \textit{axiom}$$

$$a \longrightarrow b, a$$

$$\textit{axiom}$$

Solution
The tree above is not a proof in **GL** because a decomposition rule used in the decomposition step below **does not exists** in **GL**

$$(\neg a \Rightarrow b), \neg b \longrightarrow a$$

$$\mid (\neg \rightarrow)$$

$$(\neg a \Rightarrow b) \longrightarrow b, a.$$

It is a proof is some system **GL1** that has all the rules of **GL** except its rule $(\neg \rightarrow)$

$$(\neg \rightarrow) \quad \frac{\Gamma', \Gamma \ \longrightarrow\ \Delta,\ A,\ \Delta'}{\Gamma',\ \neg A,\ \Gamma\ \longrightarrow\ \Delta, \Delta'}$$

This rule has to be replaced in by the rule:

$$(\neg \rightarrow)_1 \ \frac{\Gamma, \Gamma' \ \longrightarrow\ \Delta,\ A,\ \Delta'}{\Gamma,\ \neg A,\ \Gamma'\ \longrightarrow\ \Delta, \Delta'}$$

6.6 GL Soundness and Completeness

The system **GL** admits a constructive proof of the Completeness Theorem, similar to completeness proofs for **RS** type proof systems (Theorems 9.7, 6.6, 6.7). It also relays on strong soundness property of its inference rules. We are going to prove that the following holds.

Theorem 6.8 (GL Strong Soundness)

The proof system **GL** *is strongly sound.*

Proof We have already proved (Fact 6.4) that logical axioms of **GL** are tautologies, so we have to prove now that its rules of inference are strongly sound (definition 6.4). Proofs of strong soundness of rules of inference of **GL** are more involved then the proofs for the **RS** type rules. We prove as an example the strong soundness of four of inference rules. Proofs for all other rules follows the same patterns and is left as an exercise.

By definition 6.4 of strong soundness we have to show the condition (8.77). Written formally it says that we have to show that that if P_1, P_2 are premisses of a given rule and C is its conclusion, then for all truth assignments $v : VAR \longrightarrow \{T, F\}$,

$$v^*(P_1) = v^*(C) \text{ in case of one premiss rule, and}$$

$$v^*(P_1) \cap v^*(P_2) = v^*(C), \text{in case of a two premisses rule.}$$

In order to prove it we need additional classical equivalencies listed below. You can fond a list of most basic classical equivalences in Chapter 3.

$$((A \Rightarrow B) \cap (A \Rightarrow C)) \equiv (A \Rightarrow (B \cap C))$$
$$((A \Rightarrow C) \cap (B \Rightarrow C)) \equiv ((A \cup B) \Rightarrow C)$$
$$((A \cap B) \Rightarrow C) \equiv (A \Rightarrow (\neg B \cup C))$$

$$(\cap \rightarrow)\ \frac{\Gamma^{'}, A, B, \Gamma \ \longrightarrow\ \Delta^{'}}{\Gamma', (A \cap B), \Gamma \ \longrightarrow\ \Delta'}$$

$v^*(\Gamma^{'}, A, B, \Gamma \ \longrightarrow\ \Delta^{'}) = (v^*(\Gamma^{'}) \cap v^*(A) \cap v^*(B) \cap v^*(\Gamma)) \Rightarrow v^*(\Delta^{'}) = (v^*(\Gamma^{'}) \cap v^*(A \cap B) \cap v^*(\Gamma)) \Rightarrow v^*(\Delta^{'}) = v^*(\Gamma^{'}, (A \cap B), \Gamma \ \longrightarrow\ \Delta^{'})$

$$(\rightarrow \cap)\ \frac{\Gamma \ \longrightarrow\ \Delta, A, \Delta^{'}\ ;\ \Gamma \ \longrightarrow\ \Delta, B, \Delta^{'}}{\Gamma \ \longrightarrow\ \Delta, (A \cap B), \Delta^{'}}$$

$v^*(\Gamma \ \longrightarrow\ \Delta, A, \Delta^{'}) \cap v^*(\Gamma \longrightarrow\ \Delta, B, \Delta^{'})$
$= (v^*(\Gamma) \Rightarrow v^*(\Delta) \cup v^*(A) \cup v^*(\Delta^{'})) \cap (v^*(\Gamma) \Rightarrow v^*(\Delta) \cup v^*(B) \cup v^*(\Delta^{'}))$

[we use : $((A \Rightarrow B) \cap (A \Rightarrow C)) \equiv (A \Rightarrow (B \cap C))$]
$= v^*(\Gamma) \Rightarrow ((v^*(\Delta) \cup v^*(A) \cup v^*(\Delta^{'})) \cap (v^*(\Delta) \cup v^*(B) \cup v^*(\Delta^{'})))$
[we use commutativity and distributivity]
$= v^*(\Gamma) \Rightarrow (v^*(\Delta) \cup (v^*(A \cap B)) \cup v^*(\Delta^{'}))$
$= v^*(\Gamma \ \longrightarrow \ \Delta, (A \cap B), \Delta^{'})$

$$(\cup \rightarrow) \ \frac{\Gamma^{'}, A, \Gamma \ \longrightarrow \ \Delta^{'} \ ; \ \Gamma^{'}, B, \Gamma \ \longrightarrow \ \Delta^{'}}{\Gamma^{'}, (A \cup B), \Gamma \ \longrightarrow \ \Delta^{'}}$$

$v^*(\Gamma^{'}, A, \Gamma \ \longrightarrow \ \Delta^{'}) \cap v^*(\Gamma^{'}, B, \Gamma \ \longrightarrow \Delta^{'})$
$= (v^*(\Gamma^{'}) \cap v^*(A) \cap v^*(\Gamma)) \Rightarrow v^*(\Delta^{'})) \cap (v^*(\Gamma^{'}) \cap v^*(B) \cap v^*(\Gamma)) \Rightarrow v^*(\Delta^{'}))$
[we use: $((A \Rightarrow C) \cap (B \Rightarrow C)) \equiv ((A \cup B) \Rightarrow C$])
$= (v^*(\Gamma^{'}) \cap v^*(A) \cap v^*(\Gamma)) \cup (v^*(\Gamma^{'}) \cap v^*(B) \cap v^*(\Gamma)) \Rightarrow v^*(\Delta^{'})$
$= ((v^*(\Gamma^{'}) \cap v^*(\Gamma)) \cap v^*(A)) \cup ((v^*(\Gamma^{'}) \cap v^*(\Gamma)) \cap v^*(B)) \Rightarrow v^*(\Delta^{'})$
[we use commutativity and distributivity]
$= ((v^*(\Gamma^{'}) \cap v^*(\Gamma)) \cap (v^*(A \cup B)) \Rightarrow v^*(\Delta^{'})$
$= v^*(\Gamma^{'}, (A \cup B), \Gamma \ \longrightarrow \ \Delta^{'})$

$$(\rightarrow \neg) \ \frac{\Gamma^{'}, A, \Gamma \ \longrightarrow \ \Delta, \Delta^{'}}{\Gamma^{'}, \Gamma \ \longrightarrow \ \Delta, \neg A, \Delta^{'}}$$

$v^*(\Gamma^{'}, A, \Gamma \ \longrightarrow \ \Delta, \Delta^{'}) = v^*(\Gamma^{'}) \cap v^*(A) \cap v^*(\Gamma) \Rightarrow v^*(\Delta) \cup v^*(\Delta^{'})$
$= (v^*(\Gamma^{'}) \cap v^*(\Gamma)) \cap v^*(A) \Rightarrow v^*(\Delta) \cup v^*(\Delta^{'})$
[we use: $((A \cap B) \Rightarrow C) \equiv (A \Rightarrow (\neg B \cup C))$]
$= (v^*(\Gamma^{'}) \cap v^*(\Gamma)) \Rightarrow \neg v^*(A) \cup v^*(\Delta) \cup v^*(\Delta^{'}) = (v^*(\Gamma^{'}) \cap v^*(\Gamma)) \Rightarrow v^*(\Delta) \cup$
$v^*(\neg A) \cup v^*(\Delta^{'})$
$= v^*(\Gamma^{'}, \Gamma \ \longrightarrow \ \Delta, \neg A, \Delta^{'})$

The above shows the premises and conclusions are logically equivalent, therefore the rules of inference are strongly sound. It **ends** the proof.

Observe that the strong soundness implies soundness (not only by name!), hence we have also proved the following

Theorem 6.9 (Soundness for GL)

For any sequent $\Gamma \longrightarrow \Delta \in SQ$,
if $\vdash_{\boldsymbol{GL}} \Gamma \longrightarrow \Delta$, *then* $\models \ \Gamma \longrightarrow \Delta$. *In particular, for any* $A \in \mathcal{F}$,
if $\vdash_{\boldsymbol{GL}} A$, *then* $\models \ A$.

We know by theorem 6.8 that all the rules of inference of **GL** of are strongly sound. The strong soundness of the rules means that if *at least* one of premisses of a rule is *false*, so is its conclusion. Hence given a sequent $\Gamma \longrightarrow \Delta \in SQ$, such that its decomposition tree $\mathbf{T}_{\Gamma\longrightarrow\Delta}$ has a branch ending with a non-axiom leaf. It means that any truth assignment v that make this non-axiom leaf false also falsifies all sequences on that branch, and hence falsifies the sequent $\Gamma \longrightarrow \Delta$. In particular, given a sequent $\longrightarrow A$ and its tree $\mathbf{T}_{\longrightarrow A}$, any v, such that falsifies its a non-axiom leaf is a counter-model for A. We have hence proved the following.

Theorem 6.10 (GL Counter Model)

Given a sequent $\Gamma \longrightarrow \Delta$, such that its decomposition tree $\mathbf{T}_{\Gamma\longrightarrow\Delta}$ contains a non- axiom leaf L_A. Any truth assignment v that falsifies the non-axiom leaf L_A is a counter model for $\Gamma \longrightarrow \Delta$. In particular, given a formula $A \in \mathcal{F}$, and its decomposition tree $\mathbf{T}_A$ with a non-axiom leaf, this leaf let us define a counter-model for A **determined** *by the decomposition tree $\mathbf{T}_A$.*

Here is a simple exercise explaining how the construction of a counter-model determined by the decomposition tree of a works.

Exercise 6.17

Prove, by constructing a counter-model determined by decomposition tree that

$$\not\models \ ((b \Rightarrow a) \Rightarrow (\neg b \Rightarrow a)).$$

Solution
We construct the decomposition tree for the formula $A : ((b \Rightarrow a) \Rightarrow (\neg b \Rightarrow a))$ as follows.

$$\mathbf{T}_{\rightarrow A}$$

$$\longrightarrow ((b \Rightarrow a) \Rightarrow (\neg b \Rightarrow a))$$

$$\mid (\rightarrow\Rightarrow)$$

$$(b \Rightarrow a) \ \longrightarrow \ (\neg b \Rightarrow a)$$

$$\mid (\rightarrow\Rightarrow)$$

$$\neg b, (b \Rightarrow a) \ \longrightarrow \ a$$

$$\mid (\neg \rightarrow)$$

$$(b \Rightarrow a) \ \longrightarrow \ b, a$$

$$\bigwedge(\Rightarrow\longrightarrow)$$

$$\longrightarrow \ b, b, a \qquad\qquad\qquad a \longrightarrow b, a$$

$$non - axiom \qquad\qquad\qquad axiom$$

The non-axiom leaf L_A we want to falsify is $\longrightarrow\ b, b, a$. Let $v : VAR \longrightarrow \{T, F\}$ be a truth assignment. By definition 6.7 of semantic for **GL** we have that $v^*(L_A) = v^*(\longrightarrow b, b, a) = (T \Rightarrow v(b) \cup v(b) \cup v(a))$. Hence $v^*(\longrightarrow b, b, a) = F$ if and only if $(T \Rightarrow v(b) \cup v(b) \cup v(a)) = F$ if and only if $v(b) = v(a) = F$. The Theorem 6.10, says that the logical value **F** determined by the evaluation a non-axiom leaf L_A "climbs" the decomposition tree. We picture it as follows.

$$\mathbf{T}_{\rightarrow A}$$

$$\longrightarrow ((b \Rightarrow a) \Rightarrow (\neg b \Rightarrow a)) \quad \mathbf{F}$$

$$\mid (\rightarrow\Rightarrow)$$

$$(b \Rightarrow a) \ \longrightarrow \ (\neg b \Rightarrow a) \ \ \mathbf{F}$$

$$\mid (\rightarrow\Rightarrow)$$

$$\neg b, (b \Rightarrow a) \ \longrightarrow \ a \ \ \mathbf{F}$$

$$\mid (\neg \rightarrow)$$

$$(b \Rightarrow a) \ \longrightarrow \ b, a \ \ \mathbf{F}$$

$$\bigwedge(\Rightarrow\longrightarrow)$$

$$\longrightarrow \ b, b, a \ \ \mathbf{F} \qquad\qquad\qquad a \longrightarrow b, a$$

$$non - axiom \qquad\qquad\qquad axiom$$

So, by theorem 6.10, any truth assignment $v : VAR \longrightarrow \{T, F\}$, such that $v(b) = v(a) = F$ falsifies the sequence $\longrightarrow A$, i.e. $v^*(\longrightarrow A) = T \Rightarrow v^*(A) = F$. This is possible only if $v^*(A) = F$. This proves that v is a counter model for A and we proved that $\not\models A$.

Our main goal is to prove the Completeness Theorem for **RS**. We prove first the Completeness Theorem for formulas $A \in \mathcal{F}$ and then we generalize it to any sequences $\Gamma \in \mathcal{F}^*$.

Theorem 6.11 (Completeness Theorem)

For any formula $A \in \mathcal{F}$,

$$\vdash_{\mathbf{GL}} A \quad \textit{if and only if} \quad \models A.$$

For any sequent $\Gamma \longrightarrow \Delta \in SQ$,

$$\vdash_{\mathbf{GL}} \Gamma \longrightarrow \Delta \quad \textit{if and only if} \quad \models \Gamma \longrightarrow \Delta.$$

Proof
We have already proved the Soundness Theorem 6.9, so we need to prove only the completeness part of it, namely to prove the implication:

$$\text{if} \quad \models A, \quad \text{then} \ \vdash_{\mathbf{GL}} A. \tag{6.25}$$

We prove instead of the logically equivalent opposite implication:

$$\text{if} \quad \not\vdash_{\mathbf{GL}} A \ \text{ then } \ \not\models A. \tag{6.26}$$

Assume $\not\vdash_{\mathbf{GL}} A$. By (6.24) it means that $\not\vdash_{\mathbf{GL}} \longrightarrow A$. Let $\mathcal{T}_A$ be a set of all decomposition trees of $\longrightarrow A$. As $\not\vdash_{\mathbf{GL}} \longrightarrow A$, each tree $\mathbf{T}_{\rightarrow A}$ in the set $\mathcal{T}_A$ has a non-axiom leaf. We choose an arbitrary $\mathbf{T}_{\rightarrow A}$ from $\mathcal{T}_A$. Let $L_A = \Gamma^{'} \longrightarrow \Delta^{'}$, be a non-axiom leaf of $\mathbf{T}_{\rightarrow A}$. We define a truth assignment $v : VAR \longrightarrow \{T, F\}$ which falsifies $\Gamma^{'} \longrightarrow \Delta^{'}$ as follows.

$$v(a) = \begin{cases} T & \text{if a appears in } \Gamma' \\ F & \text{if a appears in } \Delta' \\ \text{any value} & \text{if a does not appear in } \Gamma' \rightarrow \Delta' \end{cases}$$

By the strong soundness of the rules of inference of **GL** and Theorem 6.10 it proves that $v^*(\longrightarrow A) = F$, i.e. that $\not\models \longrightarrow A$ and hence $\not\models A$.

Assume that $\Gamma \longrightarrow \Delta$ is any sequence such that $\not\vdash_{\mathbf{GL}} \Gamma \longrightarrow \Delta$. But $\vdash_{\mathbf{GL}} \Gamma \longrightarrow \Delta$ if and only if $\vdash_{\mathbf{GL}} (\sigma_\Gamma \Rightarrow \delta_\Delta)$. So $\not\vdash_{\mathbf{GL}} \Gamma \longrightarrow \Delta$ if and only if $\not\vdash_{\mathbf{GL}} \sigma_\Gamma \Rightarrow \delta_\Delta)$. By already proven Case 1, $\not\models \sigma_\Gamma \Rightarrow \delta_\Delta)$, what is obviously equivalent to $\not\models \Gamma \longrightarrow \Delta$. This ends the proof of Case 2 and Completeness Theorem.

Gentzen Sequent Proof System G (6.27)

The proof system **G** is in its structure the most similar to the proof system **RS2** defined by (6.19).

It is obtained from in the same way is a proof system obtained from **GL** by changing the indecomposable sequences $\Gamma^{'}, \ \Delta^{'}$ into any sequences $\Sigma, \ \Lambda \in \mathcal{F}^*$ in **all of the rules** of inference of **GL**.

The **logical axioms** LA remain the same; i.e. the components of **G** are as follows.

Axioms of G

As the axioms of **GL** we adopt any indecomposable sequent which contains a positive literal a (variable) that appears on both sides of the sequent arrow $\longrightarrow$, i.e any sequent of the form

$$\Gamma'_1, \ a, \ \Gamma'_2 \ \longrightarrow \ \Delta'_1, \ a, \ \Delta'_2, \tag{6.28}$$

for any $a \in VAR$ and any sequences $\Gamma'_1, \Gamma'_2, \Delta'_1, \Delta'_2 \in VAR^*$.

Inference Rules of G (6.29)

We adopt the following rules of inference.

Conjunction rules

$$(\cap \rightarrow)\ \frac{\Sigma,\ A, B,\ \Gamma \ \longrightarrow\ \Lambda}{\Sigma, (A \cap B), \Gamma \ \longrightarrow\ \Lambda},$$

$$(\rightarrow \cap)\ \frac{\Gamma \ \longrightarrow\ \Delta, A, \Lambda\ ;\ \Gamma \ \longrightarrow\ \Delta, B, \Lambda}{\Gamma \ \longrightarrow\ \Delta, (A \cap B), \Lambda},$$

Disjunction rules

$$(\rightarrow \cup)\ \frac{\Gamma \ \longrightarrow\ \Delta, A, B, \Lambda}{\Gamma \ \longrightarrow\ \Delta, (A \cup B), \Lambda},$$

$$(\cup \rightarrow)\ \frac{\Sigma, A, \Gamma \ \longrightarrow\ \Lambda\ ;\ \Sigma, B, \Gamma \ \longrightarrow\ \Lambda}{\Sigma, (A \cup B), \Gamma \ \longrightarrow\ \Lambda},$$

Implication rules

$$(\rightarrow\Rightarrow)\ \frac{\Sigma, A, \Gamma \ \longrightarrow\ \Delta, B, \Lambda}{\Sigma, \Gamma \ \longrightarrow\ \Delta, (A \Rightarrow B), \Lambda},$$

$$(\Rightarrow\rightarrow)\ \frac{\Sigma, \Gamma \ \longrightarrow\ \Delta, A, \Lambda\ ;\ \Sigma, B, \Gamma \ \longrightarrow\ \Delta, \Lambda}{\Sigma, (A \Rightarrow B), \Gamma \ \longrightarrow\ \Delta, \Lambda},$$

Negation rules

$$(\neg \rightarrow)\ \frac{\Sigma, \Gamma \ \longrightarrow\ \Delta, A, \Lambda}{\Sigma, \neg A, \Gamma \ \longrightarrow\ \Delta, \Lambda}, \qquad (\rightarrow \neg)\ \frac{\Sigma, A, \Gamma \ \longrightarrow\ \Delta, \Lambda}{\Sigma, \Gamma \ \longrightarrow\ \Delta, \neg A, \Lambda},$$

where $\Gamma, \Delta,\ \Sigma.\ \Lambda \in \mathcal{F}^*$.

Exercise 6.18 *Follow the example of the* **GL** *system and adopt all needed definitions and proofs to prove the completeness of the system* **G**.

Solution
We leave it to the reader to fill in details .In particular, one has to accomplish the steps below.

1. Explain why the system **G** is strongly sound. You can use the strong soundness of the system **GL** .

2. Prove, as an example, a strong soundness of 4 rules of **G**.

3. Prove the the following Strong Soundness Theorem for **G**.

Theorem 6.12

The proof system **G** *is* **strongly sound**.

4. Define shortly, in your own words, for any formula $A \in \mathcal{F}$, its decomposition tree $\mathbf{T}_{\rightarrow A}$ in **G**.

5. Extend your definition to a decomposition tree $\mathbf{T}_{\Gamma \rightarrow \Delta}$.

6. Prove that for any $\Gamma \rightarrow \Delta \in SQ$, the decomposition tree $\mathbf{T}_{\Gamma \rightarrow \Delta}$ are finite.

7. Give an example of formulas $A, B \in \mathcal{F}$ such that that $\mathbf{T}_{\rightarrow A}$ is unigue and $\mathbf{T}_{\rightarrow B}$ is not.

8. Prove the following Counter Model Theorem for **G**.

Theorem 6.13

Given a sequent $\Gamma \longrightarrow \Delta$*, such that its decomposition tree* $\mathbf{T}_{\Gamma \longrightarrow \Delta}$ *contains a non- axiom leaf* L_A*. Any truth assignment* v *that falsifies the non-axiom leaf* L_A *is a counter model for* $\Gamma \longrightarrow \Delta$*.*

10. Prove the following Completeness Theorem for **G**.

Theorem 6.14 *For any formula* $A \in \mathcal{F}$,
1. $\vdash_{\mathbf{G}} A$ *if and only if* $\models A$,
and for any sequent $\Gamma \longrightarrow \Delta \in SQ$,
2. $\vdash_{\mathbf{G}} \Gamma \longrightarrow \Delta$ *if and only if* $\models \Gamma \longrightarrow \Delta$.

6.7 Original Gentzen Systems LK, LI Completeness and Hauptzatz Theorems

The original systems **LK** and **LI** were created by Gentzen in 1935 for classical and intuitionistic predicate logics, respectively. The proof system **LI** for intuitionistic propositional logic is a particular case of the proof system **LK**.

Both systems **LK** and **LI** have two groups of inference rules and a special rule called a *cut rule.* One group consists of a set of rules similar to the rules of systems **GL** and **G**. We call them Logical Rules. The other group contains a new type of rules, called Structural Rules. The *cut* rule in Gentzen sequent systems corresponds to the Modus Ponens rule in Hilbert proof systems as Modus Ponens is a particular case of the cut rule. The cut rule is needed to carry the original Gentzen proof of the completeness of the system **LK** and proving the adequacy of **LI** system for intituitionistic logic. Gentzen proof of completeness of **LK** was not direct. He used the completeness of already known Hilbert proof systems H and proved that any formula provable in the systems H is also provable in **LK**, respectively. Hence the need of the cut rule.

For the system **LI** he proved only the adequacy of **LI** system for intituitionistic logic since the semantics for the intuitionistic logic didn't yet exist. He used the acceptance of the Heying intuitionistic axiom system as the definition of the intuitionistic logic and proved that any formula provable in the Heyting system is also provable in **LI**.

Observe that by presence of the cut rule, Gentzen **LK, LI** systems are also a Hilbert system. What distinguishes it from all other known Hilbert proof systems is the fact that the cut rule could be eliminated from it.

This is Gentzen famous *Hauptzatz Theorem*, also called *Cut Elimination Theorem.* The elimination of the cut rule and the structure of other rules makes it possible to define effective automatic procedures for proof search, what is impossible in a case of the Hilbert style systems.

Gentzen, in his proof of Hauptzatz Theorem, developed a powerful technique adaptable to other logics. We present it here in classical propositional case and show how to adapt it to the intuitionistic case.

Gentzen proof is purely syntactical. It defines a constructive method of transformation of any formal proof (derivation) of a sequent $\Gamma \longrightarrow \Delta$ that uses a cut rule (and other rules) into its proof without use of the cut rule. Hence the English name *Cut Elimination Theorem.*

The completeness (with respect to algebraic semantics defined in chapter 7) of the cut free system **LI** follows directly from LI Hauptzatz Theorem 6.22 and the intuitionistic completeness theorem (chapter 7). The proof is a straightforward adaptation of the proof of cut free **LK** Completeness Theorem 6.23 and is left as a homework exercise in chapter 7.

Rasiowa and Sikorski method of proving completeness theorem by constructing counter-models on the decomposition trees is a semantical equivalence to purely syntactical Gentzen proof of cut elimination. It is relatively simple, powerful and easy to understand. It was the reason it was first to be presented here. But it is more difficult and sometimes impossible to apply (generalize) to many non-classical logics then Gentzen cut elimination method. Moreover the Gentzen method is more flexible and in this sense more general and powerful. This is why we preset it here.

Components of **LK, LI** (6.30)

Language $\mathcal{L}$
The language is the same as the in case of **GL**, namely

$$\mathcal{L} = \mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow\}}.$$

Expressions

The set of all expressions $\mathcal{E}$ is, as before, the set

$$SQ = \{\Gamma \longrightarrow \Delta : \quad \Gamma, \Delta \in \mathcal{F}^*\} \tag{6.31}$$

of all sequents.

Logical Axioms
There is only one logical axiom, namely

$$A \quad \longrightarrow \quad A,$$

where A is any formula of $\mathcal{L}$.

Rules of Inference
There are two groups of rules of inference and they are defined are as follows.

GROUP ONE: STRUCTURAL RULES.

Weakening in the antecedent

$$(weak \rightarrow) \quad \frac{\Gamma \;\longrightarrow\; \Delta}{A,\; \Gamma \;\longrightarrow\; \Delta},$$

Weakening in the succedent

$$(\rightarrow weak) \quad \frac{\Gamma \;\longrightarrow\; \Delta}{\Gamma \;\longrightarrow\; \Delta,\; A},$$

Contraction in the antecedent

$$(contr \rightarrow) \quad \frac{A,\; A,\; \Gamma \;\longrightarrow\; \Delta}{A,\; \Gamma \;\longrightarrow\; \Delta},$$

Contraction in the succedent

$$(\rightarrow contr) \quad \frac{\Gamma \;\longrightarrow\; \Delta,\; A,\; A}{\Gamma \;\longrightarrow\; \Delta,\; A},$$

Exchange in the antecedent

$$(exch \rightarrow) \quad \frac{\Gamma_1,\; A,\; B,\; \Gamma_2 \;\longrightarrow\; \Delta}{\Gamma_1,\; B,\; A,\; \Gamma_2 \;\longrightarrow\; \Delta},$$

Exchange in the succedent

$$(\rightarrow exch) \quad \frac{\Delta \longrightarrow \Gamma_1,\; A,\; B,\; \Gamma_2}{\Delta \longrightarrow \Gamma_1,\; B,\; A,\; \Gamma_2},$$

Cut Rule

$$(cut) \quad \frac{\Gamma \longrightarrow \Delta,\ A \quad ; \quad A,\ \Sigma \longrightarrow \Theta}{\Gamma, \Sigma \longrightarrow \Delta, \Theta}.$$

The formula A is called a *cut formula.*

GROUP TWO: LOGICAL RULES

Conjunction

$$(\cap \rightarrow)_1 \quad \frac{A,\ \Gamma \longrightarrow \Delta}{(A \cap B),\ \Gamma \longrightarrow \Delta},$$

$$(\cap \rightarrow)_2 \quad \frac{B,\ \Gamma \longrightarrow \Delta}{(A \cap B),\ \Gamma \longrightarrow \Delta},$$

$$(\rightarrow \cap) \quad \frac{\Gamma \longrightarrow \Delta,\ A \quad ; \quad \Gamma \longrightarrow \Delta,\ B}{\Gamma \longrightarrow \Delta,\ (A \cap B)}.$$

Disjunction

$$(\rightarrow \cup)_1 \quad \frac{\Gamma \longrightarrow \Delta,\ A}{\Gamma \longrightarrow \Delta,\ (A \cup B)},$$

$$(\rightarrow \cup)_2 \quad \frac{\Gamma \longrightarrow \Delta,\ B}{\Gamma \longrightarrow \Delta,\ (A \cup B)},$$

$$(\cup \rightarrow) \quad \frac{A,\ \Gamma \longrightarrow \Delta \quad ; \quad B,\ \Gamma \longrightarrow \Delta}{(A \cup B),\ \Gamma \longrightarrow \Delta}.$$

Implication

$$(\rightarrow\Rightarrow) \quad \frac{A,\ \Gamma \longrightarrow \Delta,\ B}{\Gamma \longrightarrow \Delta,\ (A \Rightarrow B)},$$

$$(\Rightarrow\rightarrow) \quad \frac{\Gamma \longrightarrow \Delta,\ A \quad ; \quad B,\ \Gamma \longrightarrow \Delta}{(A \Rightarrow B),\ \Gamma \longrightarrow \Delta}.$$

Negation

$$(\neg \rightarrow) \quad \frac{\Gamma \longrightarrow \Delta,\ A}{\neg A,\ \Gamma \longrightarrow \Delta},$$

$$(\rightarrow \neg) \quad \frac{A,\ \Gamma \longrightarrow \Delta}{\Gamma \longrightarrow \Delta,\ \neg A}.$$

Definition 6.9 (Classical System LK)

We define the classical Gentzen system **LK** *as*

$$\boldsymbol{LK} = (\mathcal{L},\ SQ,\ AL,\ \textit{Structural Rules},\ \ \textit{Cut Rule},\ \ \textit{Logical Rules}),$$

where all the components are defined by (9.24) above.

Definition 6.10 (Intuitionistic System LI)

We define the intuitionistic Gentzen system **LI** *as*

$$\boldsymbol{LK} = (\mathcal{L},\ ISQ,\ AL,\ \textit{I-Structural Rules},\ \ \textit{I- Cut Rule},\ \ \textit{I- Logical Rules}),$$

where ISQ is the following subset of the set SQ of all sequents (6.31)

$$ISQ = \{\Gamma \ \longrightarrow\ \Delta : \quad \Delta \ \textit{consists of at most one formula}\ \}. \tag{6.32}$$

The set ISQ is called the set of all **intuitionistic sequents***.*

The I-Structural Rules, I- Cut Rule, I- Logical Rules are the **LK** *rules restricted to the set ISQ (6.32) of the intuitionistic sequents.*

We will study the intuitionistic system **LI** in chapter 7. We concentrate now on then classical **LK**.

Classical System LK

We say that a formula $A \in \mathcal{F}$, has a proof in **LK** and denote it by $\vdash_{\mathbf{LK}} A$ if the sequent $\longrightarrow A$ has a proof in **GL**, i.e. we define:

$$\vdash_{\mathbf{LK}} A \ \text{ if and only if } \ \vdash_{\mathbf{LK}}\ \longrightarrow\ A. \tag{6.33}$$

Proof Trees
We write formal proofs in **LK**, as we did for other Gentzen style proof systems in a form of **trees** in an "upside -down" form.

By a **proof tree** of a sequent $\Gamma \longrightarrow \Delta$ in **LK** we understand a tree

$$\mathbf{D}_{\Gamma \longrightarrow \Delta}$$

satisfying the following conditions:

1. The topmost sequent, i.e the **root** of $\mathbf{D}_{\Gamma \longrightarrow \Delta}$ is $\Gamma \longrightarrow \Delta$.

2. All **leaves** are axioms.

3. The **nodes** are sequents such that each sequent on the tree follows from the ones immediately preceding it by one of the rules.

The proofs are often called **derivations**. In particular, Gentzen, in his work used the term derivation we will use this notion as well. This is why we denote the proof trees by **D** (for derivation).

Finding derivations **D** in **LK** are is a more complex process, as the logical rules are different, then in **GL** and **G**. Proofs rely strongly on use of the Structural Rules. Even if we find a derivation that does not involve the Cut rule, the Structural rules are usually present. For example, a **derivation** of Excluded Middle $(A \cup \neg A)$ formula B in **LK** is as follows.

D

$$\longrightarrow (A \cup \neg A)$$

$$\mid (\to contr)$$

$$\longrightarrow (A \cup \neg A),\ (A \cup \neg A)$$

$$\mid (\to \cup)_1$$

$$\longrightarrow (A \cup \neg A),\ A$$

$$\mid (\to exch)$$

$$\longrightarrow A,\ (A \cup \neg A)$$

$$\mid (\to \cup)_1$$

$$\longrightarrow A,\ \neg A$$

$$\mid (\to \neg)$$

$$A \longrightarrow A$$

$$axiom$$

Here is as yet another example a proof **P** (also cut free) of the de Morgan Law $(\neg(A \cap B) \Rightarrow (\neg A \cup \neg B))$.

P

$$\longrightarrow (\neg(A \cap B) \Rightarrow (\neg A \cup \neg B))$$

$$\mid (\to\Rightarrow)$$

$$(\neg(A \cap B) \longrightarrow (\neg A \cup \neg B))$$

$$\mid (\to \neg)$$

$$\longrightarrow (\neg A \cup \neg B),\ (A \cap B)$$

$$\bigwedge(\Rightarrow\longrightarrow)$$

$$\begin{array}{cc}
\longrightarrow (\neg A \cup \neg B),\ A & \longrightarrow (\neg A \cup \neg B),\ B \\
\mid (\to exch) & \mid (\to exch) \\
\longrightarrow A, (\neg A \cup \neg B) & \longrightarrow B, (\neg A \cup \neg B) \\
\mid (\to \cup)_1 & \mid (\to \cup)_1 \\
\longrightarrow A, \neg A & \longrightarrow B, \neg B \\
\mid (\to \neg) & B \longrightarrow B \\
A \longrightarrow A & axiom \\
axiom &
\end{array}$$

Observe that the Logical Rules are similar in their structure to the rules of the system **G** and hence admit the same proof of their soundness.

The rules $(\rightarrow \cup)_1,\ (\rightarrow \cup)_2$ and $(\rightarrow \cup)_1,\ (\rightarrow \cup)_2$ are **not strongly sound** as $A \not\equiv (A \cap B), B \not\equiv (A \cap B)$ and $A \not\equiv (A \cap B), B \not\equiv (A \cap B)$.

All other Logical Rules are **strongly sound**.

The Contraction and Exchange structural are also strongly sound as for any formulas $A, B \in \mathcal{F},\ A \equiv (A \cap A),\ A \equiv (A \cup A)$ and $(A \cap B) \equiv (B \cap A),\ (A \cap B) \equiv (B \cap A)$. The Weakening rule is sound because (we use shorthand notation) if a formula $(\Gamma \Rightarrow \Delta) = T$ then also $((A \cap \Gamma) \Rightarrow \Delta)) = T$ for any logical value of the formula A. But obviously $(\Gamma \Rightarrow \Delta) \not\equiv ((A \cap \Gamma) \Rightarrow \Delta))$, i.e. the Weakening rule is not strongly sound.

The Cut rule is sound as the fact $(\Gamma \Rightarrow (\Delta \cup A)) = T$ and $((A \cap \Sigma) \Rightarrow \Lambda) = T$ implies that $\Gamma, \Sigma \ \longrightarrow\ \Delta, \Lambda$. It is not strongly sound. Any truth assignment such that $\Gamma = T,\ \Delta = \Sigma = \Lambda = A = F$ proves that $(\Gamma \ \longrightarrow\ \Delta,\ A) \cap (A,\ \Sigma \ \longrightarrow\ \Lambda) \not\equiv (\Gamma, \Sigma \ \longrightarrow\ \Delta, \Lambda)$. Obviously, $\models A \ \longrightarrow\ A$.
We have proved that **LK** is sound and hence the following theorem holds.

Theorem 6.15 (Soundness for LK)

For any sequent $\Gamma \longrightarrow \Delta$,

$$\textit{if } \vdash_{\boldsymbol{LK}} \Gamma \longrightarrow \Delta, \quad \textit{then} \quad \models\ \Gamma \longrightarrow \Delta.$$

In particular, for any $A \in \mathcal{F}$,

$$\textit{if } \vdash_{\boldsymbol{LK}} A, \quad \textit{then} \quad \models\ A.$$

We follow now Gentzen way of proving completeness of **LK**. We choose any complete Hilbert proof system for the **LK** language $\mathcal{L} = \mathcal{L}_{\{\neg, \cap, \cup, \Rightarrow\}}$ and prove its equivalency with **LK**.

Gentzen referred to the Hilbert-Ackerman (1920) system (axiomatization) included in chapter 5. We choose here the Rasiowa-Sikorski (1952) formalization R also included in chapter 5.

We do it for two reasons. First, it reflexes a connection between classical and intuitionistic logics very much in a spirit Gentzen relationship between **LK** and **LI**.

We obtain a complete proof system I (chapter 7) from R by just removing the last axiom A10. Second, both sets of axioms reflect the best what set of provable formulas is needed to conduct algebraic proofs of completeness of R and I, respectively.

Axioms of R (6.34)

The set of logical axioms of the Hilbert style proof system RS for classical propositional logic all formulas of the forms

A1 $((A \Rightarrow B) \Rightarrow ((B \Rightarrow C) \Rightarrow (A \Rightarrow C)))$,

A2 $(A \Rightarrow (A \cup B))$,

A3 $(B \Rightarrow (A \cup B))$,

A4 $((A \Rightarrow C) \Rightarrow ((B \Rightarrow C) \Rightarrow ((A \cup B) \Rightarrow C)))$,

A5 $((A \cap B) \Rightarrow A)$,

A6 $((A \cap B) \Rightarrow B)$,

A7 $((C \Rightarrow A) \Rightarrow ((C \Rightarrow B) \Rightarrow (C \Rightarrow (A \cap B)))$,

A8 $((A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \cap B) \Rightarrow C))$,

A9 $(((A \cap B) \Rightarrow C) \Rightarrow (A \Rightarrow (B \Rightarrow C))$,

A10 $(A \cap \neg A) \Rightarrow B)$,

A11 $((A \Rightarrow (A \cap \neg A)) \Rightarrow \neg A)$,

A12 $(A \cup \neg A)$,

where $A, B, C \in \mathcal{F}$ are any formulas in $\mathcal{L} = \mathcal{L}_{\{\neg, \cap, \cup, \Rightarrow\}}$.
We adopt a Modus Ponens

$$(MP)\ \frac{A\ ;\ (A \Rightarrow B)}{B}$$

as the only inference rule.

We define **Hilbert System** R as

$$R = (\ \mathcal{L}_{\{\neg, \cap, \cup, \Rightarrow\}},\ \mathcal{F},\ A1 - A12,\ (MP)\ ), \tag{6.35}$$

where A1 - A12 are defined by (6.34).

The system R is complete, i.e. we have the following.

Theorem 6.16

For any formula $A \in \mathcal{F}$,

$$\vdash_R A \ \text{ if and only if } \ \models A.$$

We leave it as an exercise for the reader to show that all axioms A1 - A12 of the system R are provable in **LK**. Moreover, the Modus Ponens is a particular case of the cut rule, for Γ, Δ, Σ empty sequences and Θ containing only one element, a formula B. We call it also MP rule.

$$(MP) \quad \frac{\longrightarrow A \ ; \ A \longrightarrow B}{\longrightarrow B}.$$

This proves the following.

Theorem 6.17

For any formula $A \in \mathcal{F}$,

$$\textit{if } \vdash_R A, \textit{ then} \vdash_{\mathbf{LK}} A.$$

Directly from the above theorem 6.17, soundness of **LK** (theorem 6.15) and completeness of R (theorem 6.16) we get the completeness of **LK**.

Theorem 6.18 (LK Completeness)

For any formula $A \in \mathcal{F}$,

$$\vdash_{\mathbf{LK}} A \ \textit{ if and only if } \ \models A.$$

Here is Gentzen original formulation of the Hauptzatz Theorems, which we call also the Cut Elimination Theorem.

Theorem 6.19 (Hauptzatz) *(Classical* **LK***)*

Every derivation in **LK** *can be transformed into another* **LK** *derivation of the same sequent, in which no cuts occur.*

Theorem 6.20 (Hauptzatz) *(Intuitionistic* **LI***)*

Every derivation in **LI** *can be transformed into another* **LI** *derivation of the same sequent, in which no cuts occur.*

The proof is quite long and involved. We present here its main and most important steps. To facilitate the proof we introduce a more general form of the cut rule, called a mix rule defined as follows.

$$(mix) \quad \frac{\Gamma \longrightarrow \Delta \ ; \ \Sigma \longrightarrow \Theta}{\Gamma, \Sigma^* \longrightarrow \Delta^*, \Theta}, \tag{6.36}$$

where Σ^*, Δ^* are obtained from Σ, Δ by removing all occurrences of a common formula A. The formula A is now called a *mix formula.*

Example 6.3

Here are some examples of an applications of the mix rule. Observe that the mix rule applies, as the cut does, to only one mix formula at the time.

$$(mix)\quad \frac{a \longrightarrow b,\ \neg a \quad ; \quad (b \cup c),\ b,\ b, D, b \longrightarrow}{a,\ (b \cup c),\ D \longrightarrow \neg a}$$

b *is the mix formula.*

$$(mix)\quad \frac{A \longrightarrow B,\ B,\ \neg A \quad ; \quad (b \cup c),\ B,\ B, D, B \longrightarrow \neg B}{A,\ (b \cup c),\ D \longrightarrow \neg A, \neg B}$$

B *is the mix formula.*

$$(mix)\quad \frac{A \longrightarrow B, \neg A,\ \neg A \quad ; \quad \neg A,\ B,\ B, \neg A, B \longrightarrow \neg B}{A,\ B,\ B \longrightarrow B, \neg B}$$

$\neg A$ *is the mix formula.*

Notice, that every derivation with cut may be transformed into a derivation with mix by means of a number of weakenings (multiple application of the weakening rules) and interchanges (multiple application of the exchange rules). Conversely, every mix may be transformed into a cut derivation by means of a certain number of preceding exchanges and contractions, though we do not use this fact in the proof. Observe that cut is a particular case of mix.

Proof of **Hauptzatz Theorems**
The proof for **LI** is the same as for **LK**. We must just be careful to add, at each step, the restriction to the ISQ sequences and the form of the rules. These restrictions do not alter the flow and validity of the **LK**proof. We leave it as homework exercise to the reader to re-write the proof below step by step for **LI**.

We conduct the proof in three main steps.
Step 1: we consider only derivations in which only mix rule is used.
Step 2: we consider first derivation with a certain Property H (definition 6.11) and prove lemma 6.2 for them. This lemma is the most crucial for the proof of the Hauptzatz.

Definition 6.11

We say that a derivation $\mathbf{D}_{\Gamma \longrightarrow \Delta}$ *of a sequent* $\Gamma \longrightarrow \Delta$ *has a* **Property H** *if it satisfies the the following conditions.*

1. The root $\Gamma \longrightarrow \Delta$ *of the derivation* $\mathbf{D}_{\Gamma \longrightarrow \Delta}$ *is obtained by direct use of the mix rule, i.e. the mix rule is the last rule of inference used in the proof (derivation) of* $\Gamma \longrightarrow \Delta$*.*

2. The derivation $\mathbf{D}_{\Gamma \longrightarrow \Delta}$ *does not contain any other application of the mix rule, i.e. the proof (derivation) of* $\Gamma \longrightarrow \Delta$ *does not contain any other application of the mix rule.*

Lemma 6.2 (H lemma)

Any derivation that fulfills the **Property H** *(definition 6.11) may be transformed into a derivation of the same sequent) in which no mix occurs.*

Step 3: we use the H lemma 6.2 and to prove the the Hauptzatz as follows.

Hauptzatz proof from H lemma

Let **D** be any derivation (tree proof). Let $\Gamma \longrightarrow \Delta$ be any node on **D** such that its sub-tree $\mathbf{D}_{\Gamma\longrightarrow\Delta}$ has the PropertyH (definition 6.11). By H lemma 6.2 the sub-tree $\mathbf{D}_{\Gamma\longrightarrow\Delta}$ can be replaced by a tree $\mathbf{D^*}_{\Gamma\longrightarrow\Delta}$ in which no mix occurs. The rest of **D** remains unchanged. We repeat this procedure for each node N, such that the sub-tree $\mathbf{D}_N$ has the Property H until every application of mix rule has systematically been eliminated. This **ends** the proof of Hauptzatz provided the H lemma 6.2 has already been proved.

Step 2: **proof of H lemma**.
We now consider derivation tree **D** with the Property H, i.e. such that the mix rule is the last rule of inference used, and **D** does not contain any other application of the mix rule.

We define now two important notions: *degree n* and *rank r* of the derivation **D**. Observe that **D** contains only one application of mix rule, and the mix rule, contains only one *mix formula* A. Mix rule used may contain many copies of A, but there always is only one mix formula. We call is a *mix formula* of **D**.

Definition 6.12

Given a derivation tree **D** *with the Property H.*
Let $A \in \mathcal{F}$ *be the mix formula of* **D**. *The degree* $n \geq 0$ *of* A *is called the* **degree** *of the derivation* **D**. *We write it as* $deg\mathbf{D} = degA = n$.

Definition 6.13

Given a derivation tree **D** *with the Property H. We define the rank r of* **D** *as a sum of its left rank Lr and right rank Rr of* **D**, *i.e.*

$$r = Lr \ + \ Rr,$$

where:

1. the left rank Lr of **D** *in the largest number of consecutive nodes on the branch of* **D** *staring with the node containing the left premiss of the mix rule, such that each sequent on these nodes contains the mix formula in the* **succedent***;*

2. the right rank Rr of **D** *in the largest number of consecutive nodes on the branch of* **D** *staring with the node containing the right premiss of the mix* *rule, such that each sequent on these nodes contains the mix formula in the* **antecedent**.

The lowest possible rank is evidently 2.
To prove the lemma we carry out *two complete inductions*, one on the *degree* n, the other on the *rank* r, of the derivation **D**.

It means we prove the lemma for a derivation of the degree n, assuming it to hold for derivations of a lower degree (in so far as there are such derivations, i.e., as long as $n \neq 0$), supposing, therefore, that derivations of lower degree can be already transformed into derivations without mix.

Furthermore, we shall begin by considering the case 1when the rank $r = 2$, and after that the case 2 when the rank $r > 2$, where we assume that the lemma already holds for derivations of the same degree, but a lower rank.

Case 1. Rank of r =2.

We present some cases and leave similar others to the reader as an exercise. Observe that first group contains cases that are especially simple in that they allow the mix to be immediately eliminated. The second group contains the most important cases since their consideration brings out the basic idea behind the whole proof, Here we use the *induction hypothesis* with respect do the *degree* of the derivation. We reduce each one of the cases to transformed derivations of a *lower degree.*

GROUP 1. Axioms and Structural Rules.

1. The left premiss of the mix rule is an axiom $A \longrightarrow A$.
Then the sub-tree of **D** containing mix is as follows.

$$A,\ \Sigma^* \ \longrightarrow\ \Delta$$

$$\bigwedge(mix)$$

$$A \longrightarrow A \qquad\qquad\qquad \Sigma \longrightarrow \Delta$$

We transform it, and replace it in **D** by

$$A,\ \Sigma^* \ \longrightarrow\ \Delta$$

possibly several exchanges and contractions

$$\Sigma \longrightarrow \Delta$$

Such obtained $\mathbf{D}^*$ proves the same sequent and contains no mix.

2 . The right premiss of the mix rule is an axiom $A \longrightarrow A$.
This is a case **dual** to 1. We show here the dial transformation, but will leave the dual cases to the reader in the future.

Then the sub-tree of $\mathbf{D}$ containing mix is as follows.

$$\Sigma \;\longrightarrow\; \Delta^*,\; A$$

$$\bigwedge(mix)$$

$$\Sigma \longrightarrow \Delta \qquad\qquad\qquad A \longrightarrow A$$

We transform it, and replace it in $\mathbf{D}$ by

$$\Sigma \;\longrightarrow\; \Delta^*,\; A$$

possibly several exchanges and contractions

$$\Sigma \longrightarrow \Delta$$

Such obtained $\mathbf{D}^*$ proves the same sequent and contains no mix.

Suppose that neither of premisses of mix is an axiom. As the rank r=2 , the right and left ranks are equal one. This means that in the sequents on the nodes directly below left premiss of the mix, the mix formula A does not occur in the *succedent*; in the sequents on the nodes directly below right premiss of the mix, the mix formula A does not occur in the *antecedent*.

In general, if a formula occurs in the antecedent (succedent) of a conclusion of a rule of inference, it is either obtained by a logical rule or by a contraction rule.

3. The left premiss of the mix rule is the conclusion of a contraction rule ($\rightarrow$ *contr*). The sub-tree of $\mathbf{D}$ containing mix is:

$$\Gamma,\; \Sigma^* \;\longrightarrow\; \Delta,\; \Theta$$

$$\bigwedge(mix)$$

$$\Gamma \longrightarrow \ \Delta,\ A \qquad\qquad\qquad \Sigma \longrightarrow \Theta$$

$$\mid (\rightarrow contr)$$

$$\Gamma \longrightarrow \ \Delta$$

We transform it, and replace it in **D** by

$$\Gamma,\ \Sigma^{*} \ \longrightarrow\ \Delta,\ \Theta$$

possibly several weakenings and exchanges

$$\Gamma \longrightarrow \Delta$$

Observe that the whole branch of **D** that starts with the node $\Sigma \longrightarrow \Theta$ disappears. Such obtained $\mathbf{D}^{*}$ proves the same sequent and contains no mix.

4. The right premiss of the mix rule is the conclusion of a contraction rule ($\rightarrow$ *contr*). It is a dual case to 3. and is left to the reader.

GROUP 2. Logical Rules.

1. The main connective of the mix formula is $\cap$, i.e. the mix formula is $(A \cap B)$. The left premiss of the mix rule is the conclusion of a rule $(\rightarrow \cap)$. The right premiss of the mix rule is the conclusion of a rule $(\cap \rightarrow)_1$.
The sub-tree **T** of **D** containing mix is:

$$\Gamma,\ \Sigma \ \longrightarrow\ \Delta,\ \Theta$$

$$\bigwedge(mix)$$

$$\Gamma \longrightarrow \ \Delta,\ (A \cap B) \qquad\qquad\qquad (A \cap B),\ \Sigma \longrightarrow \Theta$$

$$\bigwedge((\rightarrow \cap)) \qquad\qquad\qquad\qquad \mid (\cap \rightarrow)_1$$

$$A, \Sigma \longrightarrow \Theta$$

$$\Gamma \longrightarrow \ \Delta, A \qquad \Gamma \longrightarrow \ \Delta, B$$

We transform **T** into $\mathbf{T}^{*}$ as follows.

$$\Gamma,\ \Sigma\ \longrightarrow\ \Delta,\ \Theta$$

possibly several weakenings and exchanges

$$\Gamma,\ \Sigma^{*}\ \longrightarrow\ \Delta^{*},\ \Theta$$

$$\bigwedge(mix)$$

$$\Gamma \longrightarrow\ \Delta, A \qquad\qquad\qquad A, \Sigma \longrightarrow \Theta$$

We replace **T** by **T*** in **D** and obtain **D***. Now we can apply induction hypothesis with respect to the degree of the mix formula. The mix formula A in **D*** has a lower degree then the mix formula $(A \cap B)$ and by the inductive assumption the derivation **D***, and hence **D** may be transformed into one without mix.

2. The case when the left premiss of the mix rule is the conclusion of a rule $(\rightarrow \cap)$ and right premiss of the mix rule is the conclusion of a rule $(\cap \rightarrow)_2$

3. The main connective of the mix formula is $\cup$, i.e. the mix formula is $(A \cup B)$. The left premiss of the mix rule is the conclusion of a rule $(\rightarrow \cup)_1$ or $(\rightarrow \cup)_2$. The right premiss of the mix rule is the conclusion of a rule $(\cup \rightarrow)_1$. This is to be dealt with symmetrically to the $\cap$ cases.

4. The main connective of the mix formula is $\neg$, i.e. the mix formula is $\neg A$. The left premiss of the mix rule is the conclusion of a rule $(\rightarrow \neg)$. The right premiss of the mix rule is the conclusion of a rule $(\neg \rightarrow)$.

Here is the sub-tree **T** of **D** containing the application of the mix rule.

$$\Gamma,\ \Sigma\ \longrightarrow\ \Delta,\ \Theta$$

$$\bigwedge(mix)$$

$$\begin{array}{ccc} \Gamma \longrightarrow\ \Delta,\ \neg A & \qquad\qquad & \neg A, \Sigma \longrightarrow\ \Theta \\ \mid (\rightarrow \neg) & & \mid (\neg \rightarrow) \\ A, \Gamma \longrightarrow\ \Delta & & \Sigma \longrightarrow\ \Theta, A \end{array}$$

We transform **T** into **T*** as follows.

$$\Gamma,\ \Sigma\ \longrightarrow\ \Delta,\ \Theta$$

possibly several weakenings and exchanges

$$\Sigma,\ \Gamma^{*}\ \longrightarrow\ \Theta^{*},\ \Delta$$

$$\bigwedge(mix)$$

$$\Sigma \longrightarrow\ \Theta, A \qquad\qquad A, \Gamma \longrightarrow\ \Delta$$

We replace **T** by $\mathbf{T}^{*}$ in **D** and obtain $\mathbf{D}^{*}$. The new mix in $\mathbf{D}^{*}$ may be eliminated by virtue of inductive assumption, and so from the derivation **D**.

5. The main connective of the mix formula is $\Rightarrow$, i.e. the mix formula is $(A \Rightarrow B)$. The left premiss of the mix rule is the conclusion of a rule $((\rightarrow\Rightarrow)$. The right premiss of the mix rule is the conclusion of a rule $(\Rightarrow\rightarrow)$.

Here is the sub-tree **T** of **D** containing the application of the mix rule.

$$\Gamma,\ \Sigma\ \longrightarrow\ \Delta,\ \Theta$$

$$\bigwedge(mix)$$

$$\Gamma \longrightarrow \Delta,\ (A \Rightarrow B) \qquad\qquad (A \Rightarrow B),\ \Sigma \longrightarrow\ \Theta$$

$$\mid\ (\rightarrow\Rightarrow) \qquad\qquad \bigwedge((\rightarrow\cap))$$

$$A,\ \Gamma\ \longrightarrow \Delta,\ B$$

$$\Sigma \longrightarrow \Theta,\ A \qquad B,\ \Sigma \longrightarrow \Theta,$$

We transform **T**into $\mathbf{T}^{*}$ as follows.

$$\Gamma,\ \Sigma\ \longrightarrow\ \Delta,\ \Theta$$

possibly several weakenings and exchanges

$$\Sigma,\ \Gamma^{*}, \Sigma^{**} \longrightarrow\ \Theta^{*}, \Delta^{*}, \Theta$$

$$\bigwedge(mix)$$

$$\Sigma \longrightarrow \Theta,\ A \qquad\qquad A,\ \Gamma,\ \Sigma^{*}, \longrightarrow \Delta^{*},\ \Theta$$

$$\bigwedge(mix)$$

$$A,\ \Gamma\ \longrightarrow \Delta,\ B \quad B,\ \Sigma \longrightarrow \Theta,$$

The asteriks are, of course, intended as follows: Σ^*, Δ^* results from Σ, Δ by the omission of all formulas B; $\Gamma^*, \Sigma^{**}, \Theta^*$ results from Γ, Σ^*, Θ by the omission of all formulas A.

We replace **T** by **T*** in **D** and obtain **D***. Now we have two mixes, but both mix formulas are of a lower degree then n. We first apply the inductive to the assumption to the lower mix. Thus it can be eliminated. We can then also eliminate the upper mix. This **ends** the proof of the case of rank r=2.

Case $r > 2$.
In the case $r = 2$, we generally reduced the derivation to one of *lower degree*. Now we shall proceed to reduce the derivation to one of the *same* degree, but of a *lower rank*. This allows us to to be able to carry the induction with respect to the rank r of the derivation.

We use the inductive assuption in all cases except, as before, a case of an *axiom* or *structural rules*. In these cases the mix can be eliminated immediately, as it was eliminated in the previous case of rank $r = 2$.

In a case of *logical rules* we obtain the reduction of the mix of the lemma to derivations with mix of a lower ranks which consequently can be eleminated by the inductive assumption. We carry now proofs for two logical rules: $(\rightarrow \cap)$ and $(\cup \rightarrow$. The proof for all other rules is similar and is left to the reader. Also, we consider a case of left rank Lr= 1 and the right rank Rr = r ¿1. The symmetrical case left rank Lr = r ¿1 1 and the right rank Rr = 1 is left to the reader as an exercise.

Case: $Lr = 1, Rr = r > 1$. The right premiss of the mix is a conclusion of the rule $(\rightarrow \cap)$, i.e. it is of a form $\Gamma \longrightarrow \Delta, (A \cap B)$ and Γ contains the mix formula M. The left premiss of the mix is a sequent $\Theta \longrightarrow \Sigma$ and Θ contains the mix formula M. The end of the derivation **D**, .i.e. the sub-tree **T** of **D** containing mix is:

$$\Theta,\ \Gamma^* \ \longrightarrow\ \Sigma^*, \Delta, (A \cap B)$$

$$\bigwedge(mix)$$

$$\Theta \longrightarrow \Sigma \qquad\qquad\qquad \Gamma \longrightarrow\ \Delta,\ (A \cap B)$$

$$\qquad\qquad\qquad\qquad\bigwedge(\rightarrow \cap)$$

$$\qquad\qquad\qquad\qquad \Gamma \longrightarrow\ \Delta, A \qquad \Gamma \longrightarrow\ \Delta, B$$

We transform **T** into **T*** as follows.

$$\Theta,\ \Gamma^* \longrightarrow \Sigma^*, \Delta, (A \cap B)$$

$$\bigwedge(\rightarrow \cap)$$

$$\begin{array}{ccc} \Theta,\ \Gamma^* \longrightarrow \Sigma^*, \Delta, A & \qquad & \Theta,\ \Gamma^* \longrightarrow \Sigma^*, \Delta, B \\ \bigwedge(mix) & & \bigwedge(mix) \\ \Theta \longrightarrow \Sigma \qquad \Gamma \longrightarrow \Delta, A & & \Theta \longrightarrow \Sigma \qquad \Gamma \longrightarrow \Delta, A \end{array}$$

We replace **T** by $\mathbf{T}^*$ in **D** and obtain $\mathbf{D}^*$. Now we have two mixes, but both have the right rank Rr = r-1 and both of them can be eliminated by the inductive assumption.

Case: $Lr = 1, Rr = r > 1$. The right premiss of the mix is a conclusion of the rule $(\cup \rightarrow$, i.e. it is of a form $(A \cup B), \Gamma \longrightarrow \Delta$ and Γ contains the mix formula M. The left premiss of the mix is a sequent $\Theta \longrightarrow \Sigma$ and Θ contains the mix formula M. The end of the derivation **D**, .i.e. the sub-tree **T** of **D** containing mix is:

$$\Theta,\ (A \cup B)^*, \Gamma^* \longrightarrow \Sigma^*, \Delta, (A \cap B)$$

$$\bigwedge(mix)$$

$$\begin{array}{ccc} \Theta \longrightarrow \Sigma & \qquad & (A \cup B)\Gamma \longrightarrow \Delta \\ & & \bigwedge(\cup \rightarrow) \\ & & A, \Gamma \longrightarrow \Delta \qquad B, \Gamma \longrightarrow \Delta \end{array}$$

$(A \cup B)^*$ stands either for or for nothing according as $(A \cup B)$ is unequal or equal to the mix formula M. The mix formula M certainly occurs in Γ. For otherwise M would been equal to $(A \cup B)$ and the right rank Rr would be equal to 1 contrary to the assumption.
We transform **T** into $\mathbf{T}^*$ as follows.

$$\Theta,\ (A \cup B), \Gamma^* \longrightarrow \Sigma^*, \Delta, (A \cap B)$$

$$\bigwedge(\cup \rightarrow)$$

$$
\begin{array}{cc}
\begin{array}{c}
A, \Theta,\ \Gamma^* \ \longrightarrow\ \Sigma^*, \Delta \\
\text{some weakenings, exchanges} \\
\Theta, A^*, \Gamma^* \ \longrightarrow\ \Sigma^*, \Delta \\
\bigwedge(mix) \\
\\
\Theta \longrightarrow \Sigma \qquad A, \Gamma \longrightarrow\ \Delta
\end{array}
&
\begin{array}{c}
B, \Theta,\ \Gamma^* \ \longrightarrow\ \Sigma^*, \Delta \\
\text{some weakenings, exchanges} \\
\Theta, B^*, \Gamma^* \ \longrightarrow\ \Sigma^*, \Delta \\
\bigwedge(mix) \\
\\
\Theta \longrightarrow \Sigma \qquad B, \Gamma \longrightarrow\ \Delta
\end{array}
\end{array}
$$

Now we have two mixes, but both have the right rank Rr = r-1 and both of them can be eliminated by the inductive assumption. We replace **T** by $\mathbf{T}^*$ in **D** and obtain $\mathbf{D}^*$. This **ends** the proof of the Hauptzatz lemma and hence the proof of the **Hauptzatz Theorem** 6.19 and **Hauptzatz Theorem** 6.20.

Let's denote by **LK - c** and **LI - c** the systems **LK**, **LI** without the cut rule, i.e. we put

$$\mathbf{LK} - \mathbf{c} = \mathbf{LK} - \{(cut)\}. \tag{6.37}$$

$$\mathbf{LI} - \mathbf{c} = \mathbf{LI} - \{(cut)\}. \tag{6.38}$$

We re-write the Hauptzatz Theorems as follows.

Theorem 6.21 (LK Hauptzatz)

For every LK sequent $\Gamma \longrightarrow\ \Delta$,

$$\vdash_{LK} \Gamma \longrightarrow\ \Delta \quad \textit{if and only if} \quad \vdash_{LK-c} \Gamma \longrightarrow\ \Delta.$$

Theorem 6.22 (LI Hauptzatz)

For every LI sequent $\Gamma \longrightarrow\ \Delta$,

$$\vdash_{LK} \Gamma \longrightarrow\ \Delta \quad \textit{if and only if} \quad \vdash_{LK-c} \Gamma \longrightarrow\ \Delta.$$

This is why the cut-free Gentzen systems **LK-c** and **LI -c** are just called Gentzen **LK, LI**, respectively.

Directly from the Completeness Theorem 6.18 and the Hauptzatz Theorem 6.19 we get that the following.

Theorem 6.23 (LK-c Completeness)

For any sequent $\Gamma \longrightarrow \Delta$,

$$\vdash_{\mathbf{LK-c}} \Gamma \longrightarrow \Delta \ \ \textit{if and only if} \quad \models\ \Gamma \longrightarrow \Delta.$$

Let **G** be the Gentzen sequents proof system defined by (6.27). We replace the logical axiom of **G**

$$\Gamma'_1,\ a,\ \Gamma'_2\ \longrightarrow\ \Delta'_1,\ a,\ \Delta'_2,$$

where $a \in VAR$ is any propositional variable and $\Gamma'_1, \Gamma'_2,\ \Delta'_1,\ \Delta'_2 \in VAR^*$ are any indecomposable sequences, by a new logical axiom

$$\Gamma_1,\ A,\ \Gamma_2\ \longrightarrow\ \Delta_1,\ A,\ \Delta_2 \tag{6.39}$$

for any $A \in \mathcal{F}$ and any sequences $\Gamma_1, \Gamma_2, \Delta_1, \Delta_2 \in SQ$. We call a resulting proof system **GK**, i.e. we have that

$$\mathbf{GK} = (\ \mathcal{L}_{\{\cup,\cap,\Rightarrow,\neg\}},\ SQ,\ LA,\ \mathcal{R}\ ) \tag{6.40}$$

where LA is the axiom (6.39) and $\mathcal{R}$ is the set (6.29) of rules of **G**.

Observe that the only difference between the systems **GK** and **G** is the form of their logical axioms, both being tautologies. Hence hence get the proof completeness of **GK** in the same way as we proved it for **G**, i.e. we have the following.

Theorem 6.24

For any formula $A \in \mathcal{F}$,

$$\vdash_{\mathbf{GK}} A \quad \textit{if and only if} \quad \models\ A.$$

For any sequent $\Gamma \longrightarrow \Delta \in SQ$,

$$\vdash_{\mathbf{GK}} \Gamma \longrightarrow \Delta \qquad \textit{if and only if} \qquad \models\ \Gamma \longrightarrow \Delta.$$

By the **GK** the completeness theorem 6.24, **LK-c** completeness theorem 6.23 we get the equivalency of **GK** and the cut free **LK-c**.

Theorem 6.25 (LK, GK Equivalency)

The proof systems **GK** *and the cut free***LK** *are equivalent, i.e for any sequent $\Gamma \longrightarrow \Delta$,*

$$\vdash_{\mathbf{LK}} \Gamma \longrightarrow \Delta \quad \textit{if and only if} \quad \vdash_{\mathbf{GK}} \Gamma \longrightarrow \Delta.$$

6.8 Homework Problems

1. Write all proofs in **GL** of $(\neg(a \cap b) \Rightarrow (\neg a \cup \neg b)))$.
2. Find a formula which has a unique decomposition tree in **GL**.
3. Define shortly, in your own words, for any formula $A \in \mathcal{F}$, its decomposition tree $\mathbf{T}_{\rightarrow A}$ in **G**.

4. Extend your definition $\mathbf{T}_{\rightarrow A}$ in $\mathbf{G}$ to a decomposition tree $\mathbf{T}_{\Gamma\rightarrow\Delta}$.

5. Prove that for any $\Gamma \rightarrow \Delta \in SQ$, the decomposition tree $\mathbf{T}_{\Gamma\rightarrow\Delta}$ in $\mathbf{G}$ are finite.

6. Write all proofs in $\mathbf{G}$ of $(\neg(a \cap b) \Rightarrow (\neg a \cup \neg b)))$.

7. Find a formula A which has a unique decomposition tree in $\mathbf{G}$.

8. Prove strong soundness of rules $(\rightarrow \cup), (\rightarrow\Rightarrow)$ in $\mathbf{GL}$. List all logical equivalences used in the proofs.

9. Prove strong soundness of rules $(\Rightarrow\rightarrow), (\neg \rightarrow)$ in $\mathbf{GL}$. List all logical equivalences used in the proofs.

10. Prove strong soundness of rules $(\cup \rightarrow), (\rightarrow \neg), (\cap \rightarrow)$ in $\mathbf{G}$. List all logical equivalences used in the proofs.

11. Prove strong soundness of rules $(\Rightarrow\rightarrow), (\rightarrow \cup), (\Rightarrow\rightarrow)$ in $\mathbf{G}$. List all logical equivalences used in the proofs.

12. Explain why the system $\mathbf{G}$ is strongly sound.

13. Prove the following.

    For any sequent $\Gamma \longrightarrow \Delta \in SQ$,
    if $\vdash_{\mathbf{G}} \Gamma \longrightarrow \Delta$, then $\models \Gamma \longrightarrow \Delta$.

14. Given a formula $A = ((b \Rightarrow (a \cap c)) \Rightarrow (\neg(a \cup c) \Rightarrow (\neg b \cup a)))$.

    (i) Find all counter models determined by the decomposition trees of A in $\mathbf{GL}$. Explain why the definition of a counter model determined by the decomposition tree is correct.

    (ii) Find all counter models determined by the decomposition trees of A in $\mathbf{G}$. Explain why the definition of a counter model determined by the decomposition tree is correct.

15. Prove the following.

    Given a sequent $\Gamma \longrightarrow \Delta$, such that its decomposition tree $\mathbf{T}_{\Gamma\longrightarrow\Delta}$ in $\mathbf{G}$ contains a non- axiom leaf L_A. Any truth assignment v that falsifies the non-axiom leaf L_A is a counter model for $\Gamma \longrightarrow \Delta$.

16. Prove the following.

    For any sequent $\Gamma \longrightarrow \Delta \in SQ$,
    $\vdash_{\mathbf{G}} \Gamma \longrightarrow \Delta$ if and only if $\models \Gamma \longrightarrow \Delta$.

17. Let **LK-c** = **LK** $- \{(cut)\}$ and **GK** be proof systems defined as defined by (6.37) and (6.40), respectively

    (i) We know that **GK** is strongly sound. Prove that **LK-c** is sound but not strongly sound.

(ii) Find proofs of axioms A3, A7, and A11 of the R system (6.34) in **LK-c** and in **GK**, i.e. proofs of formulas $(B \Rightarrow (A \cup B))$, $((C \Rightarrow A) \Rightarrow ((C \Rightarrow B) \Rightarrow (C \Rightarrow (A \cap B)))$, and $((A \Rightarrow (A \cap \neg A)) \Rightarrow \neg A)$. Compare your results.

(iii) Find proofs of axioms A1, A8, and A9 of the R system (6.34) in **LK-c** and in **GK**, i.e. proofs of formulas $((A \Rightarrow B) \Rightarrow ((B \Rightarrow C) \Rightarrow (A \Rightarrow C)))$, $((A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \cap B) \Rightarrow C))$, and $(((A \cap B) \Rightarrow C) \Rightarrow (A \Rightarrow (B \Rightarrow C))$. Compare your results.

(iv) Find proofs of axioms A1, A5, and A12 of the R system (6.34) in **LK-c** and in **GK**, i.e. proofs of formulas $((A \Rightarrow B) \Rightarrow ((B \Rightarrow C) \Rightarrow (A \Rightarrow C))), ((A \cap B) \Rightarrow A)$, and $(A \cup \neg A)$. Compare your results.

18. Re- write carefully the proof of the classical Hauptzatz Theorem 6.19 for the case o the intuitionistic system **LI** (definition 6.10.

19. Define shortly, in your own words, for any formula $A \in \mathcal{F}$, its decomposition tree $\mathbf{T}_A$ in **LK-c**. Is the tree $\mathbf{T}_A$ always finite?

20. Given a formula $A = (\neg(a \cap b) \Rightarrow (\neg a \cup \neg b)))$. Construct one infinite and one infinite decomposition tree for A.

21. Describe major differences in the decomposition trees in **LK-c** and **GK**.

22. We have proved that **LK-c** and **GK** are equivalent, i.e. that for any sequent $\Gamma \longrightarrow \Delta$,

$$\vdash_{\mathbf{LK-c}} \Gamma \longrightarrow \Delta \text{ if and only if } \vdash_{\mathbf{GK}} \Gamma \longrightarrow \Delta.$$

The proof was not constructive; it was obtained from that fact that both systems are complete.

(ii) Describe a constructive procedure of transforming any proof in **GK** into a proof in **LK-c**.

(i) Transform a proof of a formula $(A \Rightarrow (A \cup B))$ in **GK** into a proof in **LK-c**.

(ii) Describe a constructive procedure of transforming any proof in **GK** into a proof in **LK-c**.

(iii) Show that the procedure of elimination of structural rules of **LK-c** leads to the rules inference of **GK** .

Chapter 7

Introduction to Intuitionistic and Modal Logics

7.1 Introduction to Intuitionictic Logic

Intuitionistic logic has developed as a result of certain philosophical views on the foundation of mathematics, known as *intuitionism.* Intuitionism was originated by L. E. J. Brouwer in 1908. The first Hilbert style formalization of the intuitionistic logic, formulated as a proof system, is due to A. Heyting (1930). In this chapter we present a Hilbert style proof system I that is equivalent to the Heyting's original formalization and discuss the relationship between intuitionistic and classical logic.

There have been, of course, several successful attempts at creating semantics for the intuitionistic logic,i.e. to define formally a notion of the intuitionistic tautology. The most recent are Kripke models were defined by Kripke in 1964. The first intuitionistic semantics was defined in a form of pseudo- Boolean algebras by McKinsey, Tarski in 1944 - 1946. McKinsey, Tarski algebraic approach to the intuitionostic semantics (and classical) was followed by many authors and developed into a new field of *Algebraic Logic.* The pseudo- Boolean algebras are called also Heyting algebras.

An uniform presentation of algebraic models for classical, intuitionistic and modal logics was first given in a now classic algebraic logic book: *"Mathematics of Metamathematics"*, Rasiowa, Sikorski (1964).

The goal of this chapter is to give a presentation of the intuitionistic logic formulated as a proof system, discuss its algebraic semantics and the basic theorems that establish the relationship between classical and intuitionistic logics.

7.1.1 Philosophical Motivation

Intuitionists' view-point on the meaning of the basic logical and set theoretical concepts used in mathematics is different from that of most mathematicians in their research.

The basic difference lies in the interpretation of the word *exists*. For example, let $A(x)$ be a statement in the arithmetic of natural numbers. For the mathematicians the sentence

$$\exists x A(x) \tag{7.1}$$

is true if it is a theorem of arithmetic, i.e. if it can be *deduced* from the axioms of arithmetic by means of classical logic. If a mathematician proves sentence (7.1), this does not mean that he is able to indicate a *method of construction* of a natural number n such that $A(n)$ holds.

For the intuitionist the sentence (7.1) is true only he is able to provide a constructive method of finding a number n such that $A(n)$ is true.

Moreover, the mathematician often obtains the proof of the existential sentence (7.1), i.e. of the sentence $\exists x A(x)$ by proving first a sentence

$$\neg\forall x\ \neg A(x). \tag{7.2}$$

Next he makes use of a classical tautology

$$(\neg\forall x\ \neg A(x)) \Rightarrow \exists x A(x)). \tag{7.3}$$

By applying Modus Ponens to (7.2) and (7.3) he obtains (7.1).

For the intuitionist such method is not acceptable, for it does not give any *method of constructing* a number n such that $A(n)$ holds. For this reason the intuitionist do not accept the classical tautology (7.3) i.e. $(\neg\forall x\ \neg A(x)) \Rightarrow \exists x A(x))$ as intuitionistic tautology, or as as an intuitionistically provable sentence.

Let us denote by $\vdash_I A$ and $\models_I A$ the fact that A is intuitionistically provable and that A is intuitionistic tautology, respectively. The proof system I for the intuitionistic logic has hence to be such that

$$\nvdash_I (\neg\forall x\ \neg A(x)) \Rightarrow \exists x A(x)).$$

The intuitionistic semantics I has to be such that one can prove in that also

$$\not\models_I (\neg\forall x\ \neg A(x)) \Rightarrow \exists x A(x)).$$

The above means also that intuitionists interpret differently the meaning of propositional connectives.

Intuitionistic implication

The intuitionistic implication $(A \Rightarrow B)$ is considered by to be true if there exists a method by which a *proof of* B can be deduced from the proof of A. In the case of the implication

$$(\neg\forall x\ \neg A(x)) \Rightarrow \exists x A(x))$$

there is no general method which, from a proof of the sentence $(\neg\forall x\ \neg A(x))$, permits is to obtain an intuitionistic proof of the sentence $\exists x A(x)$, i.e. to construct a number n such that $A(n)$ holds, hence we can't accept it as an intuitionistic theorem or tautology.

Intuitionistic negation

The negation and the disjunction are also understood differently. The sentence $\neg A$ is considered intuitionistically true if the acceptance of the sentence A leads to absurdity.

As a result of above understanding of negation and implication we have that in the intuitionistic logic I

$$\vdash_I (A \Rightarrow \neg\neg A)$$

but

$$\nvdash_I (\neg\neg A \Rightarrow A).$$

Consequently, the intuitionistic semantics I has to be such that

$$\models_I (A \Rightarrow \neg\neg A)$$

and

$$\not\models_I (\neg\neg A \Rightarrow A).$$

Intuitionistic disjunction

The intuitionist regards a disjunction $(A \cup B)$ as true if one of the sentences A, B is true and there is a method by which it is possible to find out which of them is true. As a consequence a classical law of excluded middle

$$(A \cup \neg A)$$

is not acceptable by the intuitionists since there is no general method of finding out, for any given sentence A, whether A or $\neg A$ is true. This means that the intuitionistic logic must be such that

$$\nvdash_I (A \cup \neg A)$$

and the intuitionistic semantics I has to be such that

$$\not\models_I (A \cup \neg A).$$

Intuitionists' view of the concept of infinite set also differs from that which is generally accepted in mathematics. Intuitionists reject the idea of infinite set as a closed whole. They look upon an infinite set as something which is constantly in a state of formation. Thus, for example, the set of all natural numbers is infinite in the sense that to any given finite set of natural numbers it is always possible to add one more natural number. The notion of the set of all subsets of the set of all natural numbers is not regarded meaningful. Thus intuitionists reject the general idea of a set as defined by a modern set theory.

An exact exposition of the basic ideas of intuitionism is outside the range of our investigations. Our goal is to give a presentation of of the intuitionistic logic, which is a sort of reflection of intuitionistic ideas formulated as a proof system.

7.1.2 Algebraic Intuitionistic Semantics and Completeness Theorem

There are many proof systems describing the intuitionistic logic. We define now a system I with a set of axioms that is due to Rasiowa (1959). We adopted this axiomatization for two reasons. Firs is that it is the most natural and appropriate set of axioms to carry the the algebraic proof of the completeness theorem and the second is that they visibly describe the main difference between intuitionistic and classical logic. Namely, by adding the only one more axiom $(A \cup \neg A)$ we get a (complete) formalization for classical logic. Here are the components if the proof system I.

Language We adopt a propositional language $\mathcal{L} = \mathcal{L}_{\{\cup,\cap,\Rightarrow,\neg\}}$ with the set of formulas denoted by $\mathcal{F}$.

Axioms

A1 $((A \Rightarrow B) \Rightarrow ((B \Rightarrow C) \Rightarrow (A \Rightarrow C)))$,

A2 $(A \Rightarrow (A \cup B))$,

A3 $(B \Rightarrow (A \cup B))$,

A4 $((A \Rightarrow C) \Rightarrow ((B \Rightarrow C) \Rightarrow ((A \cup B) \Rightarrow C)))$,

A5 $((A \cap B) \Rightarrow A)$,

A6 $((A \cap B) \Rightarrow B)$,

A7 $((C \Rightarrow A) \Rightarrow ((C \Rightarrow B) \Rightarrow (C \Rightarrow (A \cap B)))$,

A8 $((A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \cap B) \Rightarrow C))$,

A9 $(((A \cap B) \Rightarrow C) \Rightarrow (A \Rightarrow (B \Rightarrow C))$,

A10 $(A \cap \neg A) \Rightarrow B)$,

A11 $((A \Rightarrow (A \cap \neg A)) \Rightarrow \neg A)$, where A, B, C are any formulas in $\mathcal{L}$.

Rules of inference

We adopt the Modus Ponens rule

$$(MP)\ \frac{A\ ;\ (A \Rightarrow B)}{B}$$

as the only inference rule.

A proof system

$$\mathbf{I} = (\ \mathcal{L}, \mathcal{F}\ A1 - A11,\ (MP)\ ), \tag{7.4}$$

$A1 - A11$ defined above, is called a Hilbert style formalization for Intuitionistic propositional logic.

We introduce, as usual, the notion of a formal proof in I and denote by

$$\vdash_I A$$

the fact that A has a formal proof in I, or that that A is *intuitionistically provable* in I.

7.1.3 Algebraic Semantics and Completeness Theorem

We shortly present here Tarski, Rasiowa, and Sikorski psedo-Boolean algebra semantics and discuss the algebraic completeness theorem for the intuitionistic propositional logic. We leave the Kripke semantics for the reader to explore from other, multiple sources.

Here are some algebraic basic definitions.

Relatively Pseudo-Complemented Lattice
A lattice $(B, \cap, \cup)$ is said to be *relatively pseudo-complemented* (Birkhoff, 1935) if for any elements $a, b \in B$, there exists the greatest element c, such that $a \cap c \leq b$. Such element is denoted by $a \Rightarrow b$ and called the *pseudo-complement of a relative to b.* By definition

$$x \leq a \Rightarrow b \text{ if and only if } a \cap x \leq b \text{ for all } x, a, b \in B. \tag{7.5}$$

The equation (7.5) can serve as the definition of the **relative pseudo-complement** $a \Rightarrow b$.

Fact 7.1 *Every relatively pseudo-complemented lattice $(B, \cap, \cup)$ has the greatest element, called a unit element and denoted by 1.*

Proof Observe that $a \cap x \leq a$ for all $x, a \in B$. By (7.5) we have that $x \leq a \Rightarrow a$ for all $x \in B$, i.e. $a \Rightarrow a = 1$.

An abstract algebra

$$\mathcal{B} = (B,\ 1,\ \Rightarrow,\ \cap,\ \cup, \Rightarrow) \tag{7.6}$$

is said to be a **relatively pseudo-complemented lattice** if $(B, \cap, \cup)$ is *relatively pseudo-complemented lattice* with the *relative pseudo-complement* $\Rightarrow$ defined by (7.5) and with the unit element 1 (Fact 7.1).

Relatively Pseudo-complemented Set Lattices

Consider a topological space X with an interior operation I. Let $\mathcal{G}(X)$ be the class of all open subsets of X and $\mathcal{G}^*(X)$ be the class of all both dense and open subsets of X. Then the algebras

$$(\mathcal{G}(X),\ X,\ \cup,\ \cap, \Rightarrow), \quad (\mathcal{G}^*(X),\ X,\ \cup,\ \cap, \Rightarrow),$$

where $\cup$, $\cap$ are set-theoretical operations of union, intersection, and $\Rightarrow$ is defined by

$$Y \Rightarrow Z = I(X - Y) \cup Z$$

are relatively pseudo-complemented lattices.
Clearly, all sub algebras of these algebras are also relatively pseudo-complemented lattices, called *relatively pseudo-complemented set lattices.* They are typical examples of relatively pseudo-complemented lattices

Pseudo - Boolean Algebra (Heyting Algebra)
An algebra

$$\mathcal{B} = (B,\ 1,\ 0,\ \Rightarrow,\ \ \cap,\ \cup, \neg), \tag{7.7}$$

is said to be a *pseudo - Boolean algebra* if and only if $(B,\ 1,\ \Rightarrow,\ \ \cap,\ \cup)$ it is a relatively pseudo-complemented lattice (7.6) in which a zero element 0 exists and $\neg$ is a one argument operation defined as follows

$$\neg a = a \Rightarrow 0 \tag{7.8}$$

The operation $\neg$ is called a **pseudo-complementation**.

The pseudo - Boolean algebras are also called **Heyting algebras** to stress their connection to the intuitionistic logic.

Let X be topological space with an interior operation I. Let $\mathcal{G}(X)$ be the class of all open subsets of X. Then

$$(\mathcal{G}(X),\ X,\ \emptyset,\ \cup,\ \cap, \Rightarrow,\ \neg), \tag{7.9}$$

where $\cup$, $\cap$ are set-theoretical operations of union, intersection, $\Rightarrow$ is defined by

$$Y \Rightarrow Z = I(X - Y) \cup Z$$

and $\neg$ is defined as

$$\neg Y = Y \Rightarrow \emptyset = I(X - Y), \quad \text{for all } Y \subseteq X$$

is a pseudo - Boolean algebra.
Every sub algebra of $\mathcal{G}(X)$ is also a pseudo-Boolean algebra. They are called **pseudo-fields of sets.**

The following Theorem 7.1 states that pseudo-fields are typical examples of pseudo - Boolean algebras. The theorems of this type are often called **Stone Representation Theorems** to remember an American mathematician H.M Stone. He was one of the first to initiate the investigations between logic and general topology in the article *"The Theory of Representations for Boolean Algebras"*,Trans. of the Amer.Math, Soc 40, 1036.

Theorem 7.1 (Representation Theorem) *(McKinsey, Tarski, 1946)*

For every pseudo - Boolean (Heyting) algebra (7.7)

$$\mathcal{B} = (B,\ 1,\ 0,\ \Rightarrow,\ \ \cap,\ \cup, \neg),$$

there exists a monomorphism h of $\mathcal{B}$ into a pseudo-field (7.9) $\mathcal{G}(X)$ of all open subsets of a compact topological T_0 space X.

Another typical (and interesting) example of a class of pseudo - Boolean algebras is the following.

Linear Pseudo - Boolean Algebra

Let $(B, \leq)$ be a chain (linearly ordered set) with the greatest element 1 and the least element (smallest) 0.
An algebra

$$\mathcal{H} = (B,\ 1,\ 0,\ \Rightarrow,\ \ \cap,\ \cup, \neg) \tag{7.10}$$

is called a *linear pseudo - Boolean algebra* if and only if its operations are defined as follows.

For any $a, b \in B$,

$$a \cup b = max\{a,\ b\}, \qquad a \cap b = min\{a,\ b\},$$

$$a \Rightarrow b = \begin{cases} 1 & \text{if } a \leq b \\ b & \text{otherwise,} \end{cases} \tag{7.11}$$

and define the *pseudo-complementation* $\neg$ as

$$\neg a = a \Rightarrow 0.$$

We leave the proof that (7.10) is a pseudo-Boolean algebra as a homework exercise. Observe that the linear pseudo - Boolean algebra (7.10) is a generalization of the 3-valued Heyting semantics defined in chapter 3.

Algebraic Models

We say that a formula A is an **intuitionistoc tautology** if and only if any pseudo-Boolean algebra (7.7) is a *model* for A. This kind of models because their connection to abstract algebras are called **algebraic models**.

We put it formally as follows.

Definition 7.1 (Intuitionistic Algebraic Model)

Let A be a formula of the language $\mathcal{L}_{\{\cup,\cap,\Rightarrow,\neg\}}$ and let $\mathcal{B} = (B,\ 1,\ 0,\ \Rightarrow,\ \cap,\ \cup, \neg)$ be a pseudo - Boolean topological Boolean algebra (7.7).

We say that the algebra $\mathcal{B}$ is a **model** *for the formula A and denote it by*

$$\mathcal{B} \models A$$

if and only if $v^*(A) = 1$ holds for all variables assignments $v : VAR \longrightarrow B$.

Definition 7.2 (Intuitionistic Tautology)

The formula A is an **intuitionistic tautology** *and is denoted by*

$$\models_I A \quad \textit{if and only if}\ \mathcal{B} \models A$$

for all pseudo-Boolean algebras $\mathcal{B}$.

In Algebraic Logic the notion of tautology is often defined using a notion "a formula A is **valid** in an algebra $\mathcal{B}$ ". It is formally defined as follows.

Definition 7.3

A formula A **is valid** *in a pseudo-Boolean algebra $\mathcal{B} = (B,\ 1,\ 0,\ \Rightarrow,\ \cap,\ \cup, \neg)$, if and only if $v^*(A) = 1$ holds for all variables assignments $v : VAR \longrightarrow B$.*

Directly from definition 7.2 and definition 7.3 we get the following.

Fact 7.2 *For any formula A, $\models_I A$ if and only if A* **is valid** *in all pseudo-Boolean algebras.*

We write now $\vdash_I A$ to denote any proof system for the Intuitionistic propositional logic, and in particular the Hilbert style formalization for Intuitionistic propositional logic I defined by (7.4).

Theorem 7.2 (Intuitionistic Completeness Theorem) *(Mostowski 1948)*

For any formula A of $\mathcal{L}_{\{\cup,\ \cap,\Rightarrow,\neg\}}$,

$$\vdash_I A \quad \textit{if and only if} \quad \models_I A.$$

The intuitionistic completeness theorem 7.2 follows also directly from the following general algebraic completeness theorem 7.3 that combines results of of Mostowski (1958), Rasiowa (1951) and Rasiowa-Sikorski (1957).

Theorem 7.3 (Algebraic Completeness Theorem)

For any formula A of $\mathcal{L} = \mathcal{L}_{\{\cup,\cap,\Rightarrow,\neg\}}$ the following conditions are equivalent.

(i) $\vdash_I A$,

(ii) $\models_I A$,

(iii) *A is valid in every pseudo-Boolean algebra*

$$(\mathcal{G}(X),\ X,\ \emptyset,\ \cup,\ \cap, \Rightarrow,\ \neg)$$

of open subsets of any topological space X,

(iv) *A is valid in every pseudo-Boolean algebra $\mathcal{B}$ with at most 2^{2^r} elements, where r is the number of all sub formulas of A.*

Moreover, each of the conditions (i) - (iv) *is equivalent to the following one.*

(v) *A is valid in the pseudo-Boolean algebra $(\mathcal{G}(X),\ X,\ \emptyset,\ \cup,\ \cap, \Rightarrow,\ \neg)$ of open subsets of a dense-in -itself metric space $X \neq \emptyset$ (in particular of an n-dimensional Euclidean space X).*

The following theorem follows from the equivalence of conditions (i) and (iv).

Theorem 7.4 (Decidability)

Every proof system for the intuitionistic propositional logic is decidable.

Examples of intuitionistic propositional tautologies

The following classical tautologies are provable in I and hence are also intuitionistic tautologies.

$$(A \Rightarrow A), \tag{7.12}$$

$$(A \Rightarrow (B \Rightarrow A)), \tag{7.13}$$

$$(A \Rightarrow (B \Rightarrow (A \cap B))), \tag{7.14}$$

$$((A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C))), \tag{7.15}$$

$$(A \Rightarrow \neg\neg A), \tag{7.16}$$

$$\neg(A \cap \neg A), \tag{7.17}$$

$$((\neg A \cup B) \Rightarrow (A \Rightarrow B)), \tag{7.18}$$

$$(\neg(A \cup B) \Rightarrow (\neg A \cap \neg B)), \tag{7.19}$$

$$((\neg A \cap \neg B) \Rightarrow (\neg(A \cup B)), \tag{7.20}$$

$$((\neg A \cup \neg B) \Rightarrow (\neg A \cap \neg B)), \tag{7.21}$$

$$((A \Rightarrow B) \Rightarrow (\neg B \Rightarrow \neg A)), \tag{7.22}$$

$$((A \Rightarrow \neg B) \Rightarrow (B \Rightarrow \neg A)), \tag{7.23}$$

$$(\neg\neg\neg A \Rightarrow \neg A), \tag{7.24}$$

$$(\neg A \Rightarrow \neg\neg\neg A), \tag{7.25}$$

$$(\neg\neg(A \Rightarrow B) \Rightarrow (A \Rightarrow \neg\neg B)), \tag{7.26}$$

$$((C \Rightarrow A) \Rightarrow ((C \Rightarrow (A \Rightarrow B)) \Rightarrow (C \Rightarrow B)), \tag{7.27}$$

Examples of classical tautologies that are not intuitionistic tautologies

The following classical tautologies are not intuitionistic tautologies.

$$(A \cup \neg A), \tag{7.28}$$

$$(\neg\neg A \Rightarrow A), \tag{7.29}$$

$$((A \Rightarrow B) \Rightarrow (\neg A \cup B)), \tag{7.30}$$

$$(\neg(A \cap B) \Rightarrow (\neg A \cup \neg B)), \tag{7.31}$$

$$((\neg A \Rightarrow B) \Rightarrow (\neg B \Rightarrow A)), \tag{7.32}$$

$$((\neg A \Rightarrow \neg B) \Rightarrow (B \Rightarrow A)), \tag{7.33}$$

$$((A \Rightarrow B) \Rightarrow A) \Rightarrow A), \tag{7.34}$$

7.1.4 Connection Between Classical and Intuitionistic Tautologies

The intuitionistic logic has been created as a rival to the classical one. So a question about the relationship between these two is a natural one. We present here some examples of tautologies and some historic results about the connection between the classical and intuitionistic logic.

The first connection is quite obvious. It was proved by Rasiowa and Sikorski in 1964 that by adding the axiom

A12 $(A \cup \neg A)$

to the set of axioms of our system I defined by (7.4) we obtain a Hilbert proof system H that is **complete** with respect to classical semantics.

This proves the following.

Theorem 7.5

Every formula that is derivable intuitionistically is classically derivable, i.e.

$$\textit{if} \quad \vdash_I A, \qquad \textit{then} \quad \vdash A,$$

where we use symbol $\vdash$ *for classical (complete classical proof system) provability.*

We write

$$\models\ A \quad \text{and}\ ] \ \models_I A$$

to denote that A is a classical and intuitionistic tautology, respectively.

As both proof systems, I and H are complete under respective semantics, we can state this as the following relationship between classical and intuitionistic tautologies.

Theorem 7.6

For any formula $A \in \mathcal{F}$*,*

$$\textit{if} \quad \models_I A, \quad \textit{then} \quad \models\ A.$$

The next relationship shows how to obtain intuitionistic tautologies from the classical tautologies and vice versa. It has been proved by Glivenko in 1929 and independently by Tarski in 1938.

Theorem 7.7 (Glivenko, Tarski)

For any formula $A \in \mathcal{F}$*,* A *is a classically provable if and only if* $\neg\neg A$ *is an intuitionistically provable, i.e.*

$$\vdash_I A \quad \textit{if and only if} \quad \vdash \neg\neg A$$

where we use symbol $\vdash$ *for classical provability.*

Theorem 7.8 (McKinsey, Tarski, 1946)

For any formula $A \in \mathcal{F}$*,* A *is a classical tautology if and only if* $\neg\neg A$ *is an intuitionistic tautology, i.e.*

$$\models\ A \quad if\ and\ only\ if \quad \models_I \neg\neg A.$$

The following relationships were proved by Gödel.

Theorem 7.9 (Gödel, 1931)

For any $A, B \in \mathcal{F}$, a formula $(A \Rightarrow \neg B)$ is a classically provable if and only if it is an intuitionistically provable, i.e.

$$\vdash\ (A \Rightarrow \neg B)\ \ if\ and\ only\ if\ \ \vdash_I (A \Rightarrow \neg B).$$

Theorem 7.10 (Gödel, 1931)

If a formula A contains no connectives except $\cap$ and $\neg$, then A is a classically provable if and only if it is an intuitionistically provable.

By the completeness of classical and intuitionisctic logics we get the following equivalent semantic form of theorems 7.9 and 7.10.

Theorem 7.11

A formula $(A \Rightarrow \neg B)$ is a classical tautology if and only if it is an intuitionistic tautology, i.e.

$$\models\ (A \Rightarrow \neg B)\ \ if\ and\ only\ if\ \ \models_I (A \Rightarrow \neg B).$$

Theorem 7.12

If a formula A contains no connectives except $\cap$ and $\neg$, then A is a classical tautology if and only if it is an intuitionistic tautology.

On intuitionistically derivable disjunction

In a classical logic it is possible for the disjunction $(A \cup B)$ to be a tautology when neither A nor B is a tautology. The tautology $(A \cup \neg A)$ is the simplest example. This does not hold for the intuitionistic logic.

This fact was stated without the proof by Gödel in 1931 and proved by Gentzen in 1935 via his proof system **LI** which is presented in chapter 6 and discussed in the next section 7.2.

Theorem 7.13 (Gentzen 1935)

A disjunction $(A \cup B)$ is intuitionistically provable if and only if either A or B is intuitionistically provable , i.e.

$$\vdash_I (A \cup B)\ \ \ if\ and\ only\ if\ \ \vdash_I A\ \ or\ \ \vdash_I B.$$

We obtain, via the Completeness Theorem 7.2 the following equivalent semantic version of the above.

Theorem 7.14

A disjunction $(A \cup B)$ is intuitionistic tautology if and only if either A or B is intuitionistic tautology, i.e.

$$\models_I (A \cup B) \quad if\ and\ only\ if \quad \models_I A \quad or \quad \models_I B.$$

7.2 Gentzen Sequent System LI

In 1935 G. Gentzen formulated a first syntactically decidable formalizations for classical and intuitionistic logic and proved its equivalence with the Heyting's original Hilbert style formalization. He named his classical system **LK** (K for Klassisch) and intuitionistic system **LI** (I for Intuitionistisch). In order to prove the completeness of the system **LK** and proving the adequacy of **LI** he introduced a special rule, called *cut rule* that corresponds to the Modus Ponens rule in Hilbert proof systems. Then, as the next step he proved the now famous Gentzen *Hauptzatz*, called in English the *Cut Elimination Theorem.*

The Gentzen original proof system **LI** is a particular case of his proof system **LK** for the classical logic. Both of them were presented in chapter 6 together with the proof of the *Hauptzatz* for both, **LK** and **LI** systems.

The elimination of the cut rule and the structure of other rules makes it possible to define effective automatic procedures for proof search, what is impossible in a case of the Hilbert style systems.

The Gentzen system **LI** is defined as follows.

Language of LI

Let $SQ = \{ \Gamma \longrightarrow \Delta : \Gamma, \Delta \in \mathcal{F}^* \}$ be the set of all Gentzen sequents built out of the formulas of the language

$$\mathcal{L} = \mathcal{L}_{\{\cup,\cap,\Rightarrow,\neg\}} \tag{7.35}$$

and the additional symbol $\longrightarrow$.

In order to describe the the intuitionistic logic we deal, after Gentzen, only with sequents of the form $\Gamma \longrightarrow \Delta$, where Δ consists of at most one formula. I.e. we assume that all **LI** sequents are elements of a following subset ISQ of the set SQ of all sequents.

$$ISQ = \{\Gamma \longrightarrow \Delta : \quad \Delta \text{ consists of at most one formula } \}. \tag{7.36}$$

The set ISQ is called the set of all intuitionistic sequents; the **LI** sequents.

Axioms of LI

As the axioms of **LI** we adopt any sequent from the set ISQ defined by (7.36), which contains a formula that appears on both sides of the sequent arrow $\longrightarrow$, i.e any sequent of the form

$$\Gamma_1, A, \Gamma_2 \ \longrightarrow \ A, \tag{7.37}$$

for any formula $A \in \mathcal{F}$ of the language (7.35) and for any sequences $\Gamma_1, \Gamma_2 \in \mathcal{F}^*$.

Inference rules of LI

The set inference rules is divided into two groups: the structural rules and the logical rules. They are defined as follows.

Structural Rules of LI

Weakening

$$(\rightarrow weak) \ \frac{\Gamma \ \longrightarrow}{\Gamma \ \longrightarrow \ A} \ .$$

A is called the weakening formula.

Contraction

$$(contr \rightarrow) \ \frac{A, A, \Gamma \ \longrightarrow \ \Delta}{A, \Gamma \ \longrightarrow \ \Delta},$$

A is called the contraction formula , Δ contains at most one formula.

Exchange

$$(exchange \rightarrow) \ \frac{\Gamma_1, A, B, \Gamma_2 \ \longrightarrow \ \Delta}{\Gamma_1, B, A, \Gamma_2 \ \longrightarrow \ \Delta},$$

Δ contains at most one formula.

Logical Rules of LI

Conjunction rules

$$(\cap \rightarrow) \ \frac{A, B, \Gamma \ \longrightarrow \ \Delta}{(A \cap B), \Gamma \ \longrightarrow \ \Delta}, \qquad (\rightarrow \cap) \ \frac{\Gamma \ \longrightarrow \ A \ ; \ \Gamma \ \longrightarrow \ B}{\Gamma \ \longrightarrow \ (A \cap B)},$$

Δ contains at most one formula.

Disjunction rules

$$(\rightarrow \cup)_1 \ \frac{\Gamma \ \longrightarrow \ A}{\Gamma \ \longrightarrow \ (A \cup B)}, \qquad (\rightarrow \cup)_2 \ \frac{\Gamma \ \longrightarrow \ B}{\Gamma \ \longrightarrow \ (A \cup B)},$$

$$(\cup \rightarrow)\ \ \frac{A, \Gamma \ \longrightarrow\ \Delta\ ;\ B, \Gamma\ \longrightarrow\ \Delta}{(A \cup B), \Gamma\ \longrightarrow\ \Delta},$$

Δ contains at most one formula.

Implication rules

$$(\rightarrow\Rightarrow)\ \ \frac{A, \Gamma\ \longrightarrow\ B}{\Gamma\ \longrightarrow\ (A \Rightarrow B)}, \qquad (\Rightarrow\rightarrow)\ \ \frac{\Gamma\ \longrightarrow\ A\ ;\ B, \Gamma\ \longrightarrow\ \Delta}{(A \Rightarrow B), \Gamma\ \longrightarrow\ \Delta},$$

Δ contains at most one formula.

Negation rules

$$(\neg \rightarrow)\ \ \frac{\Gamma\ \longrightarrow\ A}{\neg A, \Gamma\ \longrightarrow\ }, \qquad (\rightarrow \neg)\ \ \frac{A, \Gamma\ \longrightarrow\ }{\Gamma\ \longrightarrow\ \neg A}.$$

Formally we define:

$$\mathbf{LI} = (\mathcal{L},\ ISQ,\ LA,\ \text{Structural rules},\ \ \text{Logical rules}\ ), \tag{7.38}$$

where ISQ is defined by (7.36), Structural rules and Logical rules are the inference rules defined above, and LA is the axiom defined by the schema (7.37).

We write

$$\vdash_{LI} \Gamma\ \longrightarrow\ \Delta$$

to denote that the sequent $\Gamma\ \longrightarrow\ \Delta$ has a proof in **LI**.

We say that a formula $A \in \mathcal{F}$ has a proof in **LI** and write it as

$$\vdash_{LI} A$$

when the sequent $\longrightarrow\ A$ has a proof in **LI**, i.e.

$$\vdash_{LI} A \quad \text{if and only if} \quad \vdash_{LI}\ \longrightarrow\ A.$$

The completeness of of our cut-free **LI** follows directly from LI Hauptzatz Theorem proved in chapter 6 and the Intuitionistic Completeness Theorem 7.2. The proof is a straightforward adaptation of the proof of cut free **LK** Completeness Theorem proved in chapter 6 and is left as a homework exercise.

Theorem 7.15 (Completeness of LI)

For any sequent $\Gamma\ \longrightarrow\ \Delta \in ISQ$*,*

$$\vdash_{LI} \Gamma\ \longrightarrow\ \Delta \quad \textit{if and only of} \quad \models_I \Gamma\ \longrightarrow\ \Delta.$$

In particular, for any formula A,

$$\vdash_{LI} A \quad \textit{if and only of} \quad \models_I A.$$

Theorem 7.16 (Intuitionistically Derivable Disjunction)

For any formulas A, B,

$$\vdash_{LI} (A \cup B) \quad \textit{if and only if} \quad \vdash_{LI} A \ \textit{ or } \vdash_{LI} B.$$

In particular, a disjunction $(A \cup B)$ is intuitionistically provable in any proof system I if and only if either A or B is intuitionistically provable in I.

The particular form the theorem 7.16 was stated without the proof by Gödel in 1931. The theorem proved by Gentzen in 1935 via his Hauptzatz Theorem.
Proof
Assume $\vdash_{LI} (A \cup B)$. This equivalent to $\vdash_{LI} \longrightarrow (A \cup B)$. The last step in the proof of $\longrightarrow (A \cup B)$i **LI** must be the application of the rule $(\rightarrow \cup)_1$ to the sequent $\longrightarrow A$, or the application of the rule $(\rightarrow \cup)_2$ to the sequent $\longrightarrow B$. There is no other possibilities. We have proved that $\vdash_{LI} (A \cup B)$ implies $\vdash_{LI} A$ or $\vdash_{LI} B$. The inverse is obvious by respective applications of rules $(\rightarrow \cup)_1$ $(\rightarrow \cup)_2$ to $\longrightarrow A$ and $\longrightarrow B$.

7.2.1 Decomposition Trees in LI

Search for proofs in **LI** is a much more complicated process then the one in classical systems **RS** or **GL** defined in chapter 6.

Here, as in any other Gentzen style proof system, proof search procedure consists of building the decomposition trees.

In **RS** the decomposition tree $\mathbf{T}_A$ of any formula A, and hence of any sequence Γ is always unique.

In **GL** the "blind search" defines, for any formula A a finite number of decomposition trees, but it can be proved that the search can be reduced to examining only one of them, due to the absence of structural rules.

In **LI** the structural rules play a vital role in the proof construction and hence, in the proof search. We consider here a number of examples to show the complexity of the problem of examining possible decomposition trees for a given formula A. We are going to see that the fact that a given decomposition tree ends with an axiom leaf does not always imply that the proof does not exist. It might only imply that our search strategy was not good. Hence the problem of deciding whether a given formula A does, or does not have a proof in **LI** becomes more complex then in the case of Gentzen system for classical logic.

Before we define a heuristic method of searching for proof and deciding whether such a proof exists or not in **LI** we make some observations.

Observation 1: the logical rules of **LI** are similar to those in Gentzen type classical formalizations we examined in previous chapters in a sense that each of them introduces a logical connective.

Observation 2: The process of searching for a proof is, as before a decomposition process in which we use the inverse of logical and structural rules as decomposition rules.

For example the implication rule:

$$(\rightarrow\Rightarrow)\ \frac{A, \Gamma \longrightarrow B}{\Gamma \longrightarrow (A \Rightarrow B)}$$

becomes an **implication decomposition rule** (we use the same name $(\rightarrow\Rightarrow)$ in both cases)

$$(\rightarrow\Rightarrow)\ \frac{\Gamma \longrightarrow (A \Rightarrow B)}{A, \Gamma \longrightarrow B}.$$

Observation 3: we write our proofs in as trees, instead of sequences of expressions, so the proof search process is a process of building a decomposition tree. To facilitate the process we write, as before, the decomposition rules, structural rules included in a "tree " form.

For example the the above implication decomposition rule is written as follows.

$$\begin{array}{c} \Gamma \longrightarrow (A \Rightarrow B) \\ \mid (\rightarrow\Rightarrow) \\ A, \Gamma \longrightarrow B \end{array}$$

The two premisses implication rule $(\Rightarrow\rightarrow)$ written as the tree decomposition rule becomes

$$\begin{array}{ccc} & (A \Rightarrow B), \Gamma \longrightarrow & \\ & \bigwedge(\Rightarrow\rightarrow) & \\ \Gamma \longrightarrow A & & B, \Gamma \longrightarrow \end{array}$$

For example the structural weakening rule is written as the decomposition rule is written as

$$(\rightarrow weak)\ \frac{\Gamma\ \longrightarrow\ A}{\Gamma\ \longrightarrow}$$

We write it in a tree form as follows.

$$\Gamma\ \longrightarrow\ A$$

$$\mid (\rightarrow weak)$$

$$\Gamma\ \longrightarrow$$

We define, as before the notion of decomposable and indecomposable formulas and sequents as follows.

Decomposable formula is any formula of the degree ≥ 1.

Decomposable sequent is any sequent that contains a decomposable formula.

Indecomposable formula is any formula of the degree 0, i.e. any propositional variable.

Remark: In a case of formulas written with use of capital letters A, B, C, .. etc, we treat these letters as propositional variables, i.e. as *indecomposable formulas.*

Indecomposable sequent is a sequent formed from indecomposable formulas only.

Decomposition tree construction (1): given a formula A we construct its decomposition tree $\mathbf{T_A}$ as follows.

Root of the tree is the sequent $\longrightarrow A$.

Given a node n of the tree we identify a decomposition rule applicable at this node and write its premisses as the leaves of the node n.

We stop the decomposition process when we obtain an axiom or all leaves of the tree are indecomposable.

Observation 4: the decomposition tree $\mathbf{T_A}$ obtained by the construction (1) most often is not unique.

Observation 5: the fact that we find a decomposition tree $\mathbf{T_A}$ with non-axiom leaf does not mean that $\nvdash_{LI} A$. This is due to the role of structural rules in **LI** and will be discussed later in the chapter.

We illustrate the problems arising with proof search procedures, i.e. decomposition trees construction in the next section 7.2.2 and give a heuristic proof searching procedure in the section 7.2.3.

7.2.2 Proof Search Examples

We perform proof search and decide the existence of proofs in **LI** for a given formula $A \in \mathcal{F}$ by constructing its decomposition trees $\mathbf{T_A}$. We examine here some examples to show the complexity of the problem.

Remark
In the following and similar examples when building the decomposition trees for formulas representing general schemas we treat the capital letters $A, B, C, D...$ as propositional variables, i.e. as *indecomposable formulas.*

Example 1

Determine whether $\vdash_{\mathbf{LI}} \longrightarrow ((\neg A \cap \neg B) \Rightarrow \neg(A \cup B))$.

This means that we have to construct some, or all decomposition trees of

$$\longrightarrow ((\neg A \cap \neg B) \Rightarrow \neg(A \cup B)).$$

If we find a decomposition tree such that all its leaves are axioms, we have a proof.

If all possible decomposition trees have a non-axiom leaf, proof of A in **LI** does not exist.

Consider the following decomposition tree of $\longrightarrow ((\neg A \cap \neg B) \Rightarrow \neg(A \cup B))$.

T1

$$\longrightarrow ((\neg A \cap \neg B) \Rightarrow (\neg(A \cup B))$$

$$\mid (\longrightarrow \Rightarrow)$$

$$(\neg A \cap \neg B) \longrightarrow \neg(A \cup B)$$

$$\mid (\longrightarrow \neg)$$

$$(A \cup B), (\neg A \cap \neg B) \longrightarrow$$

$$\mid (exch \longrightarrow)$$

$$(\neg A \cap \neg B), (A \cup B) \longrightarrow$$

$$\mid (\cap \longrightarrow)$$

$$\neg A, \neg B, (A \cup B) \longrightarrow$$

$$\mid (\neg \longrightarrow)$$

$$\neg B, (A \cup B) \longrightarrow A$$

$$\mid (\longrightarrow weak)$$

$$\neg B, (A \cup B) \longrightarrow$$

$$\mid (\neg \longrightarrow)$$

$$(A \cup B) \longrightarrow B$$

$$\bigwedge (\cup \longrightarrow)$$

$$A \longrightarrow B \qquad\qquad B \longrightarrow B$$

$$non - axiom \qquad\qquad axiom$$

The tree **T1** has a non-axiom leaf, so it does not constitute a proof in **LI**. But this fact does not yet prove that proof doesn't exist, as the decomposition tree in **LI** is not always unique.

Let's consider now the following tree.

T2

$$\longrightarrow ((\neg A \cap \neg B) \Rightarrow (\neg(A \cup B))$$

$$\mid (\longrightarrow \Rightarrow)$$

$$(\neg A \cap \neg B) \longrightarrow \neg(A \cup B)$$

$$\mid (\longrightarrow \neg)$$

$$(A \cup B), (\neg A \cap \neg B) \longrightarrow$$

$$\mid (exch \longrightarrow)$$

$$(\neg A \cap \neg B), (A \cup B) \longrightarrow$$

$$\mid (\cap \longrightarrow)$$

$$\neg A, \neg B, (A \cup B) \longrightarrow$$

$$\mid (exch \longrightarrow)$$

$$\neg A, (A \cup B), \neg B \longrightarrow$$

$$\mid (exch \longrightarrow)$$

$$(A \cup B), \neg A, \neg B \longrightarrow$$

$$\bigwedge (\cup \longrightarrow)$$

$$\begin{array}{cc}
A, \neg A, \neg B \longrightarrow & B, \neg A, \neg B \longrightarrow \\
\mid (exch \longrightarrow) & \mid (exch \longrightarrow) \\
\neg A, A, \neg B \longrightarrow & B, \neg B, \neg A \longrightarrow \\
\mid (\neg \longrightarrow) & \mid (exch \longrightarrow) \\
A, \neg B \longrightarrow A & \neg B, B, \neg A \longrightarrow \\
axiom & \mid (\neg \longrightarrow) \\
 & B, \neg A \longrightarrow B \\
 & axiom
\end{array}$$

All leaves of **T2** are axioms, what proves that **T2** is a proof of A and hence we proved that

$$\vdash_{\mathbf{LI}} ((\neg A \cap \neg B) \Rightarrow \neg(A \cup B)).$$

Example 2

Part 1: Prove that

$$\vdash_{\mathbf{LI}} \ \longrightarrow (A \Rightarrow \neg\neg A),$$

Part 2: Prove that

$$\nvdash_{\mathbf{LI}} \ \longrightarrow (\neg\neg A \Rightarrow A).$$

Solution of Part 1
To prove that

$$\vdash_{\mathbf{LI}} \ \longrightarrow (A \Rightarrow \neg\neg A)$$

we have to construct some, or all decomposition trees of

$$\longrightarrow (A \Rightarrow \neg\neg A).$$

We treat the sub formulas A, B as *indecomposable formulas.*

Consider the following decomposition tree.

$$\begin{array}{c}
\mathbf{T} \\
\longrightarrow (A \Rightarrow \neg\neg A). \\
\mid (\longrightarrow \Rightarrow) \\
A \longrightarrow \neg\neg A \\
\mid (\longrightarrow \neg) \\
\neg A, A \longrightarrow
\end{array}$$

$$\mid (\neg \longrightarrow)$$

$$A \longrightarrow A$$

$$axiom$$

All leaves of **T** are axioms what proves that **T** is a proof of $\longrightarrow (A \Rightarrow \neg\neg A)$ and we don't need to construct other decomposition trees.

Solution of Part 2
To prove that

$$\nvdash_{\mathbf{LI}} \longrightarrow (\neg\neg A \Rightarrow A)$$

we have to construct all decomposition trees of $(A \Rightarrow \neg\neg A)$ and show that each of them has an non-axiom leaf.

Consider the first decomposition tree defined as follows.

T1

$$\longrightarrow (\neg\neg A \Rightarrow A)$$

$$first\ of\ 2\ choices : (\rightarrow\Rightarrow), (\rightarrow weak)$$

$$\mid (\rightarrow\Rightarrow)$$

$$\neg\neg A \longrightarrow A$$

$$first\ of\ 2\ choices : (\rightarrow weak), (contr \rightarrow)$$

$$\mid (\rightarrow weak)$$

$$\neg\neg A \longrightarrow$$

$$first\ of\ 2\ choices : (\neg \rightarrow), (contr \rightarrow)$$

$$\mid (\neg \rightarrow)$$

$$\longrightarrow \neg A$$

$$first\ of\ 2\ choices : (\neg \rightarrow), (\rightarrow weak)$$

$$\mid (\rightarrow \neg)$$

$$A \longrightarrow$$

$$indecomposable$$

$$non-axiom$$

We use the first tree created to define all other possible decomposition trees by exploring the alternative search paths as indicated at the nodes of the tree.

T1

$\longrightarrow (\neg\neg A \Rightarrow A)$

$\mid (\longrightarrow\Rightarrow)$

$one\ of\ 2\ choices$

$\neg\neg A \longrightarrow A$

$]$

$\mid (contr \longrightarrow)$

$second\ of\ 2\ choices$

$\neg\neg A, \neg\neg A \longrightarrow A$

$\mid (\longrightarrow weak)$

$first\ of\ 2\ choices$

$\neg\neg A, \neg\neg A \longrightarrow$

$\mid (\neg \longrightarrow)$

$first\ of\ 2\ choices$

$\neg\neg A \longrightarrow \neg A$

$\mid (\longrightarrow \neg)$

$the\ only\ choice$

$A, \neg\neg A \longrightarrow$

$\mid (exch \longrightarrow)$

$the\ only\ choice$

$\neg\neg A, A \longrightarrow$

$\mid (\longrightarrow \neg)$

$the\ only\ choice$

$A \longrightarrow \neg A$

$\mid (\longrightarrow \neg)$

$first\ of\ 2\ choices$

$A, A \longrightarrow$

$indecomposable$

$non-axiom$

We can see from the above decomposition trees that the "blind" construction of all possible trees only leads to more complicated trees, due to the presence of structural rules. Observe that the "blind" application of ($contr \longrightarrow$) gives an infinite number of decomposition trees. To decide that none of them will produce a proof we need some extra knowledge about patterns of their construction, or just simply about the number useful of application of structural rules within the proofs.

In this case we can just make an "external" observation that the our first tree **T1** is in a sense a minimal one; that all other trees would only complicate this one in an inessential way, i.e. we will never produce a tree with all axioms leaves.

One can formulate a deterministic procedure giving a finite number of trees, but the proof of its correctness require some extra knowledge. We are going to discuss a motivation and an heuristics for the proof search in the next section.

Within the scope of this book we accept the "external" explanation for the heuristics we use as a sufficient solution.

As we can see from the above examples structural rules and especially the ($contr \rightarrow$) rule complicates the proof searching task.

The Gentzen type proof systems **RS** and **GL** from chapter don't contain the structural rules and are complete with respect to classical semantics, as is the original Gentzen system **LK**, which does contain the structural rules. As (via Completeness Theorem) all three classical proof system **RS, GL, LK** are equivalent we can say that the structural rules can be eliminated from the system **LK**.

A natural question of elimination of structural rules from the intutionistic Gentzen system **LI** arizes.

The following example illustrates the negative answer.

Example 3

We know, by the theorem about the connection between classical and intuitionistic logic (theorem 7.6) and corresponding Completeness Theorems that for any formula $A \in \mathcal{F}$,

$$\models A \quad \text{if and only if} \quad \vdash_I \neg\neg A,$$

where $\models A$ means that A is a classical tautology, $\vdash_I$ means that A is intutionistically provable, i.e. is provable in any intuitionistically complete proof system. The system **LI** is intuitionistically complete, so we have that for any formula A,

$$\models A \quad \text{if and only if} \quad \vdash_{\mathbf{LI}} \neg\neg A.$$

We have just proved that $\nvdash_{\mathbf{LI}} (\neg\neg A \Rightarrow A)$. Obviously $\models (\neg\neg A \Rightarrow A)$, so we know that $\neg\neg(\neg\neg A \Rightarrow A)$ must have a proof in **LI**.

We are going to prove that

$$\vdash_{\mathbf{LI}} \neg\neg(\neg\neg A \Rightarrow A)$$

and that the structural rule $(contr \longrightarrow)$ is essential to the existence of its proof, i.e. that without it the formula $\neg\neg(\neg\neg A \Rightarrow A)$ is not provable in **LI**.

The following decomposition tree **T** is a proof of $\neg\neg(\neg\neg A \Rightarrow A)$ in **LI**.

T

$$\longrightarrow \neg\neg(\neg\neg A \Rightarrow A)$$

$$first\ of\ 2\ choices:\ (\rightarrow \neg), (\rightarrow weak)$$

$$\mid (\longrightarrow \neg)$$

$$\neg(\neg\neg A \Rightarrow A) \longrightarrow$$

$$first\ of\ 2\ choices: (contr \longrightarrow), (\neg \longrightarrow)$$

$$\mid (contr \longrightarrow)$$

$$\neg(\neg\neg A \Rightarrow A), \neg(\neg\neg A \Rightarrow A) \longrightarrow$$

$$one\ of\ 2\ choices$$

$$\mid (\neg \longrightarrow)$$

$$\neg(\neg\neg A \Rightarrow A) \longrightarrow (\neg\neg A \Rightarrow A)$$

$$one\ of\ 3\ choices$$

$$\mid (\longrightarrow \Rightarrow)$$

$$\neg(\neg\neg A \Rightarrow A), \neg\neg A \longrightarrow A$$

$$one\ of\ 2\ choices$$

$$\mid (\longrightarrow weak)$$

$$\neg(\neg\neg A \Rightarrow A), \neg\neg A \longrightarrow$$

$$one\ of\ 3\ choices$$

$$\mid (exch \longrightarrow)$$

$$\neg\neg A, \neg(\neg\neg A \Rightarrow A) \longrightarrow$$

$$one\ of\ 3\ choices$$

$$\mid (\neg \longrightarrow)$$

$$\neg(\neg\neg A \Rightarrow A) \longrightarrow \neg A$$

$$one\ of\ 3\ choices$$

$$\mid (\longrightarrow \neg)$$

$$A, \neg(\neg\neg A \Rightarrow A) \longrightarrow$$

$$one\ of\ 2\ choices$$

$$\mid (exch \longrightarrow)$$

$$\neg(\neg\neg A \Rightarrow A), A \longrightarrow$$

$$one\ of\ 3\ choices$$

$$\mid (\neg \longrightarrow)$$

$$A \longrightarrow (\neg\neg A \Rightarrow A)$$

$$one\ of\ 3\ choices$$

$$\mid (\longrightarrow \Rightarrow)$$

$$\neg\neg A, A \longrightarrow A$$

$$axiom$$

Assume now that the rule ($contr \longrightarrow$) is not available. All possible decomposition trees are as follows.

T1

$$\longrightarrow \neg\neg(\neg\neg A \Rightarrow A)$$

$$\mid (\longrightarrow \neg)$$

$$one\ of\ 2\ choices$$

$$\neg(\neg\neg A \Rightarrow A) \longrightarrow$$

$$\mid (\neg \longrightarrow)$$

$$only\ one\ choice$$

$$\longrightarrow (\neg\neg A \Rightarrow A)$$

$$\mid (\longrightarrow \Rightarrow)$$

$$one\ of\ 2\ choices$$

$$\neg\neg A \longrightarrow A$$

$$\mid (\longrightarrow weak)$$

$$only\ one\ choice$$

$$\neg\neg A \longrightarrow$$

$$\mid (\neg \longrightarrow)$$

$$only\ one\ choice$$

$$\longrightarrow \neg A$$

$$\mid (\longrightarrow \neg)$$

$$one\ of\ 2\ choices$$

$$A \longrightarrow$$

$$non-axiom$$

T2

$$\longrightarrow \neg\neg(\neg\neg A \Rightarrow A)$$

$$\mid (\longrightarrow weak)$$

$$second\ of\ 2\ choices$$

$$\longrightarrow$$

$$non-axiom$$

T3

$$\longrightarrow \neg\neg(\neg\neg A \Rightarrow A)$$

$$\mid (\longrightarrow \neg)$$

$$\neg(\neg\neg A \Rightarrow A) \longrightarrow$$

$$\mid (\neg \longrightarrow)$$

$$\longrightarrow (\neg\neg A \Rightarrow A)$$

$$\mid (\longrightarrow weak)$$

$$second\ of\ 2\ choices$$

$$\longrightarrow$$

$$non-axiom$$

T4

$$\longrightarrow \neg\neg(\neg\neg A \Rightarrow A)$$

$$\mid (\longrightarrow \neg)$$

$$\neg(\neg\neg A \Rightarrow A) \longrightarrow$$

$$\mid (\neg \longrightarrow)$$

$$\longrightarrow (\neg\neg A \Rightarrow A)$$

$$\mid (\longrightarrow\Rightarrow)$$

$$]$$

$$\neg\neg A \longrightarrow A$$

$$\mid (\longrightarrow weak)$$

$$only\ one\ choice$$

$$\neg\neg A \longrightarrow$$

$$\mid (\neg \longrightarrow)$$

$$only\ one\ choice$$

$$\longrightarrow \neg A$$

$$\mid (\longrightarrow weak)$$

$$second\ of\ 2\ choices$$

$$\longrightarrow$$

$$non-axiom$$

This proves that the formula $\neg\neg(\neg\neg A \Rightarrow A)$ is not provable in **LI** without ($contr \longrightarrow$) rule and hence this rule can't be eliminated.

7.2.3 Proof Search Heuristic Method

Before we define a heuristic method of searching for proof in **LI** let's make some additional observations to the observations 1-5 from section 7.2.1.

Observation 6: Our goal while constructing the decomposition tree is to obtain axiom or indecomposable leaves. With respect to this goal the use logical decomposition rules has a priority over the use of the structural rules and we use this information while describing the proof search heuristic.

Observation 7: all logical decomposition rules ($\circ \rightarrow$), where $\circ$ denotes any connective, must have a formula we want to decompose as the first formula at the decomposition node, i.e. if we want to decompose a formula $\circ A$, the node must have a form $\circ A, \Gamma \longrightarrow \Delta$. Sometimes it is necessary to decompose a formula within the sequence Γ first in order to find a proof.

For example, consider two nodes

$$n_1 = \neg\neg A, (A \cap B) \longrightarrow B$$

and

$$n_2 = (A \cap B), \neg\neg A \longrightarrow B.$$

We are going to see that the results of decomposing n_1 and n_2 differ dramatically.

Let's decompose the node n_1. Observe that the only way to be able to decompose the formula $\neg\neg A$ is to use the rule ($\rightarrow weak$) first. The two possible decomposition trees that starts at the node n_1 are as follows.

$$\mathbf{T1}_{n_1}$$

$$\neg\neg A, (A \cap B) \longrightarrow B$$

$$\mid (\rightarrow weak)$$

$$\neg\neg A, (A \cap B) \longrightarrow$$

$$\mid (\neg \rightarrow)$$

$$(A \cap B) \longrightarrow \neg A$$

$$\mid (\cap \rightarrow)$$

$$A, B \longrightarrow \neg A$$

$$\mid (\rightarrow \neg)$$

$$A, A, B \longrightarrow$$

$$non - axiom$$

T2$_{n_1}$

$$\neg\neg A, (A \cap B) \longrightarrow B$$

$$\mid (\rightarrow weak)$$

$$\neg\neg A, (A \cap B) \longrightarrow$$

$$\mid (\neg \rightarrow)$$

$$(A \cap B) \longrightarrow \neg A$$

$$\mid (\rightarrow \neg)$$

$$A, (A \cap B) \longrightarrow$$

$$\mid (\cap \rightarrow)$$

$$A, A, B \longrightarrow$$

$$non - axiom$$

Let's now decompose the node n_2. Observe that following our **Observation 6** we start by decomposing the formula $(A \cap B)$ by the use of the rule $(\cap \rightarrow)$ first. A decomposition tree that starts at the node n_2 is as follows.

T1$_{n_2}$

$$(A \cap B), \neg\neg A \longrightarrow B$$

$$\mid (\cap \rightarrow)$$

$$A, B, \neg\neg A \longrightarrow B$$

$$axiom$$

This proves that the node n_2 is provable in **LI**, i.e.

$$\vdash_{\mathbf{LI}} \quad (A \cap B), \neg\neg A \longrightarrow B.$$

Of course, we have also that the node n_1 is also provable in **LI**, as one can obtain the node n_2 from it by the use of the rule ($exch \rightarrow$).

Observation 8: the use of structural rules are important and necessary while we search for proofs. Nevertheless we have to use them on the "must" basis and set up some guidelines and priorities for their use.

For example, use of weakening rule discharges the weakening formula, and hence an information that may be essential to the proof. We should use it only when it is absolutely necessary for the next decomposition steps. Hence, the use of weakening rule ($\rightarrow weak$) can, and should be restricted to the cases when it leads to possibility of the use of the negation rule ($\neg \rightarrow$).

This was the case of the decomposition tree $\mathbf{T1}_{n_1}$. We used it as an necessary step, but still it discharged too much information and we didn't get a proof, when proof of the node existed.

In this case the first rule in our search should have been the exchange rule, followed by the conjunction rule (no information discharge) not the weakening (discharge of information) followed by negation rule. The full proof of the node n_1 is the following.

$$\mathbf{T3}_{n_1}$$

$$\neg\neg A, (A \cap B) \longrightarrow B$$

$$\mid (exch \longrightarrow)$$

$$(A \cap B), \neg\neg A \longrightarrow B$$

$$\mid (\cap \rightarrow)$$

$$A, B, \neg\neg A \longrightarrow B$$

$$axiom$$

As a result of the **observations 1- 5** from section 7.2.1 and **observations 6 - 8** above we adopt the following.

Heuristic Procedure for Proof Search in LI.

For any $A \in \mathcal{F}$ we construct the set of decomposition trees $\mathbf{T}_{\rightarrow A}$ following the rules below.

Rules for Decomposition Tree Generation

1. Use first logical rules where applicable.

2. Use $(exch \rightarrow)$ rule to decompose, via logical rules, as many formulas on the left side of $\longrightarrow$ as possible.

3. Use $(\rightarrow weak)$ only on a "must" basis in connection with $(\neg \rightarrow)$ rule.

4. Use $(contr \rightarrow)$ rule as the last recourse and only to formulas that contain $\neg$ or $\Rightarrow$ as connectives.

5. Let's call a formula A to which we apply $(contr \rightarrow)$ rule **a contraction** formula.

6. The only contraction formulas are formulas containing $\neg$ between theirs logical connectives.

7. Within the process of construction of all possible trees use $(contr \rightarrow)$ rule only to contraction formulas.

8. Let C be a contraction formula appearing on the node n of the decomposition tree of $\mathbf{T}_{\rightarrow A}$. For any contraction formula C, any node n, we apply $(contr \rightarrow)$ rule the the formula C at most as many times as the number of sub-formulas of C.

If we find a tree with all axiom leaves we have a proof, i.e. $\vdash_{LI} A$ and if all (finite number) trees have a non-axiom leaf we have proved that proof of A does not exist, i.e. $\nvdash_{LI} A$.

7.3 Introduction to Modal S4 and S5 Logics

The non-classical logics can be divided in two groups: those that rival classical logic and those which extend it. The Lukasiewicz, Kleene, and Intuitionistic Logics are in the first group. The modal logics are in the second.

The rival logics do not differ from classical logic in terms of the language employed. Rather, rival logics differ in that certain theorems or tautologies of classical logic are rendered false, or not provable in them.

Perhaps the most notorious example of this is the law of excluded middle $(A \cup \neg A)$. This is provable in, and is a tautology of classical logic but is not provable in, and is not tautology of intuitionistic logic, or is not a tautology under any of the extensional logics semantics we have discussed.

Logics which extend classical logic sanction all the theorems of classical logic but, generally, supplement it in two ways. Firstly, the languages of these non-classical logics are extensions of those of classical logic, and secondly, the theorems of these non-classical logics supplement those of classical logic. Usually, such supplementation is provided by the enriched language. For example, modal logics are enriched by the addition of two new connectives that represent the meaning of *it is necessary that* and *it is possible that.* We use the notation **I** for *it is necessary that* and **C** for *it is possible that.* Other notations used are: ∇, N, L for *it is necessary that*, and $\Diamond$ P, M for *it is possible that.* The symbols N, L, P, M or alike, are often used in computer science investigations. The symbols ∇ and $\diamond$ were first to be used in modal logic literature, the symbols **I, C** come from algebraic and topological interpretation of modal logics. **I** corresponds to the *interior* of the set and **C** to its *closure.*

The idea of a modal logic was first formulated by an American philosopher, C.I. Lewis in 1918. He has proposed yet another interpretation of lasting consequences, of the logical implication. In an attempt to avoid, what some felt, the paradoxes of semantics for classical implication which accepts as true that a false sentence implies any sentence he created a notion of *a modal truth*, which lead to the notion of *modal logic.* The idea was to distinguish two sorts of truth: *necessary* truth and mere *possible (contingent)* truth. A *possibly* true sentence is one which, though true, could be false. A necessary truth is hence the one which could not be otherwise; a contingent (possible) truth is one which could. The distinction between them is a metaphysical one and should not be confused with the distinction between *a priori* and *a posteriori* truths. An *a priori* truth is one which can be *known* independently of experience, and an *a posteriori* truth is one which cannot. Such notions appeal to epistemic considerations and the whole area of modal logics bristles with philosophical difficulties and hence the numbers of logics have been created. Unlike the classical connectives, the modal connectives do not admit of truth-functional interpretation. This was the reason for which modal logics was first developed as a proof systems, with intuitive notion of semantics expressed by the set of adopted axioms.

The first semantics, and hence the proofs of the completeness theorems came some 20 years later. It took yet another 25 years for discovery and development of the second more general approach to the semantics. These are two established ways of interpret modal connectives, i.e. to define modal semantics.

The historically first one is due to Mc Kinsey and Tarski (1944, 1946). It is a topological interpretation that provides a powerful mathematical interpretation of some of them, namely S4 and S5. It connects the modal notion of *necessity* with the topological notion of *interior* of a set, and the notion of *possibility* with the notion of its *closure* . Our choice of symbols **I** and **C** for modal connectives comes from this interpretation. The topological interpretation powerful as it is, is less universal in providing models for other modal logics. The most recent one is due to Kripke (1964). It uses the notion *possible world.* Roughly, we say that $\mathbf{C}A$ is true if A is true in some possible world, called actual world, and $\mathbf{I}A$ is true if A is true in every possible world.

We present the formal definition later in this chapter, but this intuitive meaning can be useful in unconvincing ourselves about validity (or sense) of adopted axioms and rules of inference.

As we have already mentioned, modal logics were first developed, as was the intuitionistic logic, in a form of proof systems only. First Hilbert style modal proof system was published by Lewis and Langford in 1932. They presented a formalization for two modal logics, which they called S1 and S2. They also outlined three other proof systems, called S3, S4, and S5.

In 1933 Gödel worked with Heyting's "sentential logic" proof system, what we are calling now Intuitionistic logic. He considered a particular modal proof system and asserted that theorems of Heyting's "sentential logic" could be obtained from it by using a certain translation. His presentation of the discovered proof system, now known as S4 logic, was particularly elegant.

Since then hundreds of modal logics have been created. There are some standard texts in the subject. These are, between the others: Hughes and Cresswell (1969) for philosophical motivation for various modal logics and Intuitionistic logic, Bowen (1979) for a detailed and uniform study of Kripke models for modal logics, Segeberg (1971) for excellent classification, and Fitting (1983), for extended and uniform studies of automated proof methods for classes of modal logics.

Hilbert Style Modal Proof Systems

We present here Hilbert style formalization for S4 and S5 logics due to Mc Kinsey and Tarski (1948) and Rasiowa and Sikorski (1964). We also discuss the relationship between S4 and S5, and between the Intuitionistic logic and S4 modal logic, as first observed by Gödel.

They stress the connection between S4, S5 and topological spaces which constitute models for them. Hence the use of symbols **I**, **C** for necessity and possibility, respectively. The connective **I** corresponds to the symbol denoting a topological interior of a set and **C** to the closure of a set.

Modal Language

We add to the propositional language $\mathcal{L}_{\{\cup,\cap,\Rightarrow,\neg\}}$ two extra one argument connectives **I** and **C**. I.e. we adopt

$$\mathcal{L} = \mathcal{L}_{\{\cup,\cap,\Rightarrow,\neg,\mathbf{I},\mathbf{C}\}} \tag{7.39}$$

as our modal language. We read a formula $\mathbf{I}A$, $\mathbf{C}A$ as *necessary* A and *possible* A, respectively.

The language is common to all modal logics. Modal logics differ on a choice of axioms and rules of inference, when studied as proof systems and on a choice of semantics.

McKinsey, Tarski (1948)

As modal logics extend the classical logic, any modal logic contains two groups of axioms: classical and modal.

Axioms Group 1: classical axioms

Any modal logic adopts as its classical axioms any complete set of axioms for a classical propositional logic.

Axioms Group 2: modal axioms

M1 $(\mathbf{I}A \Rightarrow A)$,

M2 $(\mathbf{I}(A \Rightarrow B) \Rightarrow (\mathbf{I}A \Rightarrow \mathbf{I}B))$,

M3 $(\mathbf{I}A \Rightarrow \mathbf{II}A)$,

M4 $(\mathbf{C}A \Rightarrow \mathbf{IC}A)$.

Rules of inference

We adopt the Modus Ponens (MP)

$$(MP)\ \frac{A\ ;\ (A \Rightarrow B)}{B}$$

and an additional modal rule (I) introduced by Gödel

$$(I)\ \frac{A}{\mathbf{I}A}$$

referred to as *necessitation.*

We define modal proof systems S4 and S5 as follows.

$$S4 = (\ \mathcal{L},\ \mathcal{F},\ \text{classical axioms},\ M1-M3,\ (MP),\ (I)\ ), \tag{7.40}$$

$$S5 = (\ \mathcal{L},\ \mathcal{F},\ \text{classical axioms},\ M1-M4,\ (MP),\ (I)\ ). \tag{7.41}$$

Observe that the axioms of S5 extend the axioms of S4 and both system share the same inference rules, hence we have immediately the following.

Fact 7.3

For any formula $A \in \mathcal{F}$, if $\vdash_{S4} A$, then $\vdash_{S5} A$.

Rasiowa, Sikorski (1964)

It is often the case, and it is in our systems, that modal connectives are expressible by each other, i.e. that we can adopt one of them and define the other as follows.

$$\mathbf{I}A = \neg\mathbf{C}\neg A, \tag{7.42}$$

and

$$\mathbf{C}A = \neg\mathbf{I}\neg A. \tag{7.43}$$

The equality sign in equations (7.42), and (7.43) means that we replace the formula in left side of the equation by the formula in the right side anywhere where the left side (or right side) formula is appears as a sub formula of a formula of $\mathcal{L}$. In modal logics S4 and S5 the connective $\mathbf{C}$ is expressible by $\neg$ and $\mathbf{I}$, as stated above by (7.43), we hence assume now that the language $\mathcal{L}$ contains only one modal connective $\mathbf{I}$.

Language

$$\mathcal{L} = \mathcal{L}_{\{\cap, \cup, \Rightarrow, \neg, \mathbf{I}\}}. \tag{7.44}$$

There are, as before, two groups of axioms: classical and modal.

Axioms Group 1: classical axioms

We adopt as classical axioms any complete set of axioms for a classical propositional logic.

Axioms Group2: modal axioms

R1 $((\mathbf{I}A \cap \mathbf{I}B) \Rightarrow \mathbf{I}(A \cap B))$,

R2 $(\mathbf{I}A \Rightarrow A)$,

R3 $(\mathbf{I}A \Rightarrow \mathbf{II}A)$,

R4 $\mathbf{I}(A \cup \neg A)$,

R5 $(\neg\mathbf{I}\neg A \Rightarrow \mathbf{I}\neg\mathbf{I}\neg A)$

Rules of inference

We adopt the Modus Ponens (MP)

$$(MP)\ \frac{A\ ;\ (A \Rightarrow B)}{B}$$

and an additional modal rule (RI)

$$(RI)\ \frac{(A \Rightarrow B)}{(\mathbf{I}A \Rightarrow \mathbf{I}B)}.$$

We define modal logic proof systems $RS4$, $RS5$ as follows.

$$RS4 = (\ \mathcal{L},\ \mathcal{F},\ \text{classical axioms},\ R1 - R4,\ (MP),\ (RI)\ ). \tag{7.45}$$

$$RS5 = (\ \mathcal{L},\ \mathcal{F},\ \text{classical axioms},\ R1 - R5,\ (MP),\ (RI)\ ). \tag{7.46}$$

Observe that the axioms of RS5 extend, as the axioms of TS5, the axioms of TS4 and both system share the same inference rules, hence we have immediately the following.

Fact 7.4

For any formula $A \in \mathcal{F}$, if $\vdash_{RS4} A$, then $\vdash_{RS5} A$.

7.3.1 Algebraic Semantics for S4 and S5

The McKinsey, Tarski proof systems (7.40), (7.41), and Rasiowa, Sikorski proof systems (7.45), (7.45) for modal logics S4, S5 are complete with the respect to both algebraic topological semantics, and Kripke semantics.

We shortly discuss the topological semantics, and algebraic completeness theorems and leave the Kripke semantics for the reader to explore from other, multiple sources.

The topological semantics was initiated by McKinsey, Tarski's (1946, 1948) and consequently developed and examined by many authors into a field of *Algebraic Logic.*They are presented in detail in now classic algebraic logic books: *"Mathematics of Metamathematics"*, Rasiowa, Sikorski (1964) and *"An Algebraic Approach to Non-Classical Logics"*, Rasiowa (1974).

We want to point out that the first idea of a connection between modal propositional calculus and topology is due to Tang Tsao -Chen, (1938) and Dugunji (1940).

Here are some basic definitions.

Boolean Algebra
An abstract algebra

$$\mathcal{B} = (B,\ 1,\ 0,\ \Rightarrow,\ \cap,\ \cup, \neg) \tag{7.47}$$

is said to be a *Boolean algebra* if it is a distributive lattice and every element $a \in B$ has a complement $\neg a \in B$.

Topological Boolean algebra
By a *topological Boolean algebra* we mean an abstract algebra

$$\mathcal{B} = (B,\ 1,\ 0,\ \Rightarrow,\ \ \cap,\ \cup, \neg,\ I), \tag{7.48}$$

where $(B,\ 1,\ 0,\ \Rightarrow,\ \cap,\ \cup, \neg)$ is a Boolean algebra (7.47) and, moreover, the following conditions hold for any $a, b \in B$.

$$I(a \cap b) = Ia \cap Ib, \quad Ia \cap a = Ia, \quad IIa = Ia, \quad \text{and} \quad I1 = 1. \tag{7.49}$$

The element Ia is called a **interior** of a. The element $\neg I \neg a$ is called a **closure of a** and will be denoted by Ca. Thus the operations I and C are such that

$$Ca = \neg I \neg a \quad \text{and} \quad Ia = \neg C \neg a.$$

In this case we write the topological Boolean algebra (7.48) as

$$\mathcal{B} = (B,\ 1,\ 0,\ \Rightarrow,\ \ \cap,\ \cup, \neg,\ I,\ C). \tag{7.50}$$

It is easy to prove that in in any topological Boolean algebra (7.50) the following conditions hold for any $a, b \in B$.

$$C(a \cup b) = Ca \cup Cb, \quad Ca \cup a = Ca, \quad CCa = Ca, \quad \text{and} \quad C0 = 0. \tag{7.51}$$

If X is a topological space with an interior operation I, then the family $\mathcal{P}(X)$ of all subsets of X is a topological Boolean algebra with $1 = X$, the operation $\Rightarrow$ defined by the formula

$$Y \Rightarrow Z = (X - Y) \cup Z, \quad \text{for all subsets } Y, Z \text{of } X,$$

set-theoretical operations of union, intersection, complementation, and the interior operation I. Obviously, every sub algebra of this algebra is a topological Boolean algebra, called a *topological field of sets* or, more precisely, a *topological field* of subsets of X.

Given a topological Boolean algebra (7.47) $(B,\ 1,\ 0,\ \Rightarrow,\ \cap,\ \cup, \neg)$. The element $a \in B$ is said to be **open** (**closed**) if $a = Ia$ $(a = Ca)$.

Clopen Topological Boolean algebra
A topological Boolean algebra (7.50)

$$\mathcal{B} = (B,\ 1,\ 0,\ \Rightarrow,\ \cap,\ \cup, \neg,\ I,\ C\ ).$$

such that every **open** element is **closed** and every **closed** element is **open**, i.e. such that for any $a \in B$

$$CIa = Ia \quad \text{and} \quad ICa = Ca \tag{7.52}$$

is called a *clopen topological Boolean algebra.*

We loosely say that a formula A of a modal language is a modal $S4$ **tautology** if and only if any topological Boolean algebra (7.50) is a *model* for A.
We say that A is a modal $S5$ **tautology** if and only if any **clopen** topological Boolean algebra (7.52) is a *model* for A.

We put it formally as follows.

Definition 7.4 (Modal Algebraic Model)

For any formula A of a modal language $\mathcal{L}_{\{\cup,\cap,\Rightarrow,\neg,\mathbf{I},\mathbf{C}\}}$ and for any topological Boolean algebra $\mathcal{B} = (B,\ 1,\ 0,\ \Rightarrow,\ \cap,\ \cup, \neg,\ I,\ C)$,

the algebra $\mathcal{B}$ is a **model** *for the formula A and denote it by*

$$\mathcal{B} \models A$$

if and only if $v^*(A) = 1$ holds for all variables assignments $v : VAR \longrightarrow B$.

Definition 7.5 ($S4, S5$ Tautology)

The formula A is a modal $S4$ **tautology** *($S5$* **tautology***) and is denoted by*

$$\models_{S4} A \quad (\models_{S5} A)$$

if and only if for all **topological** *Boolean algebras (***clopen topological** *Boolean algebras) $\mathcal{B}$ we have that*

$$\mathcal{B} \models A \quad (\mathcal{B} \models A).$$

In Algebraic Logic the notion of tautology is often defined using a notion "a formula A is **valid** in an algebra $\mathcal{B}$ ".It is formally defined in our case as as follows.

Definition 7.6

A formula A **is valid** *in a topological Boolean algebra $\mathcal{B} = (B,\ 1,\ 0,\ \Rightarrow,\ \cap,\ \cup, \neg,\ I,\ C)$, if and only if $v^*(A) = 1$ holds for all variables assignments $v : VAR \longrightarrow B$.*

Directly form definitions 7.5, 7.6 we get the following.

Fact 7.5 *For any formula A, $\models_{S4} A$ ($\models_{S5} A$) if and only if A* **is valid** *in all topological Boolean algebras (A* **is valid** *in all clopen topological Boolean algebras).*

We write $\vdash_{S4} A$ and $\vdash_{S5} A$ do denote any proof system for modal S4, S5 logics and in particular the proof systems (7.40), (7.45), and (7.41), (7.46), respectively.

Theorem 7.17 (Completeness Theorem) *For any formula A of the modal language $\mathcal{L}_{\{\cup,\cap,\Rightarrow,\neg,\mathbf{I},\mathbf{C}\}}$,*

$$\vdash_{S4} A \ \textit{ if and only if } \ \models_{S4} A,$$

$$\vdash_{S5} A \ \textit{ if and only if } \ \models_{S5} A.$$

The completeness for $S4$ follows directly from the Theorem 7.18. The completeness for $S5$ follows from the $S4$ completeness and Embedding Theorems 7.22, 7.23. It also can be easily proved independently by adopting the Algebraic Completeness Theorem proof for $S4$ to clopen topological algebras.

Theorem 7.18 (Algebraic Completeness Theorem)

For any formula A of the modal language $\mathcal{L}= \mathcal{L}_{\{\cup,\cap,\Rightarrow,\neg,\mathbf{I},\mathbf{C}\}}$ the following conditions are equivalent.

(i) $\vdash_{S4} A$,

(ii) $\models_{S4} A$,

(iii) *A is valid in every topological field of sets $\mathcal{B}(X)$,*

(iv) *A is valid in every topological Boolean algebra $\mathcal{B}$ with at most 2^{2^r} elements, where r is the number of all sub formulas of A,*

(iv) *$v^*(A) = X$ for every variable assignment v in the topological field of sets $\mathcal{B}(X)$ of all subsets of a dense-in -itself metric space $X \neq \emptyset$ (in particular of an n-dimensional Euclidean space X).*

On S4 derivable disjunction

In a classical logic it is possible for the disjunction $(A \cup B)$ to be a tautology when neither A nor B is a tautology. The tautology $(A \cup \neg A)$ is the simplest example. This does not hold for the intuitionistic logic. We have a similar theorem for modal S4 logic, as proved by McKinsey and Tarski.

Theorem 7.19

A disjunction $(IA \cup IB)$ is S4 provable if and only if either A or B is S4 provable, i.e.

$$\vdash_{S4} (IA \cup IB) \quad \text{if and only if} \quad \vdash_{S4} A \quad \text{or} \quad \vdash_{S4} B.$$

The proof follows directly from the Completeness Theorem 7.18 and the following semantical, proof system independent version of the theorem 7.19.

Theorem 7.20 (McKinsey, Tarski, 1948)

For any $A \in \mathcal{F}$,

$$\models_{S4}(IA \cup IB) \quad \text{if and only if} \quad \models_{S4} A \ \text{or} \ \models_{S4} B.$$

The completeness theorem allows us to formulate theorems about logics in terms independent of the proof system considered. In this sense the notion of tautology is more general then the notion of provability. This is why often we use rather the tautology formulation of the known facts about the logic and their relationships instead of the notion of provability.

Following the Completeness Theorem 7.18 we get a semantical version of the theorem 7.3.

Theorem 7.21

For any formula $A \in \mathcal{F}$,

$$\text{if} \quad \models_{S4} A, \quad \text{then} \quad \models_{S5} A.$$

Consider a modal language $\mathcal{L}$ with both modal connectives, i.e.

$$\mathcal{L}= \mathcal{L}_{\{\cup,\cap,\Rightarrow,\neg,\mathbf{I},\mathbf{C}\}}.$$

The above theorem7.21 says that the S4 tautologies form a subset of S5 tautologies. We have even a more powerful relationship, namely the following.

Theorem 7.22 (Embedding 1)

For any formula $A \in \mathcal{F}$,

$$\models_{S4} A \quad \textit{if and only if} \quad \models_{S5} \mathbf{IC}A,$$

$$\vdash_{S4} A \quad \textit{if and only if} \quad \vdash_{S5} \mathbf{IC}A.$$

Theorem 7.23 (Embedding 2)

For any formula $A \in \mathcal{F}$,

$$\models_{S5} A \quad \textit{if and only if} \quad \models_{S4} \mathbf{ICI}A.$$

$$\vdash_{S5} A \quad \textit{if and only if} \quad \vdash_{S4} \mathbf{ICI}A.$$

Theorem 7.24 (Embedding 3)

For any formula $A \in \mathcal{F}$,

$$\textit{if} \ \models_{S5} A, \quad \textit{then} \quad \models_{S4} \neg\mathbf{I}\neg A.$$

The fist proof of the above embedding theorems was given by Matsumoto in 1955. Provability. Fitting semantical 1983 Ohnishi and Matsumoto 1957/59 Gentzen Methods in Modal Calculi Osaka Mathematical Journal 9.113 -130

7.3.2 S4 and Intuitionistic Logic, S5 and Classical Logic

As we have said in the introduction, Gödel was the first to consider the connection between the intuitionistic logic and a logic which was named later S4. His proof was purely syntactic in its nature, as semantics for neither intuitionistic logic nor modal logic S4 had not been invented yet.

The algebraic proof of this fact, was first published by McKinsey and Tarski in 1948. We now define the mapping establishing the connection (definition 7.7) and refer the reader to Rasiowa and Sikorski book "Mathematics of Metamathematics" for its proof.

Let $\mathcal{L}$ be a propositional language of modal logic, as defined by (7.44), i.e the language

$$\mathcal{L} = \mathcal{L}_{\{\cap,\cup,\Rightarrow,\neg,\mathbf{I}\}}.$$

Let $\mathcal{L}_0$ be a language obtained from $\mathcal{L}$ by elimination of the connective $\mathbf{I}$ and by the replacement the negation connective $\neg$ by the intuitionistic negation, which we will denote here by a symbol $\sim$. Such obtained language

$$\mathcal{L}_0 = \mathcal{L}_{\{\cap,\cup,\Rightarrow,\sim\}} \tag{7.53}$$

is a propositional language of the intuitionistic logic.

In order to establish the connection between the languages (7.44) and (7.53), and hence between modal and intuitionistic logic, we define a mapping f which to every formula $A \in \mathcal{F}_0$ of $\mathcal{L}_0$ assigns a formula $f(A) \in \mathcal{F}$ of $\mathcal{L}$.

We define a mapping f as follows.

Definition 7.7 (Gódel - Tarski)

A function $f : \mathcal{F}_0 \quad \rightarrow \mathcal{F}$ be such that

$$f(a) = \mathbf{I}a \quad for \quad any \quad a \in VAR,$$

$$f((A \Rightarrow B)) = \mathbf{I}(f(A) \Rightarrow f(B)),$$
$$f((A \cup B)) = (f(A) \cup f(B)),$$
$$f((A \cap B)) = (f(A) \cap f(B)),$$
$$f(\sim A) = \mathbf{I}\neg f(A),$$

where A, B denote any formulas in $\mathcal{L}_0$ is called a **Gödel - Tarski mapping**.

Example

Let A be a formula
$$((\sim A \cap \sim B) \Rightarrow \sim (A \cup B))$$
and f be the mapping of definition 7.7. We evaluate $f(A)$ as follows

$$f((\sim A \cap \sim B) \Rightarrow \sim (A \cup B)) =$$

$$\mathbf{I}(f(\sim A \cap \sim B) \Rightarrow f(\sim (A \cup B)) =$$
$$\mathbf{I}((f(\sim A) \cap f(\sim B)) \Rightarrow f(\sim (A \cup B)) =$$
$$\mathbf{I}((\mathbf{I}\neg fA \cap \mathbf{I}\neg fB) \Rightarrow \mathbf{I}\neg f(A \cup B)) =$$
$$\mathbf{I}((\mathbf{I}\neg A \cap \mathbf{I}\neg B) \Rightarrow \mathbf{I}\neg(fA \cup fB)) =$$
$$\mathbf{I}((\mathbf{I}\neg A \cap \mathbf{I}\neg B) \Rightarrow \mathbf{I}\neg(A \cup B)).$$

We use notation $\vdash_I A$ do denote the fact that A is intuitionistically provable, i.e. provable in any intuitionistic proof system I.

With these hypotheses and notation the following theorem holds.

Theorem 7.25

Let f be the Gödel mapping (definition 7.7). For any formula A of $\mathcal{L}_0$,

$$\vdash_I A \quad \textit{if and only if} \quad \vdash_{S4} f(A),$$

where I, $S4$ denote any proof systems for intuitionistic and and S4 logic, respectively.

In order to establish the connection between the modal logic and classical logic we consider the Gódel - Tarski mapping (definition **??**) between the modal language $\mathcal{L}_{\{\cap,\cup,\Rightarrow,\neg,\mathbf{I}\}}$ and its classical sub-language $\mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow\}}$.

Now with every classical formula A we associate a modal formula $f(A)$ defined by induction on the length of A as follows:

$$f(a) = \mathbf{I}a, \quad f((A \Rightarrow B)) = \mathbf{I}(f(A) \Rightarrow f(B)),$$

$$f((A \cup B)) = (f(A) \cup f(B)),\ f((A \cap B)) = (f(A) \cap f(B)),\ f(\neg A) = \mathbf{I}\neg f(A).$$

We use notation $\vdash_H \ A$ do denote the fact that A is classically provable, i.e. provable in any proof system for classical propositional logic.

The following theorem established relationship between classical logic and modal $S5$.

Theorem 7.26

Let f be the Gödel mapping (definition 7.7) between $\mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow\}}$ and $\mathcal{L}_{\{\cap,\cup,\Rightarrow,\neg,\mathbf{I}\}}$. For any formula A of $\mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow\}}$,

$$\vdash_H \ A \quad \textit{if and only if} \quad \vdash_{S5} f(A),$$

where H, $S5$ denote any proof systems for classical and and S5 modal logic, respectively.

7.4 Homework Problems

1. The algebraic models for the intuitionistic logic are defined in terms of *Pseudo-Boolean Algebras* in the following way. A formula A is said to be an intuitionistic tautology if and only if $v \models A$, for all v and all Pseudo-Boolean Algebras, where v maps VAR into universe of a Pseudo-Boolean Algebra. I.e. A is an intuitionistic tautology if and only if it is true in all Pseudo-Boolean Algebras under all possible variable assignments.

   A 3 element Heyting algebra as defined in chapter 3 is a 3 element Pseudo-Boolean Algebra.

(i) Show that the 3 element Heyting algebra is a model for all formulas (7.12) - (7.27).

(ii) Determine for which of the formulas (7.28) - (7.34) the 3 element Heyting algebra acts as a counter-model.

2. Find proofs by constructing proper decomposition trees in Gentzen System **LI** of axioms **A1 - A11** of the proof system I defined by (7.4).

3. The completeness with respect to algebraic semantics of system **LI** follows from the Hauptzatz Theorem and the completeness theorem 7.2. The proof is a straightforward adaptation of the proof of the completeness theorem for **LK** included in chapter 6. Write carefully all steps of the proof of completeness theorem for **LI** .

4. Find proofs by constructing proper decomposition trees in Gentzen System **LI** of the intuitionistic tautologies (7.12) - (7.27).

5. Show that none of the formulas (7.28) - (7.34) is provable in **LI**.

6. Find proofs by constructing proper decomposition trees in Gentzen System **LI** of double negation of all of the formulas (7.28) - (7.34).

7. Give the proof of the Glivenko theorem 7.7, i.e. prove that any formula A is a classically provable if and only if $\neg\neg A$ is an intuitionistically provable.

8. Give few examples of formulas illustrating that the following theorems hold.

   Gödel (1) For any $A, B \in \mathcal{F}$, a formula $(A \Rightarrow \neg B)$ is a classically provable if and only if it is intuitionistically provable.

9. Give examples of formulas illustrating that the following theorems hold.

   Gödel (2) If a formula A contains no connectives except $\cap$ and $\neg$, then A is a classically provable if and only if it is an intuitionistically provable.

10. Use the Completeness Theorem 7.18 to show that the following proof system **C**S4 is a complete proof system for the modal logic S4.

    We adopt the modal language $\mathcal{L}_{\{\cup,\cap,\Rightarrow,\neg,\mathbf{I},\mathbf{C}\}}$. We adopt, as before, two groups of axioms: classical and modal.
    Group 1: we take any complete set of axioms for a classical propositional logic. Group 2: the following modal axioms.
    C1 $(\mathbf{C}(A \cup B) \Rightarrow (\mathbf{C}A \cup \mathbf{C}B))$,

    C2 $(A \Rightarrow \mathbf{C}A)$,

    C3 $(\mathbf{CC}A \Rightarrow \mathbf{C}A)$,

    C4 $\mathbf{C}(A \cap \neg A)$.

Rules of inference: we adopt the Modus Ponens (MP) and an additional rule,

$$(\mathbf{C})\ \frac{(A \Rightarrow B)}{(\mathbf{C}\neg B \Rightarrow \mathbf{C}\neg A)}.$$

We define the proof system $CS4$ as follows

$$CS4 = (\ \mathcal{L}, \mathcal{F},\ \text{classical axioms},\ C1 - C4,\ (MP),\ (\mathbf{C})\ ).$$

11. Evaluate $f(A)$, where f is the Gödel- Tarski mapping (definition 7.7), for all the formulas A listed below.

    (i) $(\neg A \cup \neg B) \Rightarrow (\neg A \cap \neg B))$

    (ii) $((A \Rightarrow B) \Rightarrow (\neg B \Rightarrow \neg A))$

    (iii) $((A \Rightarrow \neg B) \Rightarrow (B \Rightarrow \neg A))$

    (iv) $(\neg\neg\neg A \Rightarrow \neg A)$

    (v) $(\neg A \Rightarrow \neg\neg\neg A)$

    (vi) $(\neg\neg(A \Rightarrow B) \Rightarrow (A \Rightarrow \neg\neg B))$

    (vii) $((C \Rightarrow A) \Rightarrow ((C \Rightarrow (A \Rightarrow B)) \Rightarrow (C \Rightarrow B))$

12. Use the Completeness Theorem 7.18 and Embedding Theorems 7.22, 7.23 to show the following.

    (i) For any formula A, $\vdash_{RS4}\ A$, if and only if $vdash_{RS5}$ $\mathbf{I}\neg\mathbf{I}\neg A$, where RS4, RS5 are proof system (7.45) and (7.46).

    (ii) For any formula A, $\vdash_{S5}\ A$, if and only if $\models_{S4}$ $\mathbf{I}\neg\mathbf{I}\neg\mathbf{I}A$, where S4, S5 are proof system (7.40) and (7.41).

Chapter 8

Classical Predicate Semantics and Proof Systems

8.1 Formal Predicate Languages

Propositional languages are also called zero order languages, as opposed to predicate languages that are called first order languages. The same applies to the use of terms propositional and predicate logic; they are often called zero order and first order logics and we will use both terms equally.

We define a *predicate language* $\mathcal{L}$ following the pattern established by the propositional languages definitions. The predicate language $\mathcal{L}$ is more complicated in its structure and hence its *alphabet* $\mathcal{A}$ is much richer. The definition of its set of *formulas* $\mathcal{F}$ is more complicated. In order to define the set $\mathcal{F}$ we introduce an additional set $\mathbf{T}$, called a set of *terms* of the predicate language $\mathcal{L}$. We single out this set not only because we need it for the definition of formulas, but also because of its role in the development of other notions of predicate logic.

We will work with different predicate languages, depending on what applications we have in mind. All of these languages have some common features, and we begin with a following general definition.

Definition 8.1

By a **predicate language** $\mathcal{L}$ *we understand a triple*

$$\mathcal{L} = (\mathcal{A}, \mathbf{T}, \mathcal{F}), \tag{8.1}$$

where $\mathcal{A}$ is a predicate alphabet, $\mathbf{T}$*, is the set of terms, and $\mathcal{F}$ is a set of formulas.*

The **components** of $\mathcal{L}$ are as follows.

1. **Alphabet** $\mathcal{A}$ is the set

$$\mathcal{A} = VAR \cup CON \cup PAR \cup \mathbf{Q} \cup \mathbf{P} \cup \mathbf{F} \cup \mathbf{C}, \tag{8.2}$$

where VAR is set of predicate variables, CON is a set of propositional connectives, PAR a set of parenthesis, $\mathbf{Q}$ a set of quantifiers, $\mathbf{P}$, a set of predicate symbols, $\mathbf{F}$ a set of functions symbols, and $\mathbf{C}$ a set of constant symbols. We assume that all of the sets defining the alphabet are disjoint.

Predicate Variables VAR

We assume that we always have a countably infinite set VAR of predicate variables, called usually **variables**. We denote variables by $x, y, z, ...,$ with indices, if necessary, what we often express by writing

$$VAR = \{x_1, x_2, ...\}.$$

Propositional connectives CON

We define the set of propositional connectives CON in the same way as in the case of the propositional languages. It means that we assume that CON is *non-empty* and *finite set* and that consider only the connectives with one or two arguments, i.e.

$$CON = C_1 \cup C_2$$

where C_1 is a finite set (possibly empty) of unary connectives, C_2 is a finite set (possibly empty) of binary connectives of the language $\mathcal{L}$.

Parenthesis PAR

As in the propositional case, we adopt the signs (and) for our parenthesis., i.e. we define the set PAR as

$$PAR = \{(,)\}.$$

The set of propositional connectives CON defines a *propositional part* of the predicate logic language. What really differ one predicate language from the other is the choice of additional symbols to the symbols described above. These are called quantifiers symbols, predicate symbols, function symbols, and constant symbols. I.e. a particular predicate language is determined by specifying the following sets of symbols.

Quantifiers Q

We adopt two quantifiers; $\forall$ (for all, the universal quantifier) and $\exists$ (there exists, the existential quantifier), i.e. we have the following set of quantifiers

$$\mathbf{Q} = \{\forall, \exists\}.$$

In a case of the classical logic and the logics that extend it, it is possible to adopt only one quantifier and to define the other in terms of it and propositional connectives. It is impossible in a case of many non-classical logics, for example the intuitionistic logic. But even in the case of classical logic two quantifiers express better the common intuition, so we assume that we have two of them.

Predicate symbols P

Predicate symbols represent relations. We assume that we have an non empty, finite or countably infinite set **bf P** of predicate, or relation symbols. We denote predicate symbols by $P, Q, R, \ldots$, with indices, if necessary, what we often express by writing

$$\mathbf{P} = \{P_1, P_2, \ldots\}.$$

Each predicate symbol $P \in \mathbf{P}$ has a positive integer $\#P$ assigned to it; if $\#P = n$ then say P is called an n-ary (n - place) predicate (relation) symbol.

Function symbols F

We assume that we have a finite (may be empty) or countably infinite set **F** of function symbols. When the set **F** is empty we say that we deal with a *language without functional symbols.* We denote functional symbols by $f, g, h, \ldots$, with indices, if necessary, what we often express by writing

$$\mathbf{F} = \{f_1, f_2, \ldots\}.$$

Similarly, as in the case of predicate symbols, each function symbol $f \in \mathbf{F}$ has a positive integer $\#f$ assigned to it; if $\#f = n$ then say f is called an n-ary (n - place) function symbol.

Constant symbols C

We also assume that we have a finite (may be empty) or countably infinite set **C** of constant symbols. The elements of **C** are denoted by $c, d, e\ldots$, with indices, if necessary, what we often express by writing

$$\mathbf{C} = \{c_1, c_2, \ldots\}.$$

When the set **C** is empty we say that we deal with a *language without constant symbols.*

Sometimes the constant symbols are defined as 0-ary function symbols, i.e. $\mathbf{C} \subseteq \mathbf{F}$. We single them out as a separate set for our convenience.

Observe that what distinguishes now one predicate language $\mathcal{L}$ form the pother is the choice of the components CON, and **P, F, C** of its alphabet $\mathcal{A}$. We hence will write

$$\mathcal{L}_{CON}(\mathbf{P}, \mathbf{F}, \mathbf{C}) \tag{8.3}$$

to denote the predicate language $\mathcal{L}$ determined by **P, F, C** and the set of propositional connectives CON.

Once the set of propositional connectives is fixed, the predicate language is determined by the sets **P, F** and **C** and we l write

$$\mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C}) \tag{8.4}$$

for the predicate language $\mathcal{L}$ determined by **P, F** and **C** (with a fixed set of propositional connectives). If there is no danger of confusion, we may abbreviate $\mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$ to just $\mathcal{L}$.

We sometimes allow the same symbol to be used as an n-place relation symbol, and also as an m-place one; no confusion should arise because the different uses can be told apart easily. Similarly for function symbols.

Having defined the basic elements of syntax, the alphabet, we can now complete the formal definition of the predicate language by defining two more complex sets: the set T of all terms and the set $\mathcal{F}$ of all well formed formulas of the language $\mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$.

2. **Terms T**

The set **T** of terms of a predicate language $\mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$ is defined as follows.

Definition 8.2 (Terms)

*Given a predicate language $\mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$ with an alphabet $\mathcal{A}$. The set **T** of terms of $\mathcal{L}$ is the smallest set $T \subseteq \mathcal{A}^*$ meeting the conditions:*

(i) any variable is a term, i.e. $VAR \subseteq \mathbf{T}$;

(ii) any constant symbol is a term, i.e. $\mathbf{C} \subseteq \mathbf{T}$;

(iii) if f is an nplace function symbol, i.e. $f \in \mathbf{F}$ and $\#f = n$ and $t_1, t_2, ..., t_n \in \mathbf{T}$, then $f(t_1, t_2, ..., t_n) \in \mathbf{T}$.

Example 8.1

Let $f \in \mathbf{F}, \# f = 1$, i.e. f is a one place function symbol. Let x, y be predicate variables, c, d constants, i.e. $x, y \in VAR,\ c, d \in \mathbf{C}$. The following expressions are terms:

$$x,\ y,\ f(x),\ f(y),\ f(c),\ f(d),\ f(f(x),)\ f(f(y)),\ ff((c)),\ f(f(d)), ...etc.$$

Example 8.2

If $\mathbf{F} = \emptyset, \mathbf{C} = \emptyset$, then the set $\mathbf{T}$ of terms consists of variables only, i.e.

$$\mathbf{T} = VAR = \{x_1, x_2, ...\}.$$

From the above we get the following observation.

Remark 8.1

For any predicate language $\mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$, the set $\mathbf{T}$ of its terms is always non-empty.

Example 8.3

If $f \in \mathbf{F}, \# f = 1$, $g \in \mathbf{F}, \# g = 2$, $x, y \in VAR, c, d \in \mathbf{C}$, then some of the terms are the following:

$$f(g(x, y)),\ f(g(c, x)),\ g(f(f(c)), g(x, y)),\ g(c, g(x, f(c))).$$

From time to time, the logicians are and we may be informal about how we write terms. For instance, if we denote a two place function symbol g by $+$, we may write $x + y$ instead $+(x, y)$. Because in this case we can think of $x + y$ as an unofficial way of designating the "real" term $+(x, y)$, or even $g(x, y)$.

2. **Formulas $\mathcal{F}$**

Before we define the set of formulas, we need to define one more set; the set of atomic, or elementary formulas. They are the "smallest" formulas as were the propositional variables in the case of propositional languages.

Atomic formulas

An atomic formula of a predicate language $\mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$ is any element of the alphabet $\mathcal{A}^*$ of the form

$$R(t_1, t_2, ..., t_n),$$

where $R \in \mathbf{P}, \# R = n$, i.e. R is n-ary relational symbol and $t_1, t_2, ..., t_n$ are terms. The set of all atomic formulas is denoted by $A\mathcal{F}$ and is defines as

$$A\mathcal{F} = \{R(t_1, t_2, ..., t_n) \in \mathcal{A}^* : R \in \mathbf{P},\ t_1, t_2, ..., t_n \in \mathbf{T},\ \# R = n,\ n \geq 1\}. \quad (8.5)$$

Example 8.4

Consider a language

$$\mathcal{L}(\emptyset, \{P\}, \emptyset),$$

for $\#P = 1$, i.e. a language without neither functional, nor constant symbols, and with one, one-place predicate symbol P. The set of atomic formulas contains all formulas of the form $P(x)$, for x any variable, i.e.

$$A\mathcal{F} = \{P(x) : x \in VAR\}.$$

Example 8.5

Let now

$$\mathcal{L} = \mathcal{L}(\{f, g\}, \{R\}, \{c, d\}),$$

for $\#f = 1$, $\#g = 2$, $\#R = 2$, i.e. $\mathcal{L}$ has two functional symbols: one -place symbol f and two-place symbol g; one two-place predicate symbol R, and two constants: c,d. Some of the atomic formulas in this case are the following.

$$R(c, d),\ R(x, f(c)),\ R(f(g(x, y)), f(g(c, x))),\ R(y, g(c, g(x, f(c)))).$$

Given a predicate language

$$\mathcal{L} = \mathcal{L}_{CON}(\mathbf{P}, \mathbf{F}, \mathbf{C}),$$

where CON is *non-empty, finite set* of propositional connectives such that $CON = C_1 \cup C_2$ for C_1 a finite set (possibly empty) of unary connectives, C_2 a finite set (possibly empty) of binary connectives of the language $\mathcal{L}$. We define the set $\mathcal{F}$ of all well formed formulas of the predicate language $\mathcal{L}$ as follows.

Definition 8.3 (Formulas)

The set $\mathcal{F}$ of all well formed formulas, called shortly set of formulas, of the language $\mathcal{L}_{CON}\mathbf{P}, \mathbf{F}, \mathbf{C})$ is the smallest set meeting the following conditions:

1. *any atomic formula of $\mathcal{L}_{CON}(\mathbf{P}, \mathbf{F}, \mathbf{C})$ is a formula, i.e.*

$$A\mathcal{F} \subseteq \mathcal{F};$$

2. *if A is a formula of $\mathcal{L}_{CON}(\mathbf{P}, \mathbf{F}, \mathbf{C})$, $\bigtriangledown$ is an one argument propositional connective, then $\bigtriangledown A$ is a formula of $\mathcal{L}_{CON}(\mathbf{P}, \mathbf{F}, \mathbf{C})$, i.e. if the following recursive condition holds*

$$\textit{if} \quad A \in \mathcal{F}, \quad \bigtriangledown \in C_1, \quad \textit{then} \quad \bigtriangledown A \in \mathcal{F};$$

3. *if A, B are formulas of $\mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$, $\circ$ is a two argument propositional connective, then $(A \circ B)$ is a formula of $\mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$, i.e. if the following recursive condition holds*

$$\textit{if} \ \ A \in \mathcal{F}, \quad \bigtriangledown \in C_2, \quad \textit{then} \quad (A \circ B) \in \mathcal{F};$$

4. if A is a formula of $\mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$ and x is a variable, then $\forall xA, \exists xA$ are formulas of $\mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$, i.e. if the following recursive condition holds

$$\text{if } \ A \in \mathcal{F}, \quad x \in VAR, \quad \forall, \exists \in \mathbf{Q}, \quad \text{then } \ \forall xA, \quad \exists xA \in \mathcal{F}.$$

In formulas $\forall xA$, $\exists xA$, the formula A is in the **scope of the quantifier** $\forall$, $\exists$, *respectively.*

Example 8.6

Let $\mathcal{L}$ be a language with with the set $\{\cap, \cup, \Rightarrow, \neg\}$ of connectives and with two functional symbols: one -place and one two-place, one two-place predicate symbol, and two constants. We write $\mathcal{L}$ as

$$\mathcal{L} = \mathcal{L}_{\{\neg, \cap, \cup, \Rightarrow\}}(\{R\}, \{f, g\}, \{c, d\},)$$

where $\#f = 1$, $\#g = 2$, $\#R = 2$. Some of the formulas of $\mathcal{L}$ are the following.

$$R(c, f(d)), \quad \exists xR(x, f(c)), \quad \neg R(x, y), \quad \forall z(\exists xR(x, f(c)) \Rightarrow \neg R(x, y)),$$

$$(R(c, d) \cap \exists xR(x, f(c))), \quad \forall yR(y, g(c, g(x, f(c)))), \quad \forall y\neg\exists xR(x, y).$$

The formula $R(x, f(c))$ is in a **scope** of the quantifier $\exists x$ in $\exists xR(x, f(c))$.

The formula $(\exists xR(x, f(c)) \Rightarrow \neg R(x, y))$ isn't in a **scope** of any quantifier.

The formula $(\exists xR(x, f(c)) \Rightarrow \neg R(x, y))$ is in the **scope** of $\forall$ in $\forall z(\exists xR(x, f(c)) \Rightarrow \neg R(x, y))$.

Example 8.7

Let $\mathcal{L}$ be a language with with the set $\{\neg, \Box, \Diamond, \cap, \cup, \Rightarrow\}$ of connectives and **P**, **F**, *and* **C** *the same as in previous exercise, i.e.*

$$\mathcal{L} = \mathcal{L}_{\{\neg, \Box, \Diamond, \cap, \cup, \Rightarrow\}}(\{R\}, \{f, g\}, \{c, d\},)$$

where $\#f = 1$, $\#g = 2$, $\#R = 2$.

$\mathcal{L}$ is now a language of some first order **modal** *logic. Some of the formulas of $\mathcal{L}$ are the following.*

$$\Diamond\neg R(c, f(d)), \quad \Diamond\exists x\Box R(x, f(c)), \quad \neg\Diamond R(x, y), \quad \forall z(\exists xR(x, f(c)) \Rightarrow \neg R(x, y)),$$

$$(R(c, d) \cap \exists xR(x, f(c))), \quad \forall y\Box R(y, g(c, g(x, f(c)))), \quad \Box\forall y\neg\Diamond\exists xR(x, y).$$

The formula $\Box R(x, f(c))$ is in a **scope** of the quantifier $\exists x$ in $\Diamond\exists x\Box R(x, f(c))$.

The formula $(\exists xR(x, f(c)) \Rightarrow \neg R(x, y))$ isn't in a **scope** of any quantifier.

The formula $(\exists x R(x, f(c)) \Rightarrow \neg R(x, y))$ is in the **scope** of $\forall z$ in $\forall z(\exists x R(x, f(c)) \Rightarrow \neg R(x, y))$. Formula $\neg\Diamond\exists x R(x, y)$ is in the **scope** of $\forall y$ in $\Box\forall y\neg\Diamond\exists x R(x, y)$.

Given a predicate language $\mathcal{L} = (\mathcal{A}, T, \mathcal{F})$, we must distinguish between formulas like $P(x, y)$, $\forall x P(x, y)$ and $\forall x\exists y P(x, y)$.

This is done by introducing the notion of **free** and **bound variables**, **open** and **closed** formulas (sentences). Before we formulate proper definitions, here are some simple observations.

1. Some formulas are without quantifiers.
For example formulas $R(c_1, c_2)$, $R(x, y)$, $(R(y, d) \Rightarrow R(a, z))$. A formula without quantifiers is called an **open formula**.

Variables x, y in $R(x, y)$ are called **free variables**. The variables y in R(y, d) and z in R(a,z) are also **free**.

2. Quantifiers **bind** variables within formulas.

The variable x is **bounded** by $\exists x$ in the formula $\exists x R(x, y)$, the variable y is **free**. The variable y is **bounded** by $\forall y$ in the formula $\forall y R(x, y)$, the variable y is **free**.

3. The formula $\exists x\forall y R(x, y)$ does not contain any free variables, neither does the formula $R(c_1, c_2)$. A formula without any free variables is called a **closed formula** or a **sentence**.

Sometimes in order to distinguish more easily *which variable* is **free** and which is **bound** in the formula we might use the bold face type for the quantifier bound variables and write the formulas as follows.

$$(\forall \mathbf{x} Q(\mathbf{x}, y), \quad \exists \mathbf{y} P(\mathbf{y}), \quad \forall \mathbf{y} R(\mathbf{y}, g(c, g(x, f(c)))),$$
$$(\forall \mathbf{x} P(\mathbf{x}) \Rightarrow \exists \mathbf{y} Q(x, \mathbf{y})), \quad (\forall \mathbf{x}(P(\mathbf{x}) \Rightarrow \exists \mathbf{y} Q(\mathbf{x}, \mathbf{y})))$$

Observe that the formulas $\exists \mathbf{y} P(\mathbf{y})$, $(\forall \mathbf{x}(P(\mathbf{x}) \Rightarrow \exists \mathbf{y} Q(\mathbf{x}, \mathbf{y})))$ are **closed**. We call a close formula a **sentence**.

Example 8.8

Consider atomic formulas: $P(y), Q(x, c), R(z), P_1(g(x, y), z)$. *Here are some non atomic formulas formed out of them.*

1. $(P(y) \cup \neg Q(x, c)) \in \mathcal{F}$. *This is an* **open** *formula A with two free variables x,y. We denote A this as formula* $A(x, y)$.

2. $\exists \mathbf{x}(P(y) \cup \neg Q(\mathbf{x}, c)) \in \mathcal{F}$. *We write* $\mathbf{x}$ *to denote that x is a* **bound** *variable. The variable y is* **free**. *This is a formula B with one free variable y. We denote B as a formula* $B(y)$.

3. $\forall \mathbf{y}(P(\mathbf{y}) \cup \neg Q(x, c)) \in \mathcal{F}$. *The variable y is* **bound***, the variable x is* **free**. *We denote this formula by for example* $A_1(x)$.

4. $\forall \mathbf{y} \exists \mathbf{x}(P(y) \cup \neg Q(\mathbf{x}, c)) \in \mathcal{F}$ *has no free variables. It is a* **closed** *formula called also a* **sentence**.

Exercise 8.1

Given the following formulas of $\mathcal{L}$*:*

$P(x, f(c, y))$, $\exists c P(x, f(c, y))$, $\forall x f(x, P(c, y))$, $\exists x P(x, f(c, y)) \Rightarrow \forall y P(x, f(c, y))$.

1. Indicate whether they are, or are not well formed formulas of $\mathcal{F}$*. For those which are not in* $\mathcal{F}$ *write a correct formula.*
2. For each correct, or corrected formula identify all components: connectives, quantifiers, predicate and function symbols, and list all its terms.
3. For each formula identify its s free and bound variables. State which are open and which are closed formulas (sentences), if any.

Solution
Formula $A_1 = P(x, f(c, y))$.
It is a correct **atomic** formula. P is a 2 argument *predicate* symbol, f is a 2 argument *function* symbol, c is a *constant*. We write it symbolically: $P \in \mathbf{P}$, $f \in \mathbf{F}$, $c \in \mathbf{C}$. It is an **open** formula with two *free variables* x,y. We denote it by $A_1(x, y)$. It has no *bound variables*.

Formula $A_2 = \exists c P(x, f(c, y))$.

It is a not a correct formula, i.e. $\exists c P(x, f(c, y)) \notin \mathcal{F}$. The expression $\exists c$ has no meaning because c is a constant, not a variable.

The corrected formulas are: $B_1 = \exists x P(x, f(c, y))$, $B_2 = \exists y P(x, f(c, y))$, and formulas $B = \exists z P(z, f(c, y))$ for any variable z different then x and y.

None of the correct formulas are open. Variable y is free in $B_1 = B_1(y)$, variable x is free in $B_2 = B_2(x)$, both variables x and y are free in all formulas $B = B(x, y)$. All formulas are nether close, nor open. The *terms* appearing in any of them are the same as in $A_1 = P(x, f(c, y))$ and are: $x, y, c, f(c, y)$.

Formula $A_3 = \forall x f(x, P(c, y))$.

It is a not a correct formula, i.e. $\forall x f(x, P(c, y)) \notin \mathcal{F}$. The function symbol f in front $f(x, P(c, y))$ indicate a term and terms are not formulas. Moreover, the atomic formula $P(c, y)$ can't be put inside a term!

Formula $A_4 = \exists x P(x, f(c, y)) \Rightarrow \forall y P(x, f(c, y))$.

It is a not a correct formula. The correct formula is $A = (\exists x P(x, f(c, y)) \Rightarrow \forall y P(x, f(c, y)))$. It has two free variables x and y and we write it as $A = A(x, y)$.

Informally, in the formula $P(x, y)$ both variables x and y are called **free** variables. They are not in the scope of any quantifier. The formula of that type (without quantifiers) is an **open** formula.

The formal definition of the set of free variables of a formula is the following.

Definition 8.4 (Free and Bound Variables)

The set $FV(A)$ of free variables of a formula A is defined by the induction of the degree of the formula as follows.

1. *If A is an atomic formula, i.e. $A \in \mathcal{AF}$, then $FV(A)$ is just the set of variables appearing in the expression A;*
2. *for any unary propositional connective, i.e any $\bigtriangledown \in C_1$,*

   $FV(\bigtriangledown A) = FV(A)$,

   i.e. the free variables of $\bigtriangledown A$ are the free variables of A;
3. *for any binary propositional connective, i.e any $\circ \in C_2$,*

   $FV(A \circ B) = FV(A) \cup FV(B)$,

   i.e. the free variables of $(A \circ B)$ are the free variables of A together with the free variables of B;
4. $FV(\forall xA) = FV(\exists xA) = FV(A) - \{x\}$,

   i.e. the free variables of $\forall xA$ and $\exists xA$ are the free variables of A, except for x.

A formula with no free variables is called a **sentence**.
A variable is called **bound** *if it is* **not free**.
A formula with no bound variables is called an **open formula**.

Example 8.9 *The formulas* $\exists xQ(c, g(x, d))$, $\neg\forall x(P(x) \Rightarrow \exists y(R(f(x), y) \cap \neg P(c)))$ *are* **sentences**. *The formulas* $Q(c, g(x, d))$, $\neg(P(x) \Rightarrow (R(f(x), y) \cap \neg P(c)))$ *are* **open** *formulas. The formulas* $\exists xQ(c, g(x, y))$, $\neg(P(x) \Rightarrow \exists y(R(f(x), y) \cap \neg P(c)))$ *are* **neither** *sentences* **nor** *open formulas. They contain some free and some bound variables; the variable y is free in the first formula, the variable x is free in the second.*

The definition 8.1 defines a predicate language $\mathcal{L} = \mathcal{L}_{CON}(\mathbf{P}, \mathbf{F}, \mathbf{C})$ (8.3) with its sets of predicate, function and constant symbol possibly countably infinite sets. We use its most general case with sets of predicate, function and constant symbol all countably infinite sets for defining all relevant notions concerning provability and semantics. In particular, we will define in detail the classical semantics for this most general form of $\mathcal{L}$ and prove the completeness theorem for classical predicate logic based on it.

When we deal with formal theory $\mathbf{Th}(SA)$ with a set SA of specific axioms we restrict the language $\mathcal{L}_{CON}(\mathbf{P}, \mathbf{F}, \mathbf{C})$ to the symbols characteristic for that theory. We hence introduce the following definition.

Definition 8.5

Given a language $\mathcal{L}_{CON}(\mathbf{P}, \mathbf{F}, \mathbf{C}) = (\mathcal{A}, \mathbf{T}, \mathcal{F})$.

Let $\mathcal{F}_0 \subseteq \mathcal{F}$ *be a non-empty, finite subset of formulas of* $\mathcal{L}$. *Denote by* $\mathbf{P}_0,\ \mathbf{F}_0, \mathbf{C}_0$ *the sets of all predicate, function, and constant symbols appearing in the formulas from the set* $\mathcal{F}_0$. *The language*

$$\mathcal{L}_{CON}(\mathbf{P}_0,\ \mathbf{F}_0, \mathbf{C}_0)$$

is called a **language defined** *by the set* $\mathcal{F}_0$ *of formulas.*

Example 8.10 *Consider a language* $\mathcal{L} = \mathcal{L}_{\{\neg,\Rightarrow,\cup,\cap\}}(\mathbf{P}, \mathbf{F}, \mathbf{C})$ *and a following set* $\mathcal{F}_0$ *of formulas of* $\mathcal{L}$

$\mathcal{F}_0 = \{\exists x Q(c, g(x, d)),\ \neg\forall x(P(x) \Rightarrow \exists y(R(f(x), y)\cap\neg P(e))), \neg(F(a)\cap R(y, h(c))\}$.

A **language defined** *by the set* $\mathcal{F}_0$ *of formulas is*

$$\mathcal{L}_{\{\neg,\Rightarrow,\cup,\cap\}}(\{P, R, Q, F\},\ \{g, f, h\},\ \{a, c, d, e\}),$$

where $\#\ Q = \#R = 2$, $\#P = \#\ F = 1$, $\#\ g = 2$, $\#f = \#\ h = 1$.

It is common practice to use the notation

$$A(x_1, x_2, ..., x_n) \tag{8.6}$$

to indicate that $FV(A) \subseteq \{x_1, x_2, ..., x_n\}$ without implying that all of $x_1, x_2, ..., x_n$ are actually free in A. This is similar to the practice in algebra of writing $p(x_1, x_2, ..., x_n)$ for a polynomial p in the variables $x_1, x_2, ..., x_n$ without implying that all of them have nonzero coefficients.

Definition 8.6 (Replacing x by t in A)

If $A(x)$ is a formula, and t is a term then $A(x/t)$ or, more simply, $A(t)$ denotes the result of replacing all occurrences of the free variable x by the term t throughout. When using the notation $A(t)$ we always assume that none of the variables in t occur as bound variables in A.

The assumption that none of the variables in t occur as bound variables in A is essential, otherwise by substituting t on the place of x we would distort the meaning of $A(t)$. Let $t = y$ and $A(x)$ is $\exists y(x \neq y)$, i.e. the variable y in t is bound in A. The substitution of t for x produces a formula $A(t)$ of the form $\exists y(y \neq y)$, which has a different meaning than $\exists y(x \neq y)$.

But if $t = z$, i.e. the variable z in t is not bound in A, then $A(x/t) = A(t)$ is $\exists y(z \neq y)$ and express the same meaning as $A(x)$.

Remark that if for example $t = f(z, x)$ we obtain $\exists y(f(z, x) \neq y)$ as a result of substitution of $t = f(z, x)$ for x in $\exists y(x \neq y)$.

This notation is convenient because we can agree to write as

$$A(t_1, t_2, ..., t_n) \quad \text{or} \quad A(x_1/t_1, x_2/t_2, \ldots, x_n/t_n)$$

a result of substituting in A the terms $t_1, t_2, \ldots, t_n$ for all free occurrences (if any) of $x_1, x_2, \ldots, x_n$, respectively. But when using this notation we always assume that none of the variables in $t_1, t_2, ..., t_n$ occur as bound variables in A.

The above assumption that none of the variables in $t_1, t_2, ..., t_n$ occur as bound variables in A is often expressed using the notion: $t_1, t_2, \dot{;} t_n$ *are free for all theirs variables in* A which is defined formally as follows.

Definition 8.7 (Term t free for y in A)

If $A \in \mathcal{F}$ and t is a term, then t is said to be **free for** y *if no free occurrence of y lies within the scope of any quantifier bounding variables in t.*

Example 8.11 *Let A , B be the formulas*

$$\forall y P(f(x, y), y), \quad \forall y P(f(x, z), y),$$

respectively. The term $t = f(x, y)$ is free for x and is not free for y in A. The term $t = f(x, z)$ is free for x and z in B. The term $t = y$ is not free neither for x nor for z in A, B.

Example 8.12

Let A be a formula

$$(\exists x Q(f(x), g(x, z)) \cap P(h(x, y), y)).$$

The term $t_1 = f(x)$ is not free for x in A; the term $t_2 = g(x, z)$ is free for z only, term $t_3 = h(x, y)$ is free for y only because x occurs as a bound variable in A; term t_4.

Definition 8.8 (Replacement)

If $A(x), A(x_1, x_2, ..., x_n) \in \mathcal{F}$ and $t, t_1, t_2, ..., t_n \in T$, then $A(x/t)$, $A(x_1/t_1, x_2/t_2, \ldots, x_n/t_n)$ or, more simply just

$$A(t), \; A(t_1, t_2, ..., t_n)$$

denotes the result of replacing all occurrences of the free variables x, $x_1, x_2, ..., x_n$, by the terms t, $t, t_1, t_2, ..., t_n$, respectively, assuming that $t, t_1, t_2, ..., t_n$ are free for all theirs variables in A.

Classical Restricted Domain Quantifiers

We often use logic symbols, while writing mathematical statements. For example mathematicians in order to say "all natural numbers are greater then zero and some integers are equal 1" often write it as

$$x \geq 0, \forall_{x \in N} \text{ and } \exists_{y \in Z},\ y = 1.$$

Some of them, who are more "logic oriented", would also write it as

$$\forall_{x \in N}\ x \geq 0\ \cap\ \exists_{y \in Z}\ y = 1,$$

or even as

$$(\forall_{x \in N}\ x \geq 0\ \cap\ \exists_{y \in Z}\ y = 1).$$

None of the above symbolic statements are formulas of the *predicate language* $\mathcal{L}$. These are mathematical statement written with mathematical and logic symbols. They are written with different degree of "logical precision", the last being, from a logician point of view the most precise.

Observe that the quantifiers in $\forall_{x \in N}$ and $\exists_{y \in Z}$ used in all of them are not the one used in the predicate language $\mathcal{L}$, which admits only quantifiers $\forall x$ and $\exists y$, for any variables $x, y \in VAR$. The quantifiers $\forall_{x \in N}$, $\exists_{y \in Z}$ are called **quantifiers with restricted domain**. The first is restricted to the domain of natural numbers, the second to the integers. The restriction of the quantifier domain can, and often is given by more complicated statements. For example we say "for all $x > 2$" and write $\forall_{x > 2}$, or we say "exists $x > 2$ and at same time $x + 2 < 8$" and write symbolically $\exists_{(x > 2 \cap x + 2 < 8)}$.

Our goal now is to correctly "translate " mathematical and natural language statement into formulas of the predicate language $\mathcal{L}$ of the classical predicate logic with the the set $\{\neg . \cap, \cup, \Rightarrow\}$ of propositional connectives. We say "classical predicate logic" to express that we define all notions for the classical semantics to be defined formally in the next section 8.2. One can extend these notions to non-classical logics, but we describe and will talk only about classical case. We introduce the quantifiers with restricted domain into the classical predicate logic language by expressing them within the language $\mathcal{L}$ as follows.

Definition 8.9

Given a classical predicate logic language

$$\mathcal{L} = \mathcal{L}_{\{\neg, \cap, \cup, \Rightarrow, \neg\}}(\mathbf{P}, \mathbf{F}, \mathbf{C}).$$

The quantifiers $\forall_{A(x)}$, $\exists_{A(x)}$ *are called* **quantifiers with restricted domain**, *or* **restricted quantifiers**, *where* $A(x) \in \mathcal{F}$ *is any formula with any free variable* $x \in VAR$.

A formula $\forall_{A(x)} B(x)$ *is an abbreviation of a formula* $\forall x(A(x) \Rightarrow B(x)) \in \mathcal{F}$. *We write it symbolically as*

$$\forall_{A(x)}\ B(x) = \forall x(A(x) \Rightarrow B(x)). \tag{8.7}$$

A formula $\exists_{A(x)}B(x)$ *stands for a formula* $\exists x(A(x)\cap B(x))\in\mathcal{F}$. *We write it symbolically as*

$$\exists_{A(x)}\ B(x) = \exists x(A(x)\cap B(x)) \tag{8.8}$$

The definition 8.9 of restricted quantifiers is obviously faithful to our intuitive meaning of quantifiers. We use informally a symbol = to stress that we they are in a sense equivalent with respect to classical semantics. We call (8.7) and (8.8) **transformations rules** for restricted quantifiers.

Exercise 8.2

Given a mathematical statement **S** *written with logical symbols*

$$(\forall_{x\in N}\ x\geq 0\ \cap\ \exists_{y\in Z}\ y=1)$$

1. Translate it into a proper logical formula with restricted domain quantifiers i.e. into a formula of $\mathcal{L}$ *that uses the restricted domain quantifiers.*

2. Translate your restricted domain quantifiers logical formula into a correct logical formula **without** *restricted domain quantifiers, i.e. into a formula of* $\mathcal{L}$.

Solution
We proceed to write this and other similar problems solutions in a sequence of steps.

STEP 1. We identify *basic statements* in **S**, i.e. mathematical statements that involve only relations. They will be translated into *atomic formulas*. We identify the *relations* in the basic statements and choose the *predicate symbols* as their names. We identify all *functions* and *constants* (if any) in the basic statements and choose the *function symbols* and *constant symbols* as their names.

The basic statements in **S** are: $x\in N,\ x\geq 0,\ y\in Z,\ y=1$. The relations are: $\in N,\ \in Z,\ \geq,\ =$. We use one argument predicates symbols N, Z for $\in N, \in Z$, respectively. We use two argument predicate symbols G for $\geq$, and E for $=$. There are no functions. We have two constant symbols c_1, c_2 for numbers 0 and 1, respectively.

STEP 2. We write the basic statements as *atomic formulas* of $\mathcal{L}$.

We write $N(x), Z(x)$ for $x\in N, x\in Z$, respectively. $G(x, c_1)$ for $x\geq 0$ and $E(y, c_2)$ for $y=1$. These are all atomic formulas.

STEP 3. We re-write the statement **S** a logical formula with restricted domain quantifiers.

The statement **S** becomes a restricted quantifiers formula:

$$(\forall_{N(x)}\ G(x, c_1)\ \cap\ \exists_{Z(y)}\ E(y, c_2)).$$

STEP 4. We apply (8.7) and (8.8) to the formula from STEP 3. and obtain a formula A of $\mathcal{L}$ as a representation of the given mathematical statement **S**.

A formula $A \in \mathcal{F}$ that corresponds to **S** is

$$(\forall x\ (N(x) \Rightarrow G(x, c_1))\ \cap\ \exists y\ (Z(y) \cap E(y, c_2))).$$

Here is a perfectly acceptable short solution. We presented the long solution in order to explain all steps needed to be performed when one writes a solution.

Short Solution
The *basic statements* in **S** are: $x \in N,\ x \geq 0,\ y \in Z, y = 1$. The corresponding *atomic formulas* of $\mathcal{L}$ are: $N(x),\ G(x, c_1),\ Z(y),\ E(y, c_2)$, respectively.

The statement **S** becomes becomes a restricted quantifiers formula $(\forall_{N(x)} G(x, c_1)\ \cap\ \exists_{Z(y)}\ E(y, c_2))$. Applying restricted quantifiers definition 8.9 and transformation rules (8.7), (8.8) we get a following formula $A \in \mathcal{F}$

$$(\forall x(N(x) \Rightarrow G(x, c_1)) \cap \exists y(Z(y) \cap E(y, c_2))).$$

8.2 Classical Semantics

The notion of predicate tautology is much more complicated then that of the propositional. Predicate tautologies are also called *valid formulas*, or *laws of quantifiers* to distinguish them from the propositional case. The formulas of a predicate language $\mathcal{L}$ have meaning only when an *interpretation* is given for all its symbols. We define an interpretation I by interpreting predicate, functional symbols as a concrete relation, function defined in a certain set $U \neq \emptyset$, and constants symbols as elements of the set U. The set U is called the *universe* of the *interpretation* I. These two items specify a *structure* $\mathbf{M} = (U, I)$ for the language $\mathcal{L}$.
.

The semantics for a first order language $\mathcal{L}$ in general, and for the first order classical logic in particular, is defined, after Tarski (1936) in terms of the *structure* $\mathbf{M} = [U, I]$, an assignment s of $\mathcal{L}$, and a *satisfaction relation* $(\mathbf{M}, s) \models A$ between structures, assignments and formulas of $\mathcal{L}$.

The definition of a structure $\mathbf{M} = [U, I]$ and the assignment s of $\mathcal{L}$ is common for different predicate languages and for different semantics and we define them as follows.

Definition 8.10 (Structure)

Given a predicate language $\mathcal{L} = \mathcal{L}_{CON}(\mathbf{P}, \mathbf{F}, \mathbf{C})$. A **structure** *for $\mathcal{L}$ is a pair*

$$\mathbf{M} = [U, I],$$

where U is a non empty set called a **universe** *and I is an assignment called an* **interpretation** *of the language $\mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$ in the universe U defined as follows.*

1. *I assigns to any predicate symbol $P \in \mathbf{P}$ a relation P_I defined in the universe U. I.e. for any $P \in \mathbf{P}$, if $\#P = n$, then*

$$P_I \subseteq U^n.$$

2. *I assigns to any functional symbol $f \in \mathbf{F}$ a function f_I defined in the universe U. I.e. for any $f \in \mathbf{F}$, if $\#f = n$, then*

$$f_I : \quad U^n \longrightarrow U.$$

3. *I assigns to any constant symbol $c \in \mathbf{C}$ an element c_I of the universe. I.e for any $c \in \mathbf{C}$,*

$$c_I \in U.$$

Example 8.13

Let $\mathcal{L}$ be a language with one two-place predicate symbol, two functional symbols: one -place and one two-place, and two constants, i.e.

$$\mathcal{L} = \mathcal{L}(\{R\}, \{f, g\}, \{c, d\},)$$

where $\#R = 2$, $\#f = 1$, $\#g = 2$, and $c, d \in \mathbf{C}$.

We define a structure $\mathbf{M}$ = [U, I] as follows. We take as the universe the set $U = \{1, 3, 5, 6\}$. The predicate R is interpreted as $\leq$, what we write as $R_I : \leq$. We interpret f as a function $f_I : \{1, 3, 5, 6\} \longrightarrow \{1, 3, 5, 6\}$ such that $f_I(x) = 5$ for all $x \in \{1, 3, 5, 6\}$, and we put $g_I : \{1, 3, 5, 6\} \times \{1, 3, 5, 6\} \longrightarrow \{1, 3, 5, 6\}$ such that $g_I(x, y) = 1$ for all $x \in \{1, 3, 5, 6\}$. The constant c becomes $c_I = 3$, and $d_I = 6$, what we write as We write the structure $\mathbf{M}$ as

$$\mathbf{M} = [\{1, 3, 5, 6\} \quad \leq, \ f_I, \ g_I, \ c_I = 3, \ d_I = 6]$$

Exercise 8.3

Given a language

$$\mathcal{L} = \mathcal{L}(\{R\}, \{g\}, \emptyset,)$$

where $\#R = 2$, $\#g = 2$. Define two structures for $\mathcal{L}$, both with infinite universe: one infinitely countable and one uncountable.

Solution
There are many such structures. Here are two of the very simple.

$\mathbf{M}_1 = [N, \ , \leq, \ +]$, where N is the set of natural numbers, and for example $\mathbf{M}_2 = [R, \ , \leq, \ +]$, where R is the set of real numbers.

Definition 8.11 (Assignment)

Given a first order language $\mathcal{L} = \mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$ with the set VAR of variables. Let $\mathbf{M} = [U, I]$ be a structure for $\mathcal{L}$ with the universe $U \neq \emptyset$.

An **assignment** *of $\mathcal{L}$ in $\mathbf{M} = [U, I]$ is any function*

$$s : \quad VAR \longrightarrow U \tag{8.9}$$

The assignment s is also called an **interpretation** *of variables VAR of $\mathcal{L}$ in a structure $\mathbf{M} = [U, I]$.*

Let $\mathbf{M} = [U, I]$ be a structure for $\mathcal{L}$ and $s : VAR \longrightarrow U$ be an **interpretation** of variables VAR of $\mathcal{L}$ in a structure $\mathbf{M}$.

Let $\mathbf{T}$ be the set of all terms of $\mathcal{L}$. By definition 10.11, $VAR \subset \mathbf{T}$. We use the interpretation I to **extend** the assignment s to the set the set $\mathbf{T}$ of all terms of $\mathcal{L}$. Because of that we denote this extension by I rather then by s^* as we did before. The extension I of v is hence a mapping from $\mathbf{T}$ to U. It associates with each $t \in \mathbf{T}$ an element $I(t) \in U$. We denote this element $I(t)$ by t_I. We define the extension $I(t) = t_I$ of s by the induction of the length of the term $t \in \mathbf{T}$ and call it an *interpretation of terms* of $\mathcal{L}$ in a structure $\mathbf{M} = [U, I]$.

Definition 8.12 (Interpretation of Terms)

Given a language $\mathcal{L} = \mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$ and a structure $\mathbf{M} = [U, I]$ for $\mathcal{L}$. Let an assignment

$$s : VAR \longrightarrow U$$

be any interpretation of variables VAR of $\mathcal{L}$ (assignment) in $\mathbf{M}$.

We **extend** *s to a function*

$$s_I : \mathbf{T} \longrightarrow U \tag{8.10}$$

called an **interpretation** *s_I* **of terms** *of $\mathcal{L}$ in $\mathbf{M}$.*

The function s_I is defined by induction on the complexity of terms as follows.

1. *For any $x \in VAR$,*

$$s_I(x) = s(x);$$

2. *for any $c \in \mathbf{C}$,*

$$s_I(c) = c_I;$$

3. *for any* $t_1,\ t_2,\ \ldots, t_n \in \mathbf{T},\ n \geq 1,\ f \in \mathbf{F}$, *such that* $\#f = n$, *and for any term* $t = f(t_1,\ t_2,\ \ldots, t_n)$, *we put*

$$s_I(t) = f_I(s_I(t_1),\ s_I(t_2),\ \ldots,\ s_I(t_n)),$$

i.e. we define

$$s_I(f(t_1,\ t_2,\ \ldots, t_n)) = f_I(s_I(t_1),\ s_I(t_2),\ \ldots,\ s_I(t_n)),$$

for any $t_1,\ t_2,\ \ldots, t_n \in \mathbf{T},\ f \in \mathbf{T}$, *such that* $\#f = n$.

Exercise 8.4

Consider a language

$$\mathcal{L} = \mathcal{L}(\{P, R\},\ \{f,\ g,\ h,\},\ \{c, d\})$$

for $\#\ P = \#\ R = 2,\ \ \#f = \#\ g{=}1,\ \ \#\ h{=}\ 2$.

Let $\mathbf{M} = [\{0,1\}, I]$, *where the interpretation* I *is defined as follows.*

$P_I = \{(0,0)\},\quad R_I = \{(0,0),(1,1)\}$,

$f_I(0) = 0,\quad f_I(1) = 0,\ g_I(0) = 1,\ g_I(1) = 1,\quad h_I$ *is given by a formula*

$h_I(x,y) = x$ *for all* $(x,y) \in \{0,1\} \times \{0,1\}$, *and* $c_I = 1,\ d_I = 0$.

The assignment $s : VAR \longrightarrow \{0,1\}$ *is such that* $s(x) = s(y) = s(z) = 1 = 1$, *for* $x, y, z \in VAR$ *(and any values for all other variables).*

Given a set $\mathbf{T}_0$ *of terms, evaluate* $s_I(t)$ *for all* $t \in \mathbf{T}_0$.

$\mathbf{T}_0 = \{z, y, x, c,\ f(c), f(x), g(z), f(g(d)), g(f(g(z))), h(c, f(g(d))), h(f(x), g(z))\}$.

Solution
First we evaluate terms that are variables and constants of $\mathcal{L}$ using the formulas 1. and 2. of definition 8.12: $s_I(x) = s(x), s_I(c) = c_I$, respectively and obtain: $s_I(z) = s(z) = 1,\ s_I(y) = s(y) = 1,\ s_I(x) = s(x) = 1,\ s_I(c) = c_I = 1,\ s_I(d) = d_I = 2$. We use the formula $s_I(f(t_1,\ t_2,\ \ldots, t_n)) = f_I(s_I(t_1),\ s_I(t_2),\ \ldots,\ s_I(t_n)$ to evaluate the rest of terms in $\mathbf{T}_0$ and obtain:

$s_I f(c) = f_I(s_I(c)) = f_I(c_I)) = f_I(1) = 0,\quad s_I f(x) = f_I(s_I(x)) = f_I(1) = 0$

$s_I g(z) = g_I(s_I(c)) = g_I(1) = 1,\quad s_I(f(g(d)) = f_I(s_I(f(g(d))) = f_I(g_I(s_I(d))) = f_I(g_I(c_I)) = f_I(g_I(1)) = f_I(1) = 0$,

$s_I g(f(g(z)) = g_I(f_I(g_I(s_I(z)))) = g_I(f_I(g_I(1))) = g_I(f_I(1)) = g_I(0) = 1$,

$s_I(h(c, f(g(d)))) = h_I(s_I(c), s_I(f(g(d)))) = h_I(c_I, f_I(g_I(s_I(d))))$

$= h_I(1, f_I(g_I(0))) = h_I(1, f_I(1)) = h_I(1,0) = 1$,

$$s_I(h(f(x), g(z))) = h_I(f_I(s_I(x)), g_I(s_I(x))) = h_I(f_I(1), g_I(1)) = h_I(0, 1) = 0.$$

Observe that the interpretation of predicate symbols is irrelevant when evaluating an interpretation of terms, as the terms do not involve the predicate symbols.

Example 8.14

Consider a language

$$\mathcal{L} = \mathcal{L}(\{P,\ R\},\ \{f, h\},\ \emptyset\ )$$

for $\#\ P = \#\ R = 2,\ \ \#f = 1,\ \ \#\ h = 2.$

Let $\mathbf{M} = [Z, I]$*, where* Z *is the set on integers and the interpretation* I *for elements of* $\mathbf{F}$ *and* $\mathbf{C}$ *is as follows.*

$f_I : Z \longrightarrow Z$ *is given by formula* $f(m) = m + 1$ *for all* $m \in Z$*.*

$h_I : Z \times Z \longrightarrow Z$ *is given by formula* $f(m, n) = m + n$ *for all* $m, n \in Z$*.*

Let $s : VAR \longrightarrow Z$ *be any assignment such that* $s(x) = -5,\ s(y) = 2$ *and* $t_1, t_2 \in \mathbf{T}$ *be* $t_1 = h(y, f(f(x)))$ *and* $t_2 = h(f(x), h(x, f(y))$*.*

We evaluate:

$$s_I(t_1) = s_I(h(y, f(x)) = h_I(s_I(y), f_I(s_I(x))) = +(2, f_I(-5)) = 2 - 4 = -2,$$

$$s_I(t_2) = s_I(h(f(x), h(x, f(y))) = +(f_I(-5), +(-5, 3)) = -4 + (-5 + 3) = -6.$$

For any $t \in \mathbf{T}$, let $x_1, x_2,\ \dots,\ x_n \in VAR$ be all variables appearing in t, we write it, in a similar way as we did in (8.6) for variables in formulas, as

$$t(x_1, x_2,\ \dots,\ x_n).$$

Observation 8.1

For any term $t(x_1, x_2,\ \dots,\ x_n) \in \mathbf{T}$*, any structure* $\mathbf{M} = [U, I]$ *and any assignments* s, s' *of* $\mathcal{L}$ *in* $\mathbf{M}$*, the following holds.*

If $s(x) = s'(x)$ *for all* $x \in \{x_1, x_2,\ \dots,\ x_n\}$*, i.e the assignments* s, s' *agree on all variables appearing in* t*, then,* $s_I(t) = s'_I(t)$*.*

Thus for any $t \in \mathbf{T}$, the function $s_I : \mathbf{T} \longrightarrow U$ defined by (8.10) depends on only a finite number of values of $s(x)$ for $x \in VAR$.

Given a structure $\mathbf{M} = [U, I]$ and an assignment $s : VAR \longrightarrow U$. We write

$$s(\tfrac{a}{x}) \tag{8.11}$$

to denote any assignment $s' : VAR \longrightarrow U$ such that s, s' **agree** on all variables except on x, such that $s'(x) = a$, for certain $a \in U$.

Given a first order (predicate) language $\mathcal{L} = \mathcal{L}_{CON}(\mathbf{P}, \mathbf{F}, \mathbf{C})$. The satisfaction relation $(\mathbf{M}, s) \models A$ between structures, assignments and formulas of $\mathcal{L}$ is defined by induction on the complexity of formulas of $\mathcal{L}$. It is the satisfaction relation $(\mathbf{M}, s) \models A$ that allows us to **distinguish** one one semantics for a given $\mathcal{L}$ from the other, and consequently one logic from the other. We define now only a classical satisfaction and the notion of classical predicate tautology.

Definition 8.13 (Classical Satisfaction)

Given a classical predicate (first order) language

$$\mathcal{L} = \mathcal{L}_{\{\neg, \cap, \cup, \Rightarrow, \neg\}}(\mathbf{P}, \mathbf{F}, \mathbf{C}). \tag{8.12}$$

Let $\mathbf{M} = [U, I]$ be a structure for $\mathcal{L}$, s be any assignment of $\mathcal{L}$ in $\mathbf{M}$, i.e. $s : VAR \longrightarrow U$. Let $A \in \mathcal{F}$ be any formula of $\mathcal{L}$. We define a relation

$$(\mathbf{M}, s) \models A$$

that reads: the assignment s **satisfies** *the formula A in $\mathbf{M}$, by induction on the complexity of A as follows.*

(i) *A as atomic formula (8.5), i.e. A is $P(t_1, t_2, \dots, t_n)$ for $P \in \mathbf{P}$, $\#P = n$.*

$(\mathbf{M}, s) \models P(t_1, t_2, \dots, t_n)$ *if and only if* $(s_I(t_1), s_I(t_2), \dots, s_I(t_n)) \in P_I$.

(ii) *A as not atomic formula and has one of connectives of $\mathcal{L}$ as the main connective.*

$(\mathbf{M}, s) \models \neg A$ *if and only if* $(\mathbf{M}, s) \not\models A$,

$(\mathbf{M}, s) \models (A \cap B)$ *if and only if* $(\mathbf{M}, s) \models A$ *and* $(\mathbf{M}, s) \models B$,

$(\mathbf{M}, s) \models (A \cup B)$ *if and only if* $(\mathbf{M}, s) \models A$ *or* $(\mathbf{M}, s) \models B$ *or both,*

$(\mathbf{M}, s) \models (A \Rightarrow B)$ *if and only if ether* $(\mathbf{M}, s) \not\models A$ *or else* $(\mathbf{M}, s) \models B$ *or both.*

(iii) *A as not atomic formula and begins with one of the quantifiers.*

$(\mathbf{M}, s) \models \exists x A$ *if and only if there is s' such that s, s'* **agree** *on all variables except on x, and* $(\mathbf{M}, s') \models A$,

$(\mathbf{M}, s) \models \forall x A$ *if and only if for all s' such that s, s'* **agree** *on all variables except on x, and* $(\mathbf{M}, s') \models A$.

Observe that that the truth or falsity of $(\mathbf{M}, s) \models A$ depends only on the values of $s(x)$ for variables x which are actually free in the formula A. This is why we often write the condition **(iii)** as

(iii)' $A(x)$ as not atomic formula (with a free variable x) and begins with one of the quantifiers.

$(\mathbf{M}, s) \models \exists x A(x)$ if and only if there is s' such that $s(y) = s'(y)$ for all $y \in VAR - \{x\}$, and $(\mathbf{M}, s') \models A(x)$,

$(\mathbf{M}, s) \models \forall x A$ if and only if for all s' such that $s(y) = s'(y)$ for all $y \in VAR - \{x\}$, and $(\mathbf{M}, s') \models A$.

Exercise 8.5

For the structures $\mathbf{M}_i$, find assignments $s_i,\ s'_i$ $(1 \leq i \leq 4)$, such that

$$(\mathbf{M}_i, s_i) \models Q(x, c), \quad (\mathbf{M}_i, s'_i) \not\models Q(x, c) \quad \textit{for} \ \ Q \in \mathbf{P},\ c \in \mathbf{C}.$$

The structures $\mathbf{M}_i$ are defined as follows (the interpretation I for each of them is specified only for symbols in the formula $Q(x, c)$, and N denotes the set of natural numbers.

$$\mathbf{M}_1 = [\{1\},\ Q_I :=,\ c_I : 1], \qquad \mathbf{M}_2 = [\{1,2\},\ Q_I :\leq,\ c_I : 1],$$

$$\mathbf{M}_3 = [N,\ Q_I :\geq,\ c_I : 0], \quad \textit{and} \quad \mathbf{M}_3 = [N,\ Q_I :\geq,\ c_I : 1.]$$

Solution
Consider $\mathbf{M}_1 = [\{1\},\ Q_I :=,\ c_I : 1]$. Observe that all $s : VAR \longrightarrow \{1\}$ must are defined by a formula $s(x) = 1$ for all $x \in VAR$. We evaluate (definition 8.12), $s_I(x) = 1, s_I(c) = c_I = 1$. By definition 8.13, $(\mathbf{M}_1, s) \models Q(x, c)$ if and only if $(s_I(x),\ s_I(c)) \in Q_I$, i.e. $(1, 1) \in=$ what is true as $1 = 1$. We have proved

$$(\mathbf{M}_1, s) \models Q(x, c) \text{ for all assignments } s : VAR \longrightarrow \{1\}.$$

Consider $\mathbf{M}_2 = [\{1,2\},\ Q_I :\leq,\ c_I : 1]$. Let $s : VAR \longrightarrow \{1,2\}$ be any assignment, such that $s(x) = 1$. We evaluate $s_I(x) = 1,\ s_I(c) = 1$ and verify whether $(s_I(x),\ s_I(c)) \in Q_I$, i.e. whether $(1, 1) \in \leq$. This is true as $1 \leq 1$. We have found s (in fact uncountably many such s) such that

$$(\mathbf{M}_2, s) \models Q(x, c).$$

Let now s' be any assignment $s' : VAR \longrightarrow \{1,2\}$, such that $s'(x) = 2$. We evaluate $s'_I(x) = 1,\ s'_I(c) = 1$ and verify whether $s'_I(x),\ s'_I(c)) \in Q_I$, i.e. whether $(2, 1) \in \leq$. This is not true as $2 \not\leq 1$. We have found $s' \neq s$ (in fact uncountably many such s') such that

$$(\mathbf{M}_2, s') \not\models Q(x, c).$$

Consider $\mathbf{M}_3 = [N,\ Q_I :\geq,\ c_I : 0]$. Let $s : VAR \longrightarrow N$ be any assignment, such that $s(x) = 5$. We evaluate $s_I(x) = 5,\ s_I(c) = 0$. Observe that the condition $(s_I(x),\ s_I(c)) \in Q_I$ holds as $5 \geq 0$ and

$$(\mathbf{M}_3, s) \models Q(x, c).$$

Let now s' be any assignment $s' : VAR \longrightarrow N$. By definition, $s'(x) = n$, for any $n \in N$, and $s'_I(x),\ 0) \in Q_I$ holds for any s' as $n \geq 0$ for all $n \in N$. This proves that there is no s', such that $(\mathbf{M}_3, s') \not\models Q(x, c)$.

Consider $\mathbf{M}_4 = [N,\ Q_I :\geq,\ c_I : 1]$. Let $s : VAR \longrightarrow N$ be any assignment, such that $s(x) = 5$. We evaluate $s_I(x) = 5,\ s_I(c) = 1$. Observe that the condition $(s_I(x),\ s_I(c)) \in Q_I$ holds as $5 \geq 1$ and hence

$$(\mathbf{M}_4, s) \models Q(x, c).$$

Let now s' be any assignment $s' : VAR \longrightarrow N$, such that $s'(x) = 0$. The the condition $(s'_I(x),\ s'_I(c)) \in Q_I$ does not holds as $0 \not\geq 1$ and

$$(\mathbf{M}_4, s') \not\models Q(x, c).$$

Directly from the definition8.13 we have that the following holds.

Example 8.15

Let $\mathbf{M}_i$ $(1 \leq i \leq 4)$ be structures in defined the exercise 8.5 and let corresponding assignments s_i be as defined as its solutions.

1. $(\mathbf{M}_1, s) \models Q(x, c)$, $\quad (\mathbf{M}_1, s) \models \forall x Q(x, c)$, $\quad (\mathbf{M}_1, s) \models \exists x Q(x, c)$.

2. $(\mathbf{M}_2, s) \models Q(x, c)$, $\quad (\mathbf{M}_2, s) \not\models \forall x Q(x, c)$, $\quad (\mathbf{M}_1, s) \models \exists x Q(x, c)$.

3. $(\mathbf{M}_3, s) \models Q(x, c)$, $\quad (\mathbf{M}_3, s) \models \forall x Q(x, c)$, $\quad (\mathbf{M}_3, s) \models \exists x Q(x, c)$.

4. $(\mathbf{M}_4, s) \models Q(x, c)$, $\quad (\mathbf{M}_4, s) \not\models \forall x Q(x, c)$, $\quad (\mathbf{M}_4, s) \models \exists x Q(x, c)$.

Definition 8.14 (Model)

Given a language $\mathcal{L}$, a formula A of $\mathcal{L}$, and a structure $\mathbf{M} = [U, I]$ for $\mathcal{L}$.

The structure $\mathbf{M}$ is a **model** *for the formula A if and only if $(\mathbf{M}, s) \models A$ for all $s : VAR \longrightarrow U$. We denote it as $\mathbf{M} \models A$.*

For any set $\Gamma \subseteq \mathcal{F}$ of formulas of $\mathcal{L}$, $\mathbf{M}$ is a **model** *for Γ if and only if $\mathbf{M} \models A$ for all $A \in \Gamma$. We denote it as $\mathbf{M} \models \Gamma$.*

We define now a very important semantic notion. It has different names: *logical consequence, logical implication, semantic consequence, logical (semantical) entailment.* We use a name *logical consequence* and define it as follows.

Definition 8.15 (Logical Consequence)

For any $A, B \in \mathcal{F}$ and any set $\Gamma \subseteq \mathcal{F}$ of formulas of $\mathcal{L}$, we say that a formula B is a **logical consequence** *of a set Γ and write it as $\Gamma \models B$, if and only if all models of the set Γ are models of the formula B.*

When $\Gamma \models B$ we also say that Γ **logically implies** *B. When $\Gamma = \{A\}$ we write it as $A \models B$ and say A* **logically implies** *B.*

We say that A and B are **logically equivalent** *if and only if $A \models B$ and $A \models B$.*

Directly from the model definition 9.3 we get the following.

Definition 8.16 (Counter Model)

Given a language $\mathcal{L} = \mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow,\neg\}}(\mathbf{P}, \mathbf{F}, \mathbf{C})$, a formula A of $\mathcal{L}$, and a structure $\mathbf{M} = [U, I]$ for $\mathcal{L}$.

The structure $\mathbf{M} = [U, I]$ is a **counter model** *for the formula A if and only if there is an assignment $s : VAR \longrightarrow U$, such that $(\mathbf{M}, s) \not\models A$.*

We denote it as $\mathbf{M} \not\models A$.

For any set $\Gamma \subseteq \mathcal{F}$ of formulas of $\mathcal{L}$, $\mathbf{M}$ is a **counter model** *for Γ if and only if there is $A \in \Gamma$, such that $\mathbf{M} \not\models A$.*

We denote it as $\mathbf{M} \not\models \Gamma$.

Observe that if A is a **sentence** (definition 8.4) then the truth or falsity of $(\mathbf{M}, s) \models A$ is completely independent of s. Hence if $(\mathbf{M}, s) \models A$ for some s, it holds for all s and the following holds.

Fact 8.1

For any formula A of $\mathcal{L}$,

If A is a sentence, then if there s such that $(\mathbf{M}, s) \models A$, then $\mathbf{M}$ is a model for A, i.e. $\mathbf{M} \models A$

We transform any formula A of $\mathcal{L}$ into a certain sentence by binding all its free variables. The resulting sentence is called a closure of A and is defined as follows.

Definition 8.17 (Closure)

Given a formula A of $\mathcal{L}$.

By the **closure** *of A we mean the formula obtained from A by prefixing in universal quantifiers all variables the are free in A. If A does not have free variables (i.e. is a sentence), the closure if A is defined to be A itself.*

Obviously, a closure of any formula is always a sentence. For example, if A, B are formulas

$$(P(x_1, x_2) \Rightarrow \neg\exists x_2\ Q(x_1, x_2, x_3)), \quad (\forall x_1 P(x_1, x_2) \Rightarrow \neg\exists x_2\ Q(x_1, x_2, x_3)),$$

their respective **closures** are

$$\forall x_1 \forall x_2 \forall x_3\ ((P(x_1, x_2) \Rightarrow \neg\exists x_2\ Q(x_1, x_2, x_3))),$$

$$\forall x_1 \forall x_2 \forall x_3\ ((\forall x_1 P(x_1, x_2) \Rightarrow \neg\exists x_2\ Q(x_1, x_2, x_3))).$$

Example 8.16

Let $Q \in \mathbf{P},\ \#Q = 2$ *and* $c \in \mathbf{C}$ *Consider formulas*

$$Q(x, c), \quad \exists x Q(x, c), \quad \forall x Q(x, c)$$

and the structures from exercise 8.5 defined as follows.

$$\mathbf{M}_1 = [\{1\},\ Q_I :=,\ c_I : 1], \qquad \mathbf{M}_2 = [\{1, 2\},\ Q_I :\leq,\ c_I : 1],$$

$$\mathbf{M}_3 = [N,\ Q_I :\geq,\ c_I : 0], \quad \textit{and} \quad \mathbf{M}_4 = [N,\ Q_I :\geq,\ c_I : 1.]$$

Directly from example 8.15 and Fact 8.1, we get that:

1. $\mathbf{M}_1 \models Q(x, c), \quad \mathbf{M}_1 \models \forall x Q(x, c), \quad \mathbf{M}_1 \models \exists x Q(x, c).$

2. $\mathbf{M}_2 \not\models Q(x, c), \quad \mathbf{M}_2 \not\models \forall x Q(x, c), \quad \mathbf{M}_2 \models \exists x Q(x, c).$

3. $\mathbf{M}_3 \models Q(x, c), \quad \mathbf{M}_3 \models \forall x Q(x, c), \quad \mathbf{M}_3 \models \exists x Q(x, c).$

4. $\mathbf{M}_4 \not\models Q(x, c), \quad \mathbf{M}_4 \not\models \forall x Q(x, c), \quad \mathbf{M}_4 \models \exists x Q(x, c).$

Definition 8.18 (True, False in M)

Given a structure $\mathbf{M} = [U, I]$ *for* $\mathcal{L}$ *and a formula A of* $\mathcal{L}$*. We say that:*

A is **true** *in* $\mathbf{M}$ *(written as* $\mathbf{M} \models A$*) if and only if all assignments s of* $\mathcal{L}$ *in* $\mathbf{M}$ *satisfy A, i.e. when* $\mathbf{M}$ *is a* **model** *for A.*

A is **false** *in* $\mathbf{M}$ *(written as* $\mathbf{M} =\mid A$*) if and only if no assignment s of* $\mathcal{L}$ *in* $\mathbf{M}$ *satisfies A.*

By the definition 9.3 we have that A is **true** in $\mathbf{M}$ only when the structure $\mathbf{M}$ is a **model** for A. This is why we use the notation $\mathbf{M} \models A$ in both cases.

Obviously, if A is not true in $\mathbf{M}$, then it is false, and vice versa. This proves correctness of our definition with respect to the intuitive understanding.

We get directly from definition 8.18 and the example 8.16 the following.

Example 8.17

Let $\mathbf{M}_1 - \mathbf{M}_4$ be structures defined in example 8.5.

1. Formulas $Q(x,c)$, $\forall x Q(x,c)$, $\exists x Q(x,c)$ are all **true** *in the structures $\mathbf{M}_1$ and $\mathbf{M}_3$.*

2. Formula $\exists x Q(x,c)$ is also **true** *in $\mathbf{M}_2$ and in $\mathbf{M}_3$.*

3. Formulas $\neg Q(x,c), \neg\forall x Q(x,c), \neg\exists x Q(x,c)$ are all **false** *in the structures $\mathbf{M}_1$ and $\mathbf{M}_3$.*

4. Formula $\neg\exists x Q(x,c)$ is also **false** *in $\mathbf{M}_2$ and in $\mathbf{M}_3$.*

5. Formulas $(Q(x,c) \cap \neg Q(x,c))$, $(\neg\forall x Q(x,c) \cap \forall x Q(x,c))$, and the formula $(\exists x Q(x,c) \cap \neg\exists x Q(x,c))$ are all **false** *in all structures $\mathbf{M}_1 - \mathbf{M}_4$.*

6. The formula $\forall x Q(x,c)$ is false in a structure $\mathbf{M}_5 = [N,\ Q_I :<,\ c_I : 0]$.

Here are some properties of the notions "A is true in $\mathbf{M}$, written symbolically as $\mathbf{M} \models A$, and "A is false in $\mathbf{M}$, written symbolically as $\mathbf{M} =\mid A$. They are obvious under intuitive understanding of the notion of satisfaction. Their formal proofs are left as exercise for the reader.

Property 8.1 (Truth, Falsity, Satisfaction)

Given a structure $\mathbf{M} = [U, I]$ for $\mathcal{L}$ and any formulas formula A, B of $\mathcal{L}$. The following properties hold.

P1. A is false in $\mathbf{M}$ if and only if $\neg A$ is true in $\mathbf{M}$, i.e.

$$\mathbf{M} =\mid A \ \textit{ if and only if } \ \mathbf{M} \models \neg A.$$

P2. A is true in $\mathbf{M}$ if and only if $\neg A$ is false in $\mathbf{M}$, i.e.

$$\mathbf{M} \models A \ \textit{ if and only if } \ \mathbf{M} =\mid \neg A.$$

P3. It is not the case that both $\mathbf{M} \models A$ and $\mathbf{M} \models \neg A$, i.e. no formula of $\mathcal{L}$ can be both true and false in $\mathbf{M}$, i.e. there is no formula A, such that $\mathbf{M} \models A$ and $\mathbf{M} =\mid A$.

P4. If $\mathbf{M} \models A$ and $\mathbf{M} \models (A \Rightarrow B)$, then $\mathbf{M} \models B$.

P5. $(A \Rightarrow B)$ is false in $\mathbf{M}$ if and only if $\mathbf{M} \models A$ and $\mathbf{M} \models \neg B$, i.e.

$$\mathbf{M} =\mid (A \Rightarrow B) \ \textit{ if and only if } \ \mathbf{M} \models A \textit{ and } \ \mathbf{M} \models \neg B.$$

P6. $\mathbf{M} \models A$ if and only if $\mathbf{M} \models \forall x A$.

P7. A formula A its true in $\mathbf{M}$ if and only if its closure (definition 8.17) is true in $\mathbf{M}$.

Definition 8.19 (Valid, Tautology)

Given a language $\mathcal{L} = \mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow,\neg\}}(\mathbf{P},\mathbf{F},\mathbf{C})$, a formula A of $\mathcal{L}$. .

A formula A is **predicate tautology** *(is* **valid***) if and only if $\mathbf{M} \models A$ for all structures $\mathbf{M} = [U, I]$, i.e. when A is* **true** *in all structures $\mathbf{M}$ for $\mathcal{L}$.*

We write

$$\models A \quad or \quad \models_p A,$$

to denote that a formula A is **predicate tautology** *(is* **valid***).*

We write

$$\models_p A$$

when there is a need to stress a **distinction** between propositional and predicate tautologies, otherwise we will use the symbol $\models$.

Predicate tautologies are also called **laws of quantifiers**.

Following the notation $\mathbf{T}$ for the set of all propositional tautologies (chapter 5) we denote by $\mathbf{T}_p$ the set of all predicate tautologies, i.e.

$$\mathbf{T}_p = \{A \text{ of } \mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow,\neg\}}(\mathbf{P},\mathbf{F},\mathbf{C}) : \models_p A\}. \tag{8.13}$$

Directly from the definition 8.18, the tautology definition 8.19 we get the following basic properties of logical consequence as defined by definition 8.15.

Property 8.2

For any $A, B \in \mathcal{F}$ and any set $\Gamma \subseteq \mathcal{F}$ of formulas of $\mathcal{L}$,

P1. $A \models B$ if and only if $\models (A \Rightarrow B)$.

P2. If $A \models B$ and A is true in $\mathbf{M}$, then B is true in $\mathbf{M}$.

P2. If $\Gamma \models B$ and if all formulas in Γ are true in $\mathbf{M}$, then B is true in $\mathbf{M}$.

We get immediately from the above definition 8.19 of a following definition of a notion *" A is not a predicate tautology"*.

Definition 8.20

For any formula A of predicate language $\mathcal{L}$,
A **is not** *a predicate tautology ($\not\models A$) if and only if there is a structure $\mathbf{M} = (U, I)$ for $\mathcal{L}$, such that $\mathbf{M} \not\models A$.*
We call such structure $\mathbf{M}$ a **counter-model** *for A.*

The definition 8.20 says: to prove that a formula A is not a predicate tautology one has to show a *counter- model* $\mathbf{M} = (U, I)$. It means one has to show a non-empty set U, define an interpretation I, and an assignment $s : VAR \longrightarrow U$

such that $(\mathbf{M}, s) \not\models A$.

We introduce, similarly as in a case of propositional semantic a notion of predicate contradiction.

Definition 8.21 (Contradiction)

For any formula A of predicate a language $\mathcal{L}$,

A is a **predicate contradiction** *if and only if A is* **false** *in all structures* $\mathbf{M}$.

We denote it as $=\!| \ A$ and write symbolically

$$=\!|\ A \quad \textit{if and only if} \quad \mathbf{M} =\!|\ A, \ \textit{for all structures } \mathbf{M}.$$

When there is a need to distinguish between propositional and predicate contradictions we also use symbol

$$=\!|_p\ A,$$

where "p" stands for "predicate".

Following the notation $\mathbf{C}$ for the set of all propositional tautologies (chapter 5) we denote by $\mathbf{C}_p$ the set of all predicate contradictions, i.e.

$$\mathbf{C}_p = \{A \ \text{ of } \ \mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow,\neg\}}(\mathbf{P}, \mathbf{F}, \mathbf{C}) : \ =\!|_p\ A\}. \tag{8.14}$$

Directly from the definition 8.18 and Property 8.1 we have the folowing duality property, the same as the one for propositional logic.

Fact 8.2

For any formula A of predicate a language $\mathcal{L}$,

$$A \in \mathbf{T}_p \quad \textit{if and only if} \quad \neg A \in \mathbf{C}_p,$$

$$A \in \mathbf{C}_p \quad \textit{if and only if} \quad \neg A \in \mathbf{T}_p.$$

Obviously, the formulas $(Q(x,c) \cap \neg Q(x,c))$, $(\neg\forall x Q(x,c) \cap \forall x Q(x,c))$, and the formula $(\exists x Q(x,c) \cap \neg\exists x Q(x,c))$ defined in example 8.17 are not only false in the structures $\mathbf{M}_1 - \mathbf{M}_4$, but are false in all structures $\mathbf{M}$ for $\mathcal{L}$. By definition 8.21 they all are predicate contradictions. Observe that they all are substitutions of propositional contradictions $(a \cap \neg a)$ or $(\neg a \cap a)$. By the same argument the formulas $(Q(x,c) \cup \neg Q(x,c))$, $(\neg\forall x Q(x,c) \cup \forall x Q(x,c))$, $(\exists x Q(x,c) \cap \neg\exists x Q(x,c))$ are predicate tautologies as they are substitutions of propositional tautologies $(a \cup \neg a)$ or $(\neg a \cup a)$.

We put these examples and observations in a following theorems that establish relationship between propositional and predicate tautologies and contradictions.

We write now $\models$, $=\mid$ do denote respectively propositional tautologies and contradiction, and $\models_p$, $=\mid_p$ for predicate tautologies and contradictions. We first formalize and prove (theorem 8.1) the intuitively obvious fact: *if a formula A is a propositional tautology (contradiction), then replacing propositional variables in A by any formulas of a predicate language we obtain a formula which is a predicate tautology (contradiction).*

Example 8.18

Let consider the following example of a propositional tautology and a propositional contradiction.

$$\models ((a \Rightarrow b) \Rightarrow (\neg a \cup b)) \quad \textit{and} \quad =\mid ((a \cup \neg a) \Rightarrow (\neg b \cap b)).$$

Substituting $\exists x P(x,z)$ for a, and $\forall y R(y,z)$ for b, we obtain, by theorem 8.1, that

$$\models_p \; ((\exists x P(x,z) \Rightarrow \forall y R(y,z)) \Rightarrow (\neg \exists x P(x,z) \cup \forall y R(y,z))) \quad \textit{and}$$

$$=\mid ((\exists x P(x,z) \cup \neg \exists x P(x,z)) \Rightarrow (\neg \forall y R(y,z) \cap \forall y R(y,z))).$$

We put it all in a more formal and more general and precise language as follows.

Given a propositional language $\mathcal{L}_0 = \mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow,\neg\}}$ with the set $\mathcal{F}_0$ of formulas and a predicate language$\mathcal{L} = \mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow,\neg\}}(\mathbf{P},\mathbf{F},\mathbf{C})$ with the set $\mathcal{F}$ of formulas. Let $A(a_1, a_2, \dots, a_n) \in \mathcal{F}_0$ and $A_1,\ A_2,\ \dots,\ A_n \in \mathcal{F}$. We denote by

$$A(a_1/A_1,\ a_2/A_2,\ \dots,\ a_n/A_n) \qquad (8.15)$$

the result of replacing in A the free variables $a_1, a_2, \dots,\ a_n$ by the formulas $A_1,\ A_2,\ \dots,\ A_n \in \mathcal{F}$. Of course $A(a_1/A_1,\ a_2/A_2,\ \dots,\ a_n/A_n) \in \mathcal{F}$.

Theorem 8.1

Given a propositional language $\mathcal{L}_0$ with the set $\mathcal{F}_0$ of formulas and a predicate language $\mathcal{L}$ with the set $\mathcal{F}$ of formulas.

For any $A(a_1, a_2, \dots a_n) \in \mathcal{F}_0$ and any $A_1, A_2, \dots A_n \in \mathcal{F}$ the following holds.

1. If $\models\ A(a_1, a_2, \dots, a_n)$, then $\models_p\ A(a_1/A_1,\ a_2/A_2,\ \dots,\ a_n/A_n)$.

2. If $=\mid A(a_1, a_2, \dots, a_n)$, then $=\mid_p A(a_1/A_1,\ a_2/A_2,\ \dots,\ a_n/A_n)$.

Proof 1. follows directly from satisfaction definition 8.13. 2. follows from definition 8.13, property 8.1, and definition 8.21.

Some predicate tautologies are, by theorem 8.1, substitutions of propositional formulas. Visibly a predicate formula $(\forall x\ A(x) \Rightarrow \exists x\ A(x))$ can not be obtained as a substitution of a propositional formula. We prove now that it is a predicate tautology.

Fact 8.3

For any formula $A(x)$ of $\mathcal{L}$,

$$\models\ (\forall x\ A(x) \Rightarrow \exists x\ A(x)).$$

Proof
Assume that $\not\models (\forall x\ A(x) \Rightarrow \exists x\ A(x))$. By definition 8.20 there is a structure $\mathbf{M} = (U, I)$ and $s : VAR \longrightarrow U$, such that $(\mathbf{M}, s) \not\models (\forall x\ A(x) \Rightarrow \exists x\ A(x))$. By definition 8.13, $(\mathbf{M}, s) \models \forall x\ A(x)$ and $(\mathbf{M}, s) \not\models \exists x\ A(x)$. It means that $(\mathbf{M}, s') \models A(x)$ for all s' such that s, s' **agree** on all variables except on x, and it is not true that there is s' such that s, s' **agree** on all variables except on x, and $(\mathbf{M}, s') \models A(x)$. This is impossible and this contradiction proves $\models\ (\forall x\ A(x) \Rightarrow \exists x\ A(x))$.

Given a set $\mathcal{F}$ of formulas of a predicate language $\mathcal{L}$. We denote by $O\mathcal{F}$ set of all open formulas of $\mathcal{L}$, i.e. formulas without quantifiers. We prove that any open formula in order to be predicate tautologies must be a substitution definied in theorem 8.1 of a propositional tautology. I.e. we have the following substitution theorem.

Theorem 8.2

Any open formula A of a predicate language $\mathcal{L}$ is a predicate tautology if and only if it is a substitution of a propositional tautology as defined in theorem 8.1.

Proof
Observe that every open formula from $A \in O\mathcal{F}$ is a form
$B(a_1/A_1,\ a_2/A_2,\ \ldots,\ a_n/A_n)$ for certain propositional formula $B(a_1, a_2, \ldots a_n)$, where $A_1,\ A_2,\ \ldots,\ A_n$ are predicate atomic formulas from the set $A\mathcal{F}$ as defined in (8.5). Theorem 8.2 follows directly from the following.

Lemma 8.1

Let σ be a one to one mapping from the set V_0 of propositional variables of propositional language $\mathcal{L}_0$ into the set $A\mathcal{F}$ of the atomic formulas of the predicate language $\mathcal{L}$. For any $A(a_1, a_2, \ldots a_n) \in \mathcal{F}_0$,

$$\models\ A(a_1, a_2, \ldots, a_n),\ \textit{if and only if}\ \models_p\ A(a_1/\sigma(a_1),\ \ldots,\ a_n/\sigma(a_n)).$$

Proof of lemma
The implication "if $\models A(a_1, a_2, \ldots, a_n)$, then
$\models_p\ A(a_1/\sigma(a_1),\ \ldots,\ a_n/\sigma(a_n))$" holds as a particular case of theorem 8.2. We prove now the converse implication by proving its opposite

$$\text{if } \not\models A(a_1, a_2, \ldots, a_n),\ \text{then } \not\models_p\ A(a_1/\sigma(a_1), \ldots,\ a_n/\sigma(a_n)). \tag{8.16}$$

Assume $\not\models A(a_1, a_2, \ldots, a_n)$. There exists a truth assignment $v : V_0 \longrightarrow \{T, F\}$ such that $v^*(A(a_1, a_2, \ldots, a_n)) = F$. We construct a counter model $\mathbf{M}$ for

$A(a_1/\sigma(a_1), \dots,\ a_n/\sigma(a_n))$ as follows. Let $\mathbf{M} = [\mathbf{T}, I]$, where $\mathbf{T}$ is the set of all terms of $\mathcal{L}$, and for any $c \in \mathbf{C},\ f \in \mathbf{F},\ P \in \mathbf{P}$ we put $c_I = c,\ f_I(t_1, t_2, \dots, t_n) = f(t_1, t_2, \dots t_n),\ P_I \subseteq \mathbf{T}^{\#P}$.

Let now the s assignment of $\mathcal{L}$ in $\mathbf{M}$ be an identity, i.e. $s : VAR \longrightarrow \mathbf{T}$ is such that $s(x) = x$ for all $x \in VAR$. We extend s to the interpretation of terms (definition 8.12) as follows.

$s_I(x) = s(x) = x,\ s_I(c) = c_I = c,\ s_I(f(t_1, t_2, \dots t_n)) = f_I(s_I(t_1),\ \dots,\ s_I(t_n)) = f(t_1, t_2, \dots t_n)$, i.e. we have that $s_I(t) = t$ for all $t \in \mathbf{T}$.
We have that for every atomic formula $P(t_1, t_2, \dots t_n)$ there is exactly one propositional variable a, such that $P(t_1, t_2, \dots t_n) = \sigma(a)$. We define now that P_I as follows.

$(t_1, t_2, \dots t_n) \in P_I$ if and only if $P(t_1, t_2, \dots t_n) = \sigma(a)$ and $v(a) = T$.

$(t_1, t_2, \dots t_n) \notin P_I$ if and only if $P(t_1, t_2, \dots t_n) = \sigma(a)$ and $v(a) = F$.

We assumed that $v : V_0 \longrightarrow \{T, F\}$ is such that $v^*(A(a_1, a_2, \dots, a_n)) = F$. Directly form definition of the assignment s and the interpretation I we have that $([\mathbf{T}, I], s) \not\models A(a_1/\sigma(a_1), \dots,\ a_n/\sigma(a_n))$. It end the roof of lemma 8.1 and hence the proof of theorem 8.2.

Fact 8.4

The converse implication to (8.3) is not a predicate tautology, i.e. there is a formula A of $\mathcal{L}$. such that

$$\not\models\ (\exists x\ A(x) \Rightarrow \forall x\ A(x)). \tag{8.17}$$

Proof
Observe that to prove (8.17) we have to provide an example of an instance of a formula $A(x)$ and construct a counter-model $\mathbf{M} = (U, I)$ for it. Let $A(x)$ be an atomic formula $P(x, c)$, for any $P \in \mathbf{P}, \#P = 2$. The instance is

$$(\exists x\ P(x, c) \Rightarrow \forall x\ P(x, c)).$$

We take as $\mathbf{M} = (N, P_I :<,\ c_I : 3)$ for N set of natural numbers. Let s be any assignment $s : VAR \longrightarrow N$. We show now $(\mathbf{M}, s) \models \exists x\ P(x, c)$. Take any s' such that $s'(x) = 2$ and $s'(y) = s(y)$ for all $y \in VAR - \{x\}$. We have $(2, 3) \in P_I$, as $2 < 3$ and hence there exists s' that agrees with s on all variables except on x, and $(\mathbf{M}, s') \models P(x, c)$. But $(\mathbf{M}, s) \not\models \forall x\ P(x, c)$ as for example for s' such that $s'(x) = 5$ and $s'(y) = s(y)$ for all $y \in VAR - \{x\}$, $(2, 3) \notin P_I$, as $5 \not< 3$. This proves that $\mathbf{M} = (N, P_I :<,\ c_I : 3)$ is a counter model for $\forall x\ P(x, c)$. Hence $\not\models\ (\exists x\ A(x) \Rightarrow \forall x\ A(x))$.

The "shorthand" solution is: the formula $(\exists x\ P(x, c) \Rightarrow \forall x\ P(x, c))$ becomes in $\mathbf{M} = (N, P_I :<,\ c_I : 3)$ a mathematical statement (written with logical symbols): $\exists n\ n < 3\ \Rightarrow\ \forall n\ n < 3$. It is an obviously *false* statement in the set

N of natural numbers, as there is $n \in N$, such that $n < 3$, for example $n = 2$, and it is not true that all natural numbers are smaller then 3.

We have to be very careful when we deal with **quantifiers with restricted domain** (definition 8.9). We adopt the following definition for restricted domain quantifiers.

Definition 8.22 (Restricted Quantifiers Tautology)

For any formulas $A(x), B(x) \in \mathcal{F}$ *with any free variable* $x \in VAR$, *and for the restricted domain quantifies* $\forall_{B(x)},\ \exists_{B(x)}$ *we define*

$$\models\ \forall_{B(x)}\ A(x) \quad \textit{if and only if} \quad \models\ \forall x\ (B(x) \Rightarrow A(x)),$$

$$\models \exists_{B(x)}\ A(x) \quad \textit{if and only if} \quad \models\ \exists x\ (B(x) \cap A(x)).$$

The most basic predicate tautology (8.3) fails when we use the quantifiers with restricted domain. We show now that

$$\not\models\ (\forall_{B(x)}\ A(x) \Rightarrow \exists_{B(x)}\ A(x)). \tag{8.18}$$

By definition 8.22 to prove (8.18) means to prove that corresponding proper formula of $\mathcal{L}$ obtained by the restricted quantifiers *transformations rules* (8.48), (8.49) is not a predicate tautology, i.e. to show that

$$\not\models\ (\forall x(B(x) \Rightarrow A(x)) \Rightarrow \exists x(B(x) \cap A(x))). \tag{8.19}$$

In order to prove (8.19) we have to provide an example of particular formulas $A(x), B(x)$ and to construct a counter model for these particular formulas. We take as $B(x), A(x)$ atomic formulas $Q(x, c), P(x, c)$ and we construct a counter model a corresponding formula

$$(\forall x(Q(x, c) \Rightarrow P(x, c)) \Rightarrow \exists x(Q(x, c) \cap P(x, c))) \tag{8.20}$$

as follows. We take $\mathbf{M} = (N, I)$, where N is the set of real numbers and the interpretation I is defined as $Q_I :<,\ P_I :>,\ c_I : 0$. The "shorthand" solution is as follows. The formula 8.19) becomes a mathematical statement

$$(\forall n\ (n < 0 \Rightarrow n > 0) \Rightarrow \exists_{n \in N}(n < 0 \cap n > 0)).$$

This statement is a *false* in the set N of natural numbers because the statement $n < 0$ is false for all natural numbers and $F \Rightarrow B$ is a true implication for any logical value of B, so $\forall n\ (n < 0 \Rightarrow n > 0)$ is a true statement and $\exists n\ (n < 0 \cap n > 0)$ is obviously false in the set N of natural numbers.

The restricted quantifiers law corresponding to the predicate tautology (8.3) is:

$$\models\ (\forall_{B(x)}\ A(x) \Rightarrow (\exists x\ B(x) \Rightarrow \exists_{B(x)}\ A(x))). \tag{8.21}$$

By definition 8.22 and restricted quantifiers *transformations rules* (8.7), (8.8) proving (8.19) is means proving

$$\models (\forall x(B(x) \Rightarrow A(x)) \Rightarrow (\exists x\ B(x) \Rightarrow \exists x\ (B(x) \cap A(x)))).$$

We leave the proof and an exercise for the reader.

8.3 Predicate Tautologies

We have already proved in Fact 8.3 the basic predicate tautology

$$\models\ (\forall x\ A(x) \Rightarrow \exists x\ A(x)).$$

We are going to prove now the following.

Fact 8.5 (Dictum de Omni)

For any formula $A(x)$ of $\mathcal{L}$,

$$\models\ (\forall x\ A(x) \Rightarrow A(t)), \qquad \models\ (\forall x\ A(x) \Rightarrow A(x)), \tag{8.22}$$

$$\models\ (A(t) \Rightarrow \exists x\ A(x)), \tag{8.23}$$

where t is a term, A(t) is a result of substitution of t for all free occurrences of x in $A(x)$, and t is free for x in A(x) (definition 8.7), i.e. no occurrence of a variable in t becomes a bound occurrence in A(t).

Proof of (8.22) is constructed in a sequence of steps. We leave details to the reader to complete as an exercise. Here are the steps.

S1 Consider a structure $\mathbf{M} = [U, I]$ and $s : VAR \longrightarrow U$. Let $t,\ u$ be two terms. Denote by t' a result of replacing in t all occurrences of a variable x by the term u, i.e. $t' = t(x/u)$. Let s' results from s by replacing $s(x)$ by $s_I(u)$. We prove by induction over the length of t that

$$s_I(t(x/u)) = s_I(t') = {s'}_I(t). \tag{8.24}$$

S2 Let t be free for x in $A(x)$. $A(t)$ is a results from $A(x)$ by replacing t for all free occurrences of x in $A(x)$, i.e. $A(t) = A(x/t)$. Let $s : VAR \longrightarrow U$ and s' be obtained from s by replacing $s(x)$ by $s_I(t)$. We use (8.24) and induction on the number of connectives and quantifiers in $A(x)$ and prove

$$(\mathbf{M}, s)\ \models A(x/t) \text{ if and only if } (\mathbf{M}, s')\ \models A(x). \tag{8.25}$$

S3 Directly from definition 8.13 and (8.25) we get that for any $\mathbf{M} = (U, I)$ and any $s : VAR \longrightarrow U$,

$$\text{if } (\mathbf{M}, s)\ \models\ \forall x A(x), \text{ then } (\mathbf{M}, s)\ \models\ A(t).$$

This proves that $(\forall x\ A(x) \Rightarrow A(t))$ is a predicate tautology. Observe that a term x is free for x in A(x), so we also get as a particular case of $t = x$ that $\models (\forall x\ A(x) \Rightarrow A(x))$.

Proof of (8.23) follows from (8.22), theorem 8.1, property 8.1, theorem 8.5, and definability law (8.49). We carry it as follows. First we observe that by theorem 8.1 we have that $\models ((\forall x \neg A(x) \Rightarrow \neg A(t)) \Rightarrow (A(t) \Rightarrow \neg\forall x \neg A(x)))$ as a substitution of propositional tautology $((a \Rightarrow \neg b) \Rightarrow (b \Rightarrow \neg a))$. By just proved (8.22) we have that $\models (\forall x \neg A(x) \Rightarrow \neg A(t))$ for $A(x)$ being a formula $\neg A(x)$. By $P4$ in property 8.1, we get $\models (A(t) \Rightarrow \neg \forall x \neg A(x))$. We apply the existential quantifier definability law (8.49) and equivalence substitution theorem 8.5 and get $\models\ (A(t) \Rightarrow \exists x\ A(x))$. This ends the proof of (8.22).

Remark the restrictions in (8.22) and (8.23) are essential. Here is a simple example explaining why they are needed in (8.22). The example for (8.23) is similar.

Let $A(x)$ be a formula $\neg\forall y\ P(x, y)$, for $P \in \mathbf{P}$. Notice that a term $t = y$ is *not free for y* in A(x). Consider (8.22) $A(x) = \neg\forall y\ P(x, y)$ and $t = y$.

$$(\forall x \neg\forall y\ P(x, y) \Rightarrow \neg\forall y\ P(y, y)), \tag{8.26}$$

Take $\mathbf{M} = [N, I]$ for I such that $P_I\ :=$. Obviously, $\mathbf{M} \models \forall x \neg \forall y\ P(x, y)$ as $\forall n \neg \forall n(n = n)$ is a true mathematical statement in the set N of natural numbers. $\mathbf{M} \not\models \neg\forall y\ P(y, y)$ as $\neg\forall n\ (n = n)$ is a false statement for $n \in N$. Hence $\mathbf{M}$ is a counter model for for (8.26) and we proved that without the restriction (8.22) does not hold.

Here are some useful and easy to prove properties of the notion "t free for x in $A(x)$" (definition 8.7).

Property 8.3 (t free for x in $A(x)$)

For any formula $A \in \mathcal{F}$ and any term $t \in \mathbf{T}$ the following properties hold.

P1. A closed tern t, i.e. term with no variables is free for any variable x in A.

P2. A term t is free for any variable in A if none of the variables in t is bound in A.

P3. Term $t = x$ is free for x in any formula A.

P4. Any term is free for x in A if A contains no free occurrences of x.

Here are some more important predicate tautologies.

Generalization

For any formulas $A(x), B(x), A, B$ of $\mathcal{L}$, where A, B does not contain any free occurrences of x,

$$\models\ ((B \Rightarrow A(x)) \Rightarrow (B \Rightarrow \forall x\ A(x))), \tag{8.27}$$

$$\models \; ((B(x) \Rightarrow A) \Rightarrow (\exists x B(x) \Rightarrow A)), \tag{8.28}$$

Distributivity 1

For any formulas $A(x), B(x), A, B$ of $\mathcal{L}$, such that A , B does not contain any free occurrences of x,

$$\models \; (\forall x(A \Rightarrow B(x)) \Rightarrow (A \Rightarrow \forall x \; B(x))), \tag{8.29}$$

$$\models \forall x(A(x) \Rightarrow B) \Rightarrow (\exists x A(x) \Rightarrow B) \tag{8.30}$$

$$\models \; \exists x(A(x) \Rightarrow B) \Rightarrow (\forall x A(x) \Rightarrow B) \tag{8.31}$$

The restrictions that the formulas A, B do not contain any free occurrences of x is essential for both *Generalization* and *Distributivity 1* tautologies.

Here is a simple example explaining why they are needed in (8.29). The relaxation of the assumption that A, B do not contain any free occurrences of x would lead to the following disaster. Let A and B be both atomic formula P(x). Thus x is free in A and we have the following instance of (8.29).

$$(\forall x(P(x) \Rightarrow P(x)) \Rightarrow (P(x) \Rightarrow \forall x \; P(x))).$$

Observe that $\forall x(P(x) \Rightarrow P(x))$ is a predicate tautology. Take $\mathbf{M} = [N, I]$ for I such that $P_I = ODD$, where $ODD \subseteq N$ is the set of odd numbers. Let $s : VAR \longrightarrow N$. By definition if I, $s_I(x) \in P_I$ if and only if $s_I(x) \in ODD$. Then obviously $(\mathbf{M}, s) \not\models \forall x \; P(x)$ and $\mathbf{M} = [N, I]$ is a counter model for (8.29) as $(\mathbf{M}, s) \models \forall x(P(x) \Rightarrow P(x))$.

The examples for (8.30), (8.31), and (8.29) similar.

Distributivity 2

For any formulas $A(x), B(x)$ of $\mathcal{L}$,

$$\models \; (\exists x \; (A(x) \cap B(x)) \; \Rightarrow \; (\exists x A(x) \cap \exists x B(x))), \tag{8.32}$$

$$\models \; ((\forall x A(x) \cup \forall x B(x)) \Rightarrow \forall x \; (A(x) \cup B(x))), \tag{8.33}$$

$$\models (\forall x(A(x) \Rightarrow B(x)) \Rightarrow (\forall x A(x) \Rightarrow \forall x B(x))) \tag{8.34}$$

The converse mplications to (8.32), (8.33), (8.34) are not a predicate tautologies , i.e. there are formulas $A(x), B(x)$, such that

$$\not\models \; ((\exists x A(x) \cap \exists x B(x)) \Rightarrow \exists x \; (A(x) \cap B(x))). \tag{8.35}$$

$$\not\models \; (\forall x \; (A(x) \cup B(x)) \Rightarrow (\forall x A(x) \cup \forall x B(x))), \tag{8.36}$$

$$\not\models ((\forall x A(x) \Rightarrow \forall x B(x)) \Rightarrow \forall x(A(x) \Rightarrow B(x))) \tag{8.37}$$

To prove (8.35), (8.36) we have to find particular formulas $A(x), B(x) \in \mathcal{F}$ and counter models $\mathbf{M} = [U, I]$ for these particular cases.

Consider (8.35). We take as $A(x), B(x)$ atomic formulas $Q(x,c), P(x,c)$. The particular case of (8.35) is now a formula

$$((\exists x P(x,c) \cap \exists x Q(x,c)) \Rightarrow \exists x\ (P(x,c) \cap Q(x,c))).$$

Take $\mathbf{M} = [R, I]$ where R is the set of real numbers, and the interpretation I is $Q_I :>,\ P_I :<,\ c_I : 0$. The particular case formula becomes an obviously *false* mathematical statement

$$((\exists_{x\in R}\ x > 0 \cap \exists_{x\in R}\ x < 0) \Rightarrow \exists_{x\in R}\ (x > 0 \cap x < 0)).$$

Consider (8.36). We take as Let $A(x), B(x)$ be atomic formulas $Q(x,c), R(x,c)$. The particular case of (8.36) is now a formula

$$(\forall x Q(x,c) \cup R(x,c)) \Rightarrow (\forall x Q(x,c) \cup \forall x R(x,c))).$$

Take $\mathbf{M} = (R, I)$ where R is the set of real numbers and $Q_I :\geq,\ R_I :<,\ c_I : 0$. The particular formula becomes an obviously *false* mathematical statement

$$(\forall_{x\in R}\ (x \geq 0 \cup x < 0) \Rightarrow (\forall_{x\in R}\ x \geq 0 \cup \forall_{x\in R}\ x < 0)).$$

De Morgan

For any formulas $A(x), B(x)$ of $\mathcal{L}$,

$$\models (\neg\forall x A(x) \Rightarrow \exists x \neg A(x)), \tag{8.38}$$

$$\models (\neg\exists x A(x) \Rightarrow \forall x \neg A(x)), \tag{8.39}$$

$$\models (\exists x \neg A(x) \Rightarrow \neg\forall x A(x)), \tag{8.40}$$

$$\models (\forall x \neg A(x)) \Rightarrow \neg\exists x A(x)). \tag{8.41}$$

We prove (8.38) as an example.The proofs of all other laws are similar. Assume that (8.38) does not hold. By definition 8.16 there is $\mathbf{M} = (U, I)$ and $s : VAR \longrightarrow U$, such that $(\mathbf{M}, s) \models \neg\forall x \neg A(x))$ and $(\mathbf{M}, s) \not\models \exists x \neg A(x)$.
Consider $(\mathbf{M}, s) \models \neg\forall x A(x)$. By satisfaction definition 8.13, $(\mathbf{M}, s) \not\models \forall x A(x)$. This holds only if for all s', such that s, s' agree on all variables except on x, $(\mathbf{M}, s') \not\models A(x)$.
Consider $(\mathbf{M}, s) \not\models \exists x \neg A(x)$. This holds only if there is no s', such that $(\mathbf{M}, s') \models \neg A(x)$, i.e. there is no s', such that $(\mathbf{M}, s') \not\models A(x)$. This means that for all s', $(\mathbf{M}, s') \models A(x)$. Contradiction with $(\mathbf{M}, s') \not\models A(x)$.

Quantifiers Alternations

For any formula $A(x,y)$ of $\mathcal{L}$,

$$\models (\exists x \forall y A(x,y) \Rightarrow \forall y \exists x A(x,y)). \tag{8.42}$$

The converse implications to (8.42) is not a predicate tautology. Take as $A(x, y)$ an atomic formulas $R(x, y)$. Take $\mathbf{M} = (R, I)$ where R is the set of real numbers and $R_I :<$. The instance of (8.42) particular formula becomes a mathematical statement

$$(\forall y \exists x(x < y) \Rightarrow \exists x \forall y(x < y))$$

that obviously *false* in the set of real numbers. We proved

$$\not\models (\forall y \exists x A(x, y) \Rightarrow \exists x \forall y A(x, y)). \tag{8.43}$$

8.3.1 Equational Laws of Quantifiers

The most frequently used laws of quantifiers have a form of a *logical equivalence*, symbolically written as $\equiv$. This not a new logical connective. This is a very useful symbol. It has the same properties as the equality $=$ and can be used in the same way we use the equality symbol $=$.

Note that we use the same equivalence symbol $\equiv$ and the tautology symbol $\models$ for propositional and predicate languages and semantics when there is no confusion. Formally we define the predicate equivalence as follows.

Definition 8.23 (Logical equivalence)

For any formulas $A,\ B \in \mathcal{F}$ of the **predicate language** *$\mathcal{L}$,*

$$A \equiv B \ \ \textit{if and only if} \ \models \ (A \Rightarrow B) \ \textit{and} \ \models (B \Rightarrow A).$$

Remark that our predicate language $\mathcal{L} = \mathcal{L}_{\{\neg, \cap, \cup, \Rightarrow, \neg\}}(\mathbf{P}, \mathbf{F}, \mathbf{C})$ we defined the semantics for (definition 8.13) does not include the equivalence connective $\Leftrightarrow$. If it does we extend the satisfaction definition 8.13 in a natural way and can adopt the following definition 8.24 of logical equivalence that is obviously equivalent definition to the propositional one and to our definition 8.23.

Definition 8.24

For any formulas $A, B \in \mathcal{F}$ of the **predicate language** *$\mathcal{L}$,*

$$A \equiv B \ \ \textit{if and only if} \ \models \ (A \Leftrightarrow B).$$

We re-write the basic theorem 8.1 establishing relationship between propositional and some predicate tautologies as follows.

Theorem 8.3 (Tautologies)

If a formula A is a propositional tautology, then by substituting for propositional variables in A any formula of the predicate language $\mathcal{L}$ we obtain a formula which is a predicate tautology.

Directly from the theorem 8.3 and logical equivalence definition 8.23 we get that the following is true.

Theorem 8.4 (Equivalences)

Given propositional formulas A, B.
If $A \equiv B$ is a propositional equivalence, and A', B' are formulas of the predicate language L obtained by a substitution of any formula of $\mathcal{L}$ for propositional variables in A and B, respectively, then $A' \equiv B'$ holds under predicate semantics.

Example 8.19

Consider the following propositional logical equivalence:

$$(a \Rightarrow b) \equiv (\neg a \cup b).$$

Substituting $\exists x P(x,z)$ for a, and $\forall y R(y,z)$ for b, we get from theorem 8.4 that the following equivalence holds:

$$(\exists x P(x,z) \Rightarrow \forall y R(y,z)) \equiv (\neg \exists x P(x,z) \cup \forall y R(y,z)).$$

We prove in similar way as in the propositional case (chapter 3) the following.

Theorem 8.5 (Equivalence Substitution)

Let a formula B_1 be obtained from a formula A_1 by a substitution of a formula B for one or more occurrences of a sub-formula A of A_1, what we denote as

$$B_1 = A_1(A/B).$$

Then the following holds for any formulas $A,\ A_1,\ B,\ B_1$ of $\mathcal{L}$.

$$\textit{If} \quad A \equiv B, \quad \textit{then} \quad A_1 \equiv B_1. \tag{8.44}$$

Directly from the Dictum de Omi (8.22) and the Generalization (**??**) tautologies we get the proof of the following theorem 8.6 useful for building new logical equivalences from the old, known ones.

Theorem 8.6

For any formulas $A(x), B(x)$ of $\mathcal{L}$.

$$\textit{if} \ \ A(x) \equiv B(x), \textit{then} \ \ \forall x A(x) \equiv \forall x B(x),$$

$$\textit{if} \ \ A(x) \equiv B(x), \textit{then} \ \ \exists x A(x) \equiv \exists x B(x).$$

Example 8.20

We know from the example 8.19 that the formulas $(\exists x P(x,z) \Rightarrow \forall y R(y,z))$ *and* $(\neg\exists x P(x,z) \cup \forall y R(y,z))$ *are logically equivalent. We get, as the direct consequence of the theorem 8.6 the following equivalences:*

$$\forall z(\exists x P(x,z) \Rightarrow \forall y R(y,z)) \equiv \forall z(\neg\exists x P(x,z) \cup \forall y R(y,z)),$$

Theorem 8.4 and theorem 8.6 allow us to use propositional tautologies and predicate formulas to build predicate equivalences. Here is a simple example.

Exercise 8.6

Prove that for any formulas $A(x),\ B(x)$ *of* $\mathcal{L}$

$$\neg\forall x\neg(A(x) \cup B(x)) \equiv \neg\forall x(\neg A(x) \cap \neg B(x)). \tag{8.45}$$

Solution
By the substituting $A(x)$ for a, and any formula $B(x)$ for b, in the propositional de Morgan Law: $\neg(a \cup b) \equiv (\neg a \cap \neg b)$, we get via theorem 8.4 that

$$\neg(A(x) \cup B(x)) \equiv (\neg A(x) \cap \neg B(x)).$$

Applying the theorem 8.6 to the above we obtain that

$$\forall x\neg(A(x) \cup B(x)) \equiv \forall x(\neg A(x) \cap \neg B(x)).$$

We know, from the propositional logic, that for any propositional variables a, b, $a \equiv b$ if and only if $\neg a \equiv \neg b$. Substituting $\forall x\neg(A(x) \cup B(x))$ and $\forall x(\neg A(x) \cap \neg B(x))$ for a and b, respectively, we get that

$$\forall x\neg(A(x) \cup B(x)) \equiv \forall x(\neg A(x) \cap \neg B(x))$$

if and only if

$$\neg\forall x\neg(A(x) \cup B(x)) \equiv \neg\forall x(\neg A(x) \cap \neg B).$$

But we have proved that $\forall x\neg(A(x) \cup B) \equiv \forall x(\neg A(x) \cap \neg B)$ holds, so we conclude that the equivalence (8.45) also holds.

Exercise 8.7

Prove that for any formulas $A(x),\ B$ *of* $\mathcal{L}$

$$\forall x\neg(A(x) \cup B) \equiv \forall x(\neg A(x) \cap \neg B)$$

Solution
By the substituting $A(x)$ for a, and any formula B for b, in the propositional de Morgan law: $\neg(a \cup b) \equiv (\neg a \cap \neg b)$, we get that

$$\neg(A(x) \cup B) \equiv (\neg A(x) \cap \neg B).$$

Applying the theorem 8.4 to the above we obtain that

$$\forall x\neg(A(x) \cup B) \equiv \forall x(\neg A(x) \cap \neg B).$$

As we can see, it is possible to obtain a fair amount of predicate tautologies from the propositional tautologies and theorems 8.3, 8.4 and 8.6, but as we have proved will for never obtain for example the most basic law: $(\forall x A(x) \Rightarrow \exists x A(x))$, any many the most important others.

We concentrate now only on these laws which have a form of a logical equivalence.They are called the **equational laws for quantifiers**.

Directly from the definition 8.23 and the de Morgan tautologies (8.38)-(8.41) we get one of the most important equational laws, called also De Morgan Laws.

De Morgan Laws

$$\neg\forall x A(x) \equiv \exists x\neg A(x) \tag{8.46}$$

$$\neg\exists x A(x) \equiv \forall x\neg A(x) \tag{8.47}$$

Now we will apply them to show that the quantifiers can be defined one by the other i.e. that the following Definability Laws hold.

Definability Laws

$$\forall x A(x) \equiv \neg\exists x\neg A(x) \tag{8.48}$$

$$\exists x A(x) \equiv \neg\forall x\neg A(x) \tag{8.49}$$

The law (8.48) is often used as a definition of the universal quantifier in terms of the existential one (and negation), the law (8.49) as a **definition** of the existential quantifier in terms of the universal one (and negation).

Proof of (8.48)
Substituting any formula $A(x)$ for a variable a in the propositional equivalence $a \equiv \neg\neg a$ we get by theorem 8.4 that $A(x) \equiv \neg\neg A(x)$. Applying the theorem 8.6 to the above we obtain $\exists x A(x) \equiv \exists x\neg\neg A(x)$. By the de Morgan Law (8.46) $\exists x\neg\neg A(x) \equiv \neg\forall x\neg A(x)$ and hence $\exists x A(x) \equiv \neg\forall x\neg A(x)$, what ends the proof.

Proof of (8.49)
We obtain $\forall x A(x) \equiv \forall\neg\neg A(x)$ in a similar way as above. By the de Morgan Law (8.47), $\forall\neg\neg A(x) \equiv \neg\exists\neg A(x)$ and hence $\forall x A(x) \equiv \neg\exists\neg A(x)$, what ends the proof.

Other important equational laws are the following introduction and elimination laws. We prove later the first two of them. We show that the laws (11.44) - (10.39) can be deduced from laws (10.32) and (11.43), the de Morgan laws (8.46), (8.47), definability laws (8.48), (8.49), propositional tautologies and theorems 8.3, 8.4, and theorem 8.5.

Introduction and Elimination Laws

If B is a formula such that B **does not contain any free occurrence** of x, then the following logical equivalences hold.

$$\forall x(A(x) \cup B) \equiv (\forall x A(x) \cup B) \tag{8.50}$$

$$\forall x(A(x) \cap B) \equiv (\forall x A(x) \cap B) \tag{8.51}$$

$$\exists x(A(x) \cup B) \equiv (\exists x A(x) \cup B) \tag{8.52}$$

$$\exists x(A(x) \cap B) \equiv (\exists x A(x) \cap B) \tag{8.53}$$

$$\forall x(A(x) \Rightarrow B) \equiv (\exists x A(x) \Rightarrow B) \tag{8.54}$$

$$\exists x(A(x) \Rightarrow B) \equiv (\forall x A(x) \Rightarrow B) \tag{8.55}$$

$$\forall x(B \Rightarrow A(x)) \equiv (B \Rightarrow \forall x A(x)) \tag{8.56}$$

$$\exists x(B \Rightarrow A(x)) \equiv (B \Rightarrow \exists x A(x)) \tag{8.57}$$

The equivalences (10.32)-(11.45) make it possible to introduce a quantifier that precedes a disjunction or a conjunction into one component on the condition that the other component does not contain any free occurrence of a variable which is bound by that quantifier. These equivalences also make possible to eliminate a quantifier from a component of a disjunction or a conjunction and to place it before that disjunction or conjunction as a whole, on the condition that the other component does not contain any free occurrence of a variable which that quantifier would then bind.

The equivalences (11.46)-(10.39)make it possible to introduce a quantifier preceding an implication into the consequent of that implication, on the condition that that antecedent does not contain any free occurrence of a variable which is bound by that quantifier; they also make it possible to introduce a universal quantifier preceding an implication into the consequent of that implication while changing it into an existential quantifier in the process,on the condition that the consequent of that implication does not contain any free occurrence of a variable bound by that quantifier. Equivalences (11.46)-(10.39) further enable us to eliminate quantifiers from the antecedent of an implication to the position preceding the whole implication, while changing a universal quantifier into an existential one, and vice versa, in the process, and also to eliminate quantifiers from the consequent of an implication to the position preceding the whole implication; the conditions that the other component of the implication in question does not contain any free occurrence of a variable which that quantifier would then bind, must be satisfied, respectively.

As we said before, the equivalences (10.32)-(10.39) are not independent, some of them are the consequences of the others. Assuming that we have already proved (10.32) and (11.43), the proof of (11.44) is as follows.

Proof of (11.44) $\exists x(A(x) \cup B)$ is logically equivalent, by the definability law (8.49) to $\neg\forall x\neg(A(x) \cup B)$. By the reasoning presented in the proof of (8.45) for B instead of $B(x)$, we have that $\neg\forall x\neg(A(x) \cup B) \equiv \neg\forall x(\neg A(x) \cap \neg B)$. By the introduction law (11.43), $\neg\forall x(\neg A(x) \cap \neg B) \equiv \neg(\forall x\neg A(x) \cap \neg B)$. Substituting $\forall x\neg A(x)$ for a and $\neg B$ for b in propositional equivalence $\neg(a \cap \neg b) \equiv (\neg a \cup \neg\neg b)$, we get, by the theorem 8.4 that $\neg(\forall x\neg A(x) \cap \neg B) \equiv (\neg\forall x\neg A(x) \cup \neg\neg B)$. In a similar way we prove that $\neg\neg B \equiv B$, by the definability law (8.49) $\neg\forall x\neg A(x) \equiv \exists x A(x)$, hence by theorem 8.5 $\neg\forall x\neg A(x) \cup \neg\neg B \equiv (\exists x A(x) \cup B)$ and finally, $\exists x(A(x) \cup B) \equiv (\exists x A(x) \cup B)$, what end the proof.

We can write this proof in a shorter, symbolic way as follows:

$$\begin{aligned}
\exists x(A(x) \cup B) \quad &\overset{\text{law 8.49}}{\equiv} \quad \neg\forall x\neg(A(x) \cup B) \\
&\overset{\text{thm 8.3, 8.4}}{\equiv} \quad \neg\forall x(\neg A(x) \cap \neg B) \\
&\overset{\text{law 11.43}}{\equiv} \quad \neg(\forall x\neg A(x) \cap \neg B) \\
&\overset{\text{(8.46), thm 8.5}}{\equiv} \quad (\neg\forall x\neg A(x) \cup \neg\neg B) \\
&\overset{\text{thm 8.5}}{\equiv} \quad (\exists x A(x) \cup B)
\end{aligned}$$

Distributivity Laws

Let $A(x), B(x)$ be any formulas with a free variable x.
Law of distributivity of **universal quantifier** over **conjunction**

$$\forall x \; (A(x) \cap B(x)) \;\equiv\; (\forall x A(x) \cap \forall x B(x)) \tag{8.58}$$

Law of distributivity of **existential quantifier** over **disjunction**.

$$\exists x \; (A(x) \cup B(x)) \;\equiv\; (\exists x A(x) \cup \exists x B(x)) \tag{8.59}$$

Alternations of Quantifiers Laws

Let $A(x, y)$ be any formula with a free variables x,y.

$$\forall x\forall y \; (A(x, y) \;\equiv\; \forall y\forall x \; (A(x, y) \tag{8.60}$$

$$\exists x\exists y \; (A(x, y) \;\equiv\; \exists y\exists x \; (A(x, y) \tag{8.61}$$

Renaming the Variables

Let $A(x)$ be any formula with a free variable x and let y be a variable that **does not occur** in A(x).
Let $A(x/y)$ be a result of **replacement** of each occurrence of x by y, then the following holds.

$$\forall x A(x) \equiv \forall y A(y), \tag{8.62}$$

$$\exists x A(x) \equiv \exists y A(y). \tag{8.63}$$

Restricted De Morgan Laws

For any formulas $A(x), B(x) \in \mathcal{F}$ with a free variable x,

$$\neg\forall_{B(x)} \; A(x) \equiv \exists_{B(x)} \; \neg A(x), \qquad \neg\exists_{B(x)} \; A(x) \equiv \forall_{B(x)}\neg A(x). \tag{8.64}$$

Here is a poof of first equality. The proof of the second one is similar and is left as an exercise.

$$\neg\forall_{B(x)}\, A(x) \equiv \neg\forall x\, (B(x) \Rightarrow A(x)) \equiv \neg\forall x\, (\neg B(x) \cup A(x)) \equiv \exists x\, \neg(\neg B(x) \cup A(x))$$
$$\equiv \exists x\ (\neg\neg B(x) \cap \neg A(x)) \equiv \exists x\ (B(x) \cap \neg A(x)) \equiv \exists_{B(x)}\, \neg A(x)).$$

Restricted Introduction and Elimination Laws

If B is a formula such that B *does not contain any free occurrence* of x, then the following logical equivalences hold for any formulas $A(x), B(x), C(x)$.

$$\forall_{C(x)}(A(x) \cup B) \equiv (\forall_{C(x)} A(x) \cup B), \tag{8.65}$$
$$\exists_{C(x)}\ (A(x) \cap B) \equiv (\exists_{C(x)}\ A(x) \cap B), \tag{8.66}$$
$$\forall_{C(x)}(A(x) \Rightarrow B) \equiv (\exists_{C(x)} A(x) \Rightarrow B), \tag{8.67}$$
$$\forall_{C(x)}(B \Rightarrow A(x)) \equiv (B \Rightarrow \forall_{C(x)} A(x)). \tag{8.68}$$

The proofs are similar to the proof of the restricted de Morgan Laws.
The similar generalization of the other *Introduction and Elimination Laws* (11.43), (11.44), (11.47), (10.39) for restricted domain quantifiers fails. We can easily follow the proof of (8.18) and construct proper counter-models proving the following.

$$\exists_{C(x)}(A(x) \cup B) \not\equiv (\exists_{C(x)} A(x) \cup B),$$
$$\forall_{C(x)}(A(x) \cap B) \not\equiv (\forall_{C(x)} A(x) \cap B),$$
$$\exists_{C(x)}(A(x) \Rightarrow B) \not\equiv (\forall_{C(x)} A(x) \Rightarrow B),$$
$$\exists_{C(x)}(B \Rightarrow A(x)) \not\equiv (B \Rightarrow \exists x A(x)).$$

Nevertheless it is possible to correctly generalize them all as to cover quantifiers with restricted domain. We show it in a case of (11.43) and leave the other cases to the reader as an exercise.

Example 8.21

The restricted quantifiers version of (11.43) is the following.

$$\exists_{C(x)}(A(x) \cup B) \equiv (\exists_{C(x)} A(x) \cup (\exists x\ C(x) \cap B)). \tag{8.69}$$

We derive (8.74) as follows.

$$\exists_{C(x)}(A(x) \cup B) \equiv \exists x (C(x) \cap (A(x) \cup B)) \equiv \exists x ((C(x) \cap A(x)) \cup (C(x) \cap B))$$
$$\equiv (\exists x (C(x) \cap A(x)) \cup \exists x (C(x) \cap B)) \equiv (\exists_{C(x)} A(x) \cup (\exists x\ C(x) \cap B)).$$

We leave it as an exercise to specify and write references to transformation or equational laws used at each step of our computation.

8.4 Hilbert Proof Systems Soundness and Completeness

We adopt now general definition from chapter 4 concerning proof systems to the case of classical first order (predicate) logic.

We refer the reader to chapters 4 and 5 for a great array of example, exercises, homework problems explaining in a great detail all notions we introduce here for the predicate case. The examples and exercises we provide here are not numerous and restricted to the laws of quantifiers.

Given a language $\mathcal{L} = \mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow,\neg\}}(\mathbf{P}, \mathbf{F}, \mathbf{C})$. Any proof system

$$S = (\mathcal{L} = \mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow,\neg\}}(\mathbf{P}, \mathbf{F}, \mathbf{C}), \quad \mathcal{F}, \quad LA, \quad \mathcal{R}) \tag{8.70}$$

is a **predicate (first order)** proof system.

The predicate proof system (11.1) is a **Hilbert** proof system if the set $\mathcal{R}$ of its rules contains the Modus Ponens rule

$$(MP) \ \ \frac{A \ \ ; \ \ (A \Rightarrow B)}{B},$$

where $A, B \in \mathcal{F}$.

Semantic Link: Logical Axioms LA

We want the set LA of logical axioms to be a non-empty set of classical predicate tautologies (8.13), i.e.

$$LA \subseteq \mathbf{T}_p,$$

where

$$\mathbf{T}_p = \{A \ \text{ of } \ \mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow,\neg\}}(\mathbf{P}, \mathbf{F}, \mathbf{C}) : \ \models_p A\}.$$

Remark 8.2

We use symbols $\models_p$, $\mathbf{T}_p$ to stress the fact that we talk about predicate lah=nguage and classical predicate tautologies.

Semantic Link 2: Rules of Inference $\mathcal{R}$

We want the the rules of inference $r \in \mathcal{R}$ of S to preserve truthfulness. Rules that do so are called sound. We define it formally as follows.

Definition 8.25 (Sound Rule)

Given an inference rule $r \in \mathcal{R}$ of the form

$$(r) \quad \frac{P_1 \ ; \ P_2 \ ; \ .... \ ; \ P_m}{C},$$

where $P_1.P_2, \ldots, P_m, C \in \mathcal{F}$.

(i) We say that the rule (r) is **sound** *if and only if the following condition holds for all structures $\mathbf{M} = [U, I]$ for $\mathcal{L}$.*

$$\textit{If} \ \ \mathbf{M} \ \models \ \{P_1, P_2, .P_m\} \ \ \textit{then} \ \ \ \mathbf{M} \ \models \ C. \tag{8.71}$$

(ii) The rule (r) is **not sound** *if and only if there is a structure $\mathbf{M} = [U, I]$, such that*

$$\mathbf{M} \ \models \ \{P_1, P_2, .P_m\} \ \ \textit{and} \ \ \ \mathbf{M} \ \not\models \ C. \tag{8.72}$$

In order to prove that the rule (r) is **sound** we have to show the implication (8.71). It means, by definitions 9.3, 8.18, we have to show that that if all premisses of the rule (r) are **true** in $\mathbf{M} = [U, I]$, so is its conclusion. This also justifies correctness of the definition 8.25; sound rules do preserve the **truthfulness** as it is defined in our semantics.

Exercise 8.8

Prove the soundness of the rule

$$(r1) \ \ \frac{\neg\forall x A(x)}{\exists x \neg A(x)}. \tag{8.73}$$

Proof
Assume that the soundness condition (8.71) does not hold for for all structures $\mathbf{M} = [U, I]$. It means we assume that there is a structure $\mathbf{M} = [U, I]$, such that $\mathbf{M} \models \neg\forall x A(x)$ and $\mathbf{M} \not\models \exists x \neg A(x)$.

Let $\mathbf{M} \models \neg\forall x A(x)$. By definition 9.3, for all $s : VAR \longrightarrow U$ we have $(\mathbf{M}, s) \models \neg\forall x \neg A(x))$. Hence by satisfaction definition 8.13, $(\mathbf{M}, s) \not\models \forall x A(x)$. This holds only if for all s', such that s, s' agree on all variables except on x, $(\mathbf{M}, s') \not\models A(x)$.
Observe that $(\mathbf{M}, s) \not\models \exists x \neg A(x)$ only if there is no s', such that $(\mathbf{M}, s') \models \neg A(x)$, i.e. there is no s', such that $(\mathbf{M}, s') \not\models A(x)$. This means that for all s', $(\mathbf{M}, s') \models A(x)$. Contradiction with $(\mathbf{M}, s') \not\models A(x)$.

Exercise 8.9

Prove the soundness of the rule

$$(r2) \ \ \frac{\forall x A(x)}{\exists x A(x)}. \tag{8.74}$$

Proof
Assume that the soundness condition (8.71) does not hold for for all structures $\mathbf{M} = [U, I]$. It means we assume that there is a structure $\mathbf{M} = [U, I]$, such that $\mathbf{M} \models \forall x A(x)$ and $\mathbf{M} \not\models \exists x A(x)$.

Let $\mathbf{M} \models \forall x A(x)$. By definition 9.3, for all $s : VAR \longrightarrow U$ we have $(\mathbf{M}, s) \models \forall x \neg A(x))$.

By definition 8.13, $(\mathbf{M}, s) \models \forall x\ A(x)$ and $(\mathbf{M}, s) \not\models \exists x\ A(x)$. It means that $(\mathbf{M}, s') \models A(x)$ for all s' such that s, s' agree on all variables except on x, and it is not true that there is s' such that s, s' agree on all variables except on x, and $(\mathbf{M}, s') \models A(x)$. This is impossible and this contradiction proves soundness of $(r2)$.

Exercise 8.10

Prove that the rule

$$(r3)\quad \frac{\exists x A(x)}{\forall x A(x)}. \tag{8.75}$$

is not sound.

Proof
Observe that to prove that the rule (8.75) is not sound we have to provide an example of an instance of a formula $A(x)$ and construct prove (ii) of definition 8.25 for it.

Let $A(x)$ be an atomic formula $P(x, c)$, for any $P \in \mathbf{P}, \#P = 2$. We take as $\mathbf{M} = (N, P_I :<,\ c_I : 3)$ for N set of natural numbers. Let s be any assignment $s : VAR \longrightarrow N$. Obviously $(\mathbf{M}, s) \models \exists x\ P(x, c)$.

Take any s' such that $s'(x) = 2$ and $s'(y) = s(y)$ for all $y \in VAR - \{x\}$. We have $(2, 3) \in P_I$, as $2 < 3$ and hence there exists s' that agrees with s on all variables except on x, and $(\mathbf{M}, s') \models P(x, c)$. But $(\mathbf{M}, s) \not\models \forall x\ P(x, c)$ as for example for s' such that $s'(x) = 5$ and $s'(y) = s(y)$ for all $y \in VAR - \{x\}$, $(2, 3) \notin P_I$, as $5 \not< 3$.

This proves that $\mathbf{M} = (N, P_I :<,\ c_I : 3)$ is a model for $(\exists x\ P(x, c)$ and hence $\not\models\ \forall x\ A(x))$.

The "shorthand" solution is: the formula $(\exists x\ P(x, c)$ becomes in $\mathbf{M} = (N, P_I :<,\ c_I : 3)$ a true mathematical statement (written with logical symbols): $\exists n\ n < 3$. The formula $(\forall x\ P(x, c)$ becomes a mathematical frmula $\forall n\ n < 3$ which is an obviously *false* statement in the set N of natural numbers, as there is $n \in N$, such that $n < 3$, for example $n = 2$, and it is not true that all natural numbers are smaller then 3. So the rule $(r3)$ is not sound.

Definition 8.26 (Strongly Sound Rule)

An inference rule $r \in \mathcal{R}$ of the form

$$(r) \quad \frac{P_1 \ ; \ P_2 \ ; \ .... \ ; \ P_m}{C}$$

is **strongly sound** *if the following condition holds for all structures* $\mathbf{M} = [U, I]$ *for* $\mathcal{L}$.

$$\mathbf{M} \models \{P_1, P_2, .P_m\} \quad \textit{if and only if} \quad \mathbf{M} \models C. \tag{8.76}$$

We can, and we do state it informally as: " an inference rule $r \in \mathcal{R}$ is strongly sound when the conjunction of all its premisses is logically equivalent to its conclusion". We denote it informally as

$$P_1 \cap P_2 \cap \ \dots \ \cap P_m \equiv C. \tag{8.77}$$

Example 8.22

The sound rule (8.73)

$$(r1) \quad \frac{\neg\forall x A(x)}{\exists x \neg A(x)}$$

is **strongly sound** *by De Morgan Law (8.46).*

The sound rule (8.75)

$$(r2) \quad \frac{\forall x A(x)}{\exists x A(x)}$$

is **not strongly sound** *by exercise 8.10.*

Definition 8.27 (Sound Proof System)

Given the **predicate (first order)** *proof system (11.1)*

$$S = (\mathcal{L}, \quad \mathcal{F}, \quad LA, \quad \mathcal{R}).$$

We say that the proof system S is **sound** *if the following conditions hold.*

(1) $LA \subseteq \mathbf{T}_p$;

(2) Each rule of inference $r \in \mathcal{R}$ *is* **sound**.

The proof system S is **strongly sound** *if the condition (2) is replaced by the following condition (2')*

(2') Each rule of inference $r \in \mathcal{R}$ *is* **strongly sound** *under* $\mathbf{M}$.

The set of **all provable expressions** of S is denoted by $\mathbf{P}_S$ and is defined as follows.

$$\mathbf{P}_S = \{A \in \mathcal{F} : \ \vdash_S A\}. \tag{8.78}$$

When we define (develop) a proof system S our first goal is to make sure that it a "sound" one, i.e. that all we prove in it is true. Proving the following theorem establishes this goal.

Theorem 8.7 (Soundness Theorem for S)

Given a predicate proof system S.
For any $A \in \mathcal{F}$, the following implication holds.

$$\textit{If } \vdash_S A \quad \textit{then} \quad \models_p A. \tag{8.79}$$

We write (8.79) it in a more concise form as

$$\mathbf{P}_S \subseteq \mathbf{T}_p. \tag{8.80}$$

Proof
Observe that if we have already proven that S is sound as stated in the definition 8.27, the proof of the implication (8.79) is straightforward mathematical induction over the length of a proof.

It means that in order to prove the Soundness Theorem 8.7 for a proof system Sit is enought to verify the two conditions of the definition 8.27 (1) $LA \subseteq \mathbf{T}_p$ and (2) each rule of inference $r \in \mathcal{R}$ is **sound**.

We again refer the reader to chapter 4 for detailed examples, exercises and problems.

As we can see, proving Soundness Theorem 8.7 for any proof system we develop is indispensable and the proof is quite easy. The next step in developing a logic (classical predicate logic in our case now) is to answer necessary and a difficult question: *Given a proof system S, about which we know that all it proves it true (tautology). Can we prove all we know to be true (all tautologies)?*

Proving the following theorem establishes this goal.

Theorem 8.8 (Completeness Theorem for S)

Given a predicate proof system S.
For any $A \in \mathcal{F}$, the following implication holds.

$$\vdash_S A \quad \textit{if and only if} \quad \models_p A. \tag{8.81}$$

We write (8.81) it in a more concise form as

$$\mathbf{P}_S = \mathbf{T}_p. \tag{8.82}$$

The Completeness Theorem consists of two parts:

Part 1: Soundness Theorem: $\mathbf{P}_S \subseteq \mathbf{T}_p$.

Part 2: Completeness part of the Completeness Theorem: $\mathbf{T}_p \subseteq \mathbf{P}_S$.

Proving the Soundness Theorem for S is usually a straightforward and not a very difficult task. Proving the *Completeness part* of the Completeness Theorem is always a crucial and very difficult task. There are many methods and techniques for doing so, even for classical proof systems (logics) alone. Non-classical logics often require new sometimes very sophisticated methods. We presented two proofs of the Completeness Theorem for classical propositional Hilbert style proof system in chapter 5, and a constructive proofs for automated theorem proving systems for classical propositional logic the chapter 6.

We present a proof of the Completeness Theorem for predicate (first order) logic in the next chapter 9.

8.5 Homework Problems

Predicate Languages

1. Given the following formulas $A_1 - A_5$ of a predicate language $\mathcal{L}$.

$$A_1 = R(x, y, g(c, x)),\ A_2 = \exists x P(x, f(x, y)),\ A_3 = \exists d R(x, y, g(c, d)),$$

$$A_4 = \forall z (f(x, P(c, y)),\quad A_5 = \exists y P(x, f(c, y)) \cup \forall y P(x, f(c, y)).$$

   (a) Indicate whether they are, or are not well formed formulas of $\mathcal{F}$. For those which are not in $\mathcal{F}$ write a correct formula.

   (b) For each correct, or corrected formula identify all components: connectives, quantifiers, predicate and function symbols, and list all its terms.

   (c) For each formula identify its s free and bound variables. State which are open and which are closed formulas (sentences), if any.

   (d) Describe a language defined by the set $\mathcal{F}_0 = \{A_1,\ A_2,\ \dots\ A_5\}$ of formulas that are correct or corrected.

2. For the following mathematical statements write their corresponding formulas of predicate language $\mathcal{L}$.

   (a) $\forall_{n>1}(n + 3 < 8 \cup \exists_{x \in R}\ x + n > 8)$

   (b) $\forall_{x \in R}\ \exists_{n \in N}(x + n > 0 \Rightarrow \exists_{m \in N}(m = x + n))$

   (c) If all natural numbers are smaller then zero, then the sum of any two integers is smaller then zero.

   (d) For all natural numbers The following implication holds for all natural numbers: if $n > 0$, then there is a real number x, such that $n + x = 0$ or there is an integer m, such that $m > 0$.

3. For each of the following formulas (some with restricted quantifiers) write 2 corresponding natural language sentences.

   (a) $\forall x(P(x) \Rightarrow \exists y Q(x,y))$.

   (b) $\forall x \exists y(P(x) \cap \neg Q(x,y))$.

   (c) $\forall_{A(x)} \exists_{A(y)} B(y)$.

   (d) $\exists_{P(x)} \forall_{N(x)} R(x,y)$.

4. Is the term $t = f(x,y)$ free for x in the following formulas?

   (a) $(P(x,y) \Rightarrow \forall y P(x,y)$.

   (b) $(\forall y P(x,y) \cup \exists y P(x,y))$.

   (c) $\forall x P(x,y)$.

   (d) $\forall y P(x,y)$.

   (e) $(\forall y Q(y) \Rightarrow P(x,y))$.

5. Justify that for any formula $A \in \mathcal{F}$ and any term $t \in \mathbf{T}$ the following facts hold.

   (a) A closed tern t, i.e. term with no variables is free for any variable x in A.

   (b) A term t is free for any variable in A if none of the variables in t is bound in A.

   (c) Term $t = x$ is free for x in any formula A.

   (d) Any term is free for x in A if A contains no free occurrences of x.

6. Translate the following formulas in everyday English.

   (a) $\forall x(P(x) \cap \forall y(\neg L(x,y) \Rightarrow \neg H(x)))$, where $P(x)$ means " x is a person", $L(x,y)$means "x likes y" , and $H(x)$ means "x is happy".

   (b) $\forall x((E(x) \cap P(x)) \Rightarrow E(x,b))$, where E(x) means " x is an even integer", P(x) means " x is prime", Q(x,y) means " x is equal to x", and b denotes 2.

   (c) $\neg \forall y((P(y) \cap \forall x(P(x)) \Rightarrow E(x,y)))$, where P(x) means "x is an integer, and $L(x,y)$means "$x \leq y$".

7. Use the restricted quantifiers to translate the following natural language sentences into a proper formulas of a proper formal predicate language $\mathcal{L} = \mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow,\neg\}}(\mathbf{P}, \mathbf{F}, \mathbf{C})$. In each case specify the sets **P, F, C**.

   (a) Some politician are honest, some are not.

   (b) Any sets that have the same elements are equal.

   (c) Somebody hates everyone who does not hate himself.

(d) Birds can fly and if anyone can fly Tweety can.

(e) Anyone who knows logic loves it.

Classical Semantics

1. Given a predicate language $\mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$ and a structure $\mathbf{M} = [U, I]$ such that $U = N$ and $P_I : =,\ f_I :\ +,\ g_I :\ \cdot,\ a_I : 0,\ b_I : 1$ for N set on natural numbers. For each of the following formula A decide whether $\mathbf{M} \models A$ or not. Do so by examining the corresponding mathematical statement defined by $\mathbf{M}$.

   (a) $\forall x \exists y (P(x, f(y, y)) \cup P(x, f(f(y, y), b)))$.

   (b) $\forall x \exists y (P(g(x, y), a) \Rightarrow (P(x, a) \cup P(y, a)))$.

   (c) $\exists y P(f(y, y), b)$.

2. Let $\mathbf{M} = [U, I]$ be a structure such that $U = Z$ and $P_I : =,\ f_I :\ +$ for Z set of integers. For each of the following formula A decide whether $\mathbf{M} \models A$ or not. Do so by examining the corresponding mathematical statement defined by $\mathbf{M}$.

   (a) $\forall x \forall y P(f(x, y), f(y, x))$.

   (b) $\forall x \forall y \forall z P(f(x, f(y, z)), f(f(x, y), z))$.

   (c) $\forall x \forall y \exists z P(f(x, z), y)$

3. Let $\mathbf{M} = [U, I]$ be a structure such that $U = N - \{0\}$ or N set of natural numbers and $P_I : =,\ f_I(x, y)$ is x^y For each of the following formula A decide whether $\mathbf{M} \models A$ or not. Do so by examining the corresponding mathematical statement defined by $\mathbf{M}$.

   (a) $\forall x \forall y P(f(x, y), f(y, x))$.

   (b) $\forall x \forall y \forall z P(f(x, f(y, z)), f(f(x, y), z))$.

   (c) $\forall x \forall y \exists z P(f(x, z), y)$.

4. For each formula below, where $P, Q \in \mathbf{P}$, find a structure $\mathbf{M} = [U, I]$ for $\mathcal{L} = \mathcal{L}_{\{\neg, \cap, \cup, \Rightarrow, \neg\}}(\mathbf{P}, \mathbf{F}, \mathbf{C})$ that is its counter model. Justify its correctness.

   (a) $(\ (\forall x P(x) \Rightarrow \forall y Q(y)) \Rightarrow (\forall x (P(x) \Rightarrow Q(x))))$.

   (b) $(\ (\forall x P(x) \cup \forall y Q(y)) \Rightarrow (\forall x (P(x) \cup \forall x Q(x))))$.

5. Show that the following formulas are predicate tautologies for any formulas A, B in $\mathcal{L}$.

   (a) $(\forall x \forall y A(x, y) \Rightarrow \forall y \forall x A(x, y))$.

   (b) $(\exists x \exists y A(x, y) \Rightarrow \exists y \exists x A(x, y))$.

(c) $(\forall x(A(x) \Rightarrow B(x)) \Rightarrow (\forall x A(x) \Rightarrow \forall x B(x)))$.

6. Prove that the following formulas are not predicate tautologies by finding their proper instances and constructing counter models for them.

   (a) $((\exists x A(x) \cap \exists x B(x)) \Rightarrow \exists x\ (A(x) \cap B(x)))$.

   (b) $(\forall x\ (A(x) \cup B(x)) \Rightarrow (\forall x A(x) \cup \forall x B(x)))$.

   (c) $((\forall x A(x) \Rightarrow \forall x B(x)) \Rightarrow \forall x(A(x) \Rightarrow B(x)))$.

7. Prove that the following formulas are predicate tautologies for any formulas $A(x), B(x), A, B$ of $\mathcal{L}$, such that A , B does not contain any free occurrences of x.

   (a) $(\forall x(A \Rightarrow B(x)) \Rightarrow (A \Rightarrow \forall x\ B(x)))$,

   (b) $(\exists x(A(x) \Rightarrow B) \Rightarrow (\forall x A(x) \Rightarrow B))$.

   (c) $(\forall x(A(x) \Rightarrow B) \Rightarrow (\exists x A(x) \Rightarrow B))$.

8. Prove that the restrictions: "A , B does not contain any free occurrences of x" are essential for all of the following tautologies, i.e. give examples of formulas for which the laws without these restrictions fail and construct counter models for them.

   (a) $(\forall x(A(x) \Rightarrow B) \Rightarrow (\exists x A(x) \Rightarrow B))$.

   (b) $(\exists x(A(x) \Rightarrow B) \Rightarrow (\forall x A(x) \Rightarrow B))$.

9. Prove that the *converse implication* to the formulas listed below are predicate tautologies for any formulas $A(x), B(x), A, B$ of $\mathcal{L}$, such that A, B does not contain any free occurrences of x.

   (a) $(\forall x(A \Rightarrow B(x)) \Rightarrow (A \Rightarrow \forall x\ B(x)))$,

   (b) $(\exists x(A(x) \Rightarrow B) \Rightarrow (\forall x A(x) \Rightarrow B))$.

   (c) $(\forall x(A(x) \Rightarrow B) \Rightarrow (\exists x A(x) \Rightarrow B))$.

Chapter 9

Hilbert Proof Systems Completeness of Classical Predicate Logic

There are several quite distinct approaches to the Completeness Theorem, corresponding to the ways of thinking about proofs. Within each of the approaches there are endless variations in exact formulation, corresponding to the choice of methods we want to use to proof the Completeness Theorem. Different basic approaches are important, though, for they lead to different applications. We have presented two of the approaches for the propositional logic: Hilbert style formalizations (proof systems) in chapter 5, and Gentzen style formalizations (automated proof systems) in chapter 6. We have also presented for each of the approaches methods of proving the completeness theorem. Two proofs of completeness theorem for Hilbert style proof system in chapter 5 and a constructive proofs for several Gentzen style proof systems in chapter 6.

There are many proofs of the Completeness Theorem for predicate (first order) logic. We present here in a great detail, a version of Henkin's proof as included in a classic *Handbook of Mathematical Logic* (1977). It contains a method for reducing certain problems of first-order logic back to problems about propositional logic. We give independent proof of Compactness Theorem 9.1 for propositional logic. Reduction to Propositional Logic Theorem 9.2, Compactness Theorem 9.3 for first-order logic, Löwenheim-Skolem Theorem 9.4 and Gödel Completeness Theorem 9.7 fall out of the Henkin method.

We choose this particular proof of completeness of first order logic not only for it being one of the oldest and most classical, but also for its connection with the propositional logic. Moreover, the proof of the Compactness Theorem 9.1 is based on semantical version of syntactical notions and techniques crucial to the second proof of completeness theorem for propositional logic covered in chapter 5 and hence is familiar to the reader.

9.1 Reduction Predicate Logic to Propositional Logic

Let $\mathcal{L} = \mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$ be a first order language with equality (definition 9.12). We assume that the sets **P, F, C** are infinitely enumerable. We also assume that it has a full set of propositional connectives, i.e.

$$\mathcal{L} = \mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow\}}(\mathbf{P}, \mathbf{F}, \mathbf{C}).$$

Our goal now is to define a **propositional logic** within $\mathcal{L} = \mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$. We do it in a sequence of steps.

First we define a special subset $P\mathcal{F}$ of formulas of $\mathcal{L}$, called a set of all **propositional formulas** of $\mathcal{L}$.

Intuitively these are formulas of $\mathcal{L}$ which are not direct propositional combination of simpler formulas, that are atomic formulas or formulas beginning with quantifiers. Formally, we have the following.

Definition 9.1 (Prime Formulas)

Any formula from that set $\mathcal{P}$ defined by (9.1) is called a **prime formula** *of $\mathcal{L}$.*

$$\mathcal{P} = A\mathcal{F} \cup \{\forall xB, \quad B \in \mathcal{F}\} \cup \{\forall xB : \quad B \in \mathcal{F}\}, \tag{9.1}$$

where the set $A\mathcal{F}$ is the set of all atomic formulas of $\mathcal{L}$.

The set $\mathcal{P} \subseteq \mathcal{F}$, called a set of all prime formulas of $\mathcal{L}$ plays in the propositional logic we define the role

Example 9.1

The following are primitive formulas.

$R(t_1, t_2)$, $\quad \forall x(A(x) \Rightarrow \neg A(x))$, $\quad (c = c)$, $\quad \exists x(Q(x, y) \cap \forall yA(y))$.

The following are not primitive formulas.

$(R(t_1, t_2) \Rightarrow (c = c))$, $\quad (R(t_1, t_2) \cup \forall x(A(x) \Rightarrow \neg A(x))$.

Given a set $\mathcal{P}$ of primitive formulas we define in a standard way the set $P\mathcal{F}$ of *propositional formulas* of $\mathcal{L}$. as follows.

Definition 9.2 (Propositional Formulas of $\mathcal{L}$)

Let $\mathcal{F}, \mathcal{P}$ be sets of all formulas and prime formulas (9.1) of $\mathcal{L}$, respectively. The smallest set $P\mathcal{F} \subseteq \mathcal{F}$ such that

(i) $\mathcal{P} \subseteq P\mathcal{F}$,

(ii)If $A, B \in P\mathcal{F}$, *then* $(A \Rightarrow B), (A \cup B), (A \cap B)$, *and* $\neg A \in P\mathcal{F}$

is called a set of all **propositional formulas** *of the predicate language* $\mathcal{L}$.

The set $\mathcal{P}$ *is called* **atomic propositional** *formulas of* $\mathcal{L}$.

Propositional Semantics for $\mathcal{L}$

We define propositional semantics for propositional formulas in $P\mathcal{F}$ as follows.

Definition 9.3 (Truth assignment)

Let $\mathcal{P}$ be a set of **atomic propositional** *formulas of $\mathcal{L}$ and $\{T, F\}$ be the set of logical values "true" and "false". Any function*

$$v : \mathcal{P} \longrightarrow \{T, F\}$$

is called a **truth assignment** *in* $\mathcal{L}$.

We extend v to the set $P\mathcal{F}$ of **all propositional** formulas of $\mathcal{L}$ by defining the mapping

$$v^* : \ P\mathcal{F} \longrightarrow \{T, F\}$$

as follows.

$v^*(A) = v(A)$ for $A \in \mathcal{P}$,

and for any $A, B \in P\mathcal{F}$,

$v^*(A \Rightarrow B) = v^*(A) \Rightarrow v^*(B)$,

$v^*(A \cup B) = v^*(A) \cup v^*(B)$,

$v^*(A \cap B) = v^*(A) \cap v^*(B)$,

$v^*(\neg A) = \neg v^*(A)$.

Definition 9.4 (Propositional Model)

A truth assignment $v : \mathcal{P} \longrightarrow \{T, F\}$ is called a propositional model *for a formula $A \in P\mathcal{F}$ if and only if $v^*(A) = T$.*

Definition 9.5 (Propositional Tautology)

For any formula $A \in P\mathcal{F}$,
$A \in P\mathcal{F}$ is a propositional tautology *of $\mathcal{L}$ if and only if*
$v^*(A) = T$ for all $v : \mathcal{P} \longrightarrow \{T, F\}$.

For the sake of simplicity we will often say *model, tautology* instead *propositional model, propositional tautology* for $\mathcal{L}$.

Definition 9.6 (Model for the Set)

Given a set S of propositional formulas. We say that v is a model for the set *S if and only if v is a model for all formulas $A \in S$.*

Definition 9.7 (Consistent Set)

A set $S \subseteq P\mathcal{F}$ of propositional formulas of $\mathcal{L}$ is **consistent** *(in a sense of propositional logic) if it has a (propositional) model.*

Definition 9.8 (Inconsistent Set)

A set $S \subseteq P\mathcal{F}$ of propositional formulas of $\mathcal{L}$ is **inconsistent** *if it has no model.*

Theorem 9.1 (Compactness Theorem for Propositional Logic of $\mathcal{L}$)

A set $S \subseteq P\mathcal{F}$ of propositional formulas of $\mathcal{L}$ is **consistent** *if and only if every* **finite** *subset of S is* **consistent***.*

Proof
Assume that S is a consistent set. By definition 9.7, it has a model. Tts model is also a model for all its subsets, including all finite subsets, and so all its finite subsets are consistent.

To prove the converse implication, i.e. the **nontrivial half** of the Compactness Theorem we write it in a slightly modified form. To do so, we introduce the following definition.

Definition 9.9 (Finitely Consistent Set (FC))

Any set S such that all its finite subsets are consistent is called finitely consistent.

We use this definition 9.9 to re-write the Compactness Theorem as follows.

A set S of propositional formulas of $\mathcal{L}$ is consistent if and only if it is finitely consistent.

The nontrivial half of it still to be proved is now stated now as

Every finitely consistent set of propositional formulas of $\mathcal{L}$ is consistent.

The **proof** of the nontrivial half of the Compactness Theorem 9.1, as stated above, consists of the following four steps.

S1 We introduce the notion of a *maximal finitely consistent set.*

S2 We show that every *maximal finitely consistent set* is consistent by constructing its model.

S3 We show that every *finitely consistent set* S can be extended to a *maximal finitely consistent set* S^*. I.e we show that for every finitely consistent set S there is a set S^*, such that $S \subseteq S^*$ and S^* is maximal finitely consistent.

S4 We use **S2** and **S3** to justify the following reasoning.

Given a *finitely consistent* set S. We extend it, via construction to be defined in the step **S3** to a *maximal finitely consistent* set S^*. By the **S2**, S^* is consistent and hence so is the set S.
This **ends the proof** of the Compactness Theorem 9.1.

Here are the details and proofs needed for completion of steps **S1 - S4**.

Step **S1**
We introduce the following definition.

Definition 9.10 (Maximal Finitely Consistent Set (MFC))

Any set $S \subseteq P\mathcal{F}$ is maximal finitely consistent *if it is finitely consistent and for every formula A, either $A \in S$ or $\neg A \in S$.*

We use notation **MFC** for maximal finitely consistent set, and **FC** for the finitely consistent set.

Step **S2**
We prove the following **MFC** lemma 6.3 and the Property 9.1.

Lemma 9.1

Any MFC set is consistent.

Proof
Given a MFC set denoted by S^*. We prove its consistency by constructing model for it, i.e. by constructing a truth assignment $v : \mathcal{P} \longrightarrow \{T, F\}$, such that for all $A \in S^*$, $v^*(A) = T$.

Observe that directly from the definition 9.10 we have the following property of the the **MFC** sets.

Property 9.1 (MFC)

For any MFC set S^* and for every $A \in P\mathcal{F}$, exactly one of the formulas A, $\neg A$ belongs to S^*.

In particular, for any $P \in P\mathcal{F}$, we have that exactly one of formulas P, $\neg P$ belongs to S^*. This justifies the correctness of the following definition.

Definition 9.11

For any **MFC** *set S^*, mapping $v : \mathcal{P} \longrightarrow \{T, F\}$, such that*

$$v(P) = \begin{cases} T & \textit{if } P \in S^* \\ F & \textit{if } P \notin S^* \end{cases}$$

is called a truth assignment defined by S^*.
We extend v to $v^* : P\mathcal{F} \longrightarrow \{T, F\}$ in a usual way.

We prove now that the truth assignment v defined by S^* (definition 9.11) is a **model** for S^*, we show for any $A \in P\mathcal{F}$,

$$v^*(A) = \begin{cases} T & \text{if } A \in S^* \\ F & \text{if } A \notin S^* \end{cases}$$

We prove it by induction on the degree of the formula A as follows.

The base case of $A \in \mathcal{P}$ follows immediately from the definition of v.

Case $A = \neg C$ Assume that $A \in S^*$. This means $\neg C \in S^*$ and by **MCF Property** we have that $C \notin S^*$. So by the inductive assumption $v^*(C) = F$ and $v^*(A) = v^*(\neg C) = \neg v^*(C) = \neg F = T$.

Assume now that $A \notin S^*$. By **MCF** Property **??** we have that $C \in S^*$. By the inductive assumption $v^*(C) = T$ and $v^*(A) = v^*(\neg C) = \neg v^*(T) = \neg T = F$.

This proves that for any formula A,

$$v^*(\neg A) = \begin{cases} T & \text{if } \neg A \in S^* \\ F & \text{if } \neg A \notin S^* \end{cases}$$

Case $A = (B \cup C)$ Let $(B \cup C) \in S^*$. It is enough to prove that in this case $B \in S^*$ and $C \in S^*$, because then from the inductive assumption $v^*(C) = v^*(D) = T$ and $v^*(B \cup C) = v^*(B) \cup v^*(C) = T \cup T = T$.

Assume that $(B \cup C) \in S^*$, $B \notin S^*$ and $C \notin S^*$. Then by**MCF** Property **??** we have that $\neg B \in S^*$, $\neg C \in S^*$ and consequently the set

$$\{(B \cup C), \neg B, \neg C\}$$

is a finite inconsistent subset of S^*, what contradicts the fact that S^* is finitely consistent.

Assume now that $(B \cup C) \notin S^*$. By **MCF** Property **??**, $\neg(B \cup C) \in S^*$ and by the $A = \neg C$ we have that $v^*(\neg(B \cup C)) = T$. But $v^*(\neg(B \cup C)) = \neg v^*((B \cup C)) = T$ means that $v^*((B \cup C)) = F$, what end the proof of this case.

The remaining cases of $A = (B \cap C), A = (B \Rightarrow C)$ are similar to the above and are left to the reader as an exercise.

This **end the proof** of lemma 9.1 and completes the step **S2**.

S3: Maximal finitely consistent extension

Given a finitely consistent set S, we construct its *maximal finitely consistent extension* S^* as follows.

The set of all formulas of $\mathcal{L}$ is countable, so is $P\mathcal{F}$. We assume that all propositional formulas form a one-to-one sequence

$$A_1, A_2, ..., A_n, ... \tag{9.2}$$

We define a chain

$$S_0 \subseteq S_1 \subseteq S_2 .... \subseteq S_n \subseteq .... \tag{9.3}$$

of *extentions* of the set S by

$$S_0 = S;$$

$$S_{n+1} = \begin{cases} S_n \cup \{A_n\} & \text{if } S_n \cup \{A_n\} \text{ is finitely consistent} \\ S_n \cup \{\neg A_n\} & \text{otherwise.} \end{cases}$$

We take

$$S^* = \bigcup_{n \in N} S_n. \tag{9.4}$$

Clearly, $S \subseteq S^*$ and for every A, either $A \in S^*$ or $\neg A \in S^*$. To finish the proof that S^* is MCF we have to show that it is finitely consistent.

First, let observe that if all sets S_n are finitely consistent, so is $S^* = \bigcup_{n \in N} S_n$. Namely, let $S_F = \{B_1, ..., B_k\}$ be a finite subset of S^*. This means that there are sets $S_{i_1}, ...S_{i_k}$ in the chain (9.3) such that $B_m \in S_{i_m}$, $m = 1, ..k$. Let $M = max(i_1, ...i_k)$. Obviously $S_F \subseteq S_M$ and S_M is finitely consistent as an element of the chain (9.3). This proves the if all sets S_n are finitely consistent, so is S^*.

Now we have to prove only that all S_n in the chain (9.3) are finitely consistent. We carry the proof by induction over the length of the chain. $S_0 = S$, so it is FC by assumption of the Compactness Theorem 9.1. Assume now that S_n is FC, we prove that so is S_{n+1}. We have two cases to consider.

Case 1 $S_{n+1} = S_n \cup \{A_n\}$, then S_{n+1} is FC by the definition of the chain (9.3).

Case 2 $S_{n+1} = S_n \cup \{\neg A_n\}$. Observe that this can happen only if $S_n \cup \{A_n\}$ is not FC, i.e. there is a finite subset $S_n^{'} \subseteq S_n$, such that $S_n^{'} \cup \{A_n\}$ is not consistent.

Suppose now that S_{n+1} is not FC. This means that there is a finite subset $S_n^{''} \subseteq S_n$, such that $S_n^{''} \cup \{\neg A_n\}$ is not consistent.

Take $S_n^{'} \cup S_n^{''}$. It is a finite subset of S_n so is consistent by the inductive assumption. Let v be a model of $S_n^{'} \cup S_n^{''}$. Then *one* of $v^*(A), v^*(\neg A)$ *must be* T. This contradicts the inconsistency of both $S_n^{'} \cup \{A_n\}$ and $S_n^{'} \cup \{\neg A_n\}$.

Thus, in ether case, S_{n+1}, is after all consistent.

This **completes** the proof of the step **S3** and the proof of the **compactness theorem** for propositional logic of $\mathcal{L}$ (theorem 9.1) via the argument presented in the step **S4**.

9.1.1 Henkin Method

Propositional tautologies within $\mathcal{L}$, as defined here (definition 9.5) barely scratch the surface of the collection of predicate (first -order) tautologies, i.e. of the predicate *valid* formulas, as they are often called. For example the following first-order formulas are propositional tautologies,

$$(\exists x A(x) \cup \neg \exists x A(x)),$$

$$(\forall x A(x) \cup \neg \forall x A(x)),$$

$$(\neg(\exists x A(x) \cup \forall x A(x)) \Rightarrow (\neg \exists x A(x) \cap \neg \forall x A(x))),$$

but the following are predicate (first order) tautologies (valid formulas) that are not propositional tautologies:

$$\forall x (A(x) \cup \neg A(x)),$$

$$(\neg \forall x A(x) \Rightarrow \exists x \neg A(x)).$$

The first formula above is just a prime formula, the second is of the form $(\neg B \Rightarrow C)$, for B and C prime.

To stress the difference between the propositional and predicate (first order) tautologies some books reserve the word *tautology* for the propositional tautologies alone, using the notion of *valid formula* for the predicate (first order) tautologies. We use here both notions, with the preference to *predicate tautology* or *tautology* for short when there is no room for misunderstanding.

To make sure that there is no misunderstandings we remind the following definitions from chapter 8.

Given a first order language $\mathcal{L}$ with the set of variables VAR and the set of formulas $\mathcal{F}$. Let $\mathcal{M} = [M, I]$ be a structure for the language $\mathcal{L}$, with the universe M and the interpretation I and let $s : VAR \longrightarrow M$ be an assignment of $\mathcal{L}$ in M. We bring back some basic definitions from Chapter 8

A is satisfied in $\mathcal{M}$

Given a structure $\mathcal{M} = [M, I]$, we say that a formula A is **satisfied** in $\mathcal{M}$ if there is an assignment $s : VAR \longrightarrow M$ such that

$$(\mathcal{M}, s) \models A.$$

A is true in $\mathcal{M}$

Given a structure $\mathcal{M} = [M, I]$, we say that a formula A is **true** in $\mathcal{M}$ if $(\mathcal{M}, s) \models A$ for all assignments $s : VAR \longrightarrow M$.

Model $\mathcal{M}$

If A is **true** in a structure $\mathcal{M} = [M, I]$, then $\mathcal{M}$ is called a **model** for A. We denote it as

$$\mathcal{M} \models A.$$

A is predicate tautology (valid)

A formula A is a predicate tautology (valid) if it is true in all structures $\mathcal{M} = [M, I]$, i.e. if all structures are models of A.

We use use the term **predicate tautology** and and denote it, when there is no confusion with propositional case as

$$\models A.$$

Case: A is a sentence

If A is a sentence, then the truth or falsity of $(\mathcal{M}, s) \models A$ is completely independent of s. Thus we write

$$\mathcal{M} \models A$$

and read $\mathcal{M}$ *is a model of* A, if for some (hence every) valuation s, $(\mathcal{M}, s) \models A$.

Model of a set of sentences

$\mathcal{M}$ is a model of a set S of sentences if and only if $\mathcal{M} \models A$ for all $A \in S$. We write it

$$\mathcal{M} \models S.$$

Predicate and Propositional Models

The relationship between the predicate models that are defined in terms of structures $\mathcal{M} = [M, I]$ and assignments $s : VAR \longrightarrow M$ and propositional models that are defined in terms of truth assignments $v : \mathcal{P} \longrightarrow \{T, F\}$ is established by the following lemma.

Lemma 9.2

Let $\mathcal{M} = [M, I]$ be a structure for the language $\mathcal{L}$ and let $s : VAR \longrightarrow M$ an assignment in $\mathcal{M}$. There is a truth assignments $v : \mathcal{P} \longrightarrow \{T, F\}$ such that for all formulas A of $\mathcal{L}$,

$$(\mathcal{M}, s) \models A \;\; if \; and \; only \; if \;\; v^*(A) = T.$$

In particular, for any set S of sentences of $\mathcal{L}$,
if $\mathcal{M} \models S$ then S is consistent in sense of propositional logic.

Proof For any prime formula $A \in P$ we define

$$v(A) = \begin{cases} T & \text{if } (\mathcal{M}, s) \models A \\ F & \text{otherwise.} \end{cases}$$

Since every formula in $\mathcal{L}$ is either prime or is built up from prime formulas by means of propositional connectives, the conclusion is obvious.

Observe, that the converse of the lemma is far from true. Consider a set

$$S = \{\forall x(A(x) \Rightarrow B(x)), \forall x A(x), \exists x \neg B(x)\}.$$

All formulas of S are different prime formulas, S is hence consistent in the sense of propositional logic and obviously has no predicate (first-order) model.

Definition 9.12 (Language with Equality)

A predicate language
$$\mathcal{L} = \mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$$
is called a first order (predicate) language with equality if we one it its predicate symbols is a two argument symbol $E \in \mathbf{P}$ representing an identity relation.

We write $t = s$ as the abbreviation of $E(t, s)$ for any terms $t, s \in \mathbf{T}$ and $t \neq s$ as the abbreviation of $\neg E(t, s)$.

Let $\mathcal{L} = \mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow\}}(\mathbf{P}, \mathbf{F}, \mathbf{C})$ be a predicate (first order) language with equality. We adopt a following set of axioms.

Equality Axioms (9.5)

For any free variable or constant of $\mathcal{L}$, i.e for any $u, w, u_i, w_i \in (VAR \cup \mathbf{C})$,

E1 $u = u$,

E2 $(u = w \Rightarrow w = u)$,

E3 $((u_1 = u_2 \cap u_2 = u_3) \Rightarrow u_1 = u_3)$,

E4 $((u_1 = w_1 \cap ... \cap u_n = w_n) \Rightarrow (R(u_1, ..., u_n) \Rightarrow R(w_1, ..., w_n)))$,

E5 $((u_1 = w_1 \cap ... \cap u_n = w_n) \Rightarrow (t(u_1, ..., u_n) \Rightarrow t(w_1, ..., w_n)))$,

where $R \in \mathbf{P}$ and $t \in \mathbf{T}$, i.e. R is an arbitrary n-ary relation symbol of $\mathcal{L}$ and $t \in \mathbf{T}$ is an arbitrary n-ary term of $\mathcal{L}$.

Observe that given any structure $\mathcal{M} = [M, I]$. We have by simple verification that for all $s : VAR \longrightarrow M$, and for all $A \in \{E1, E2, E3, E4, E5\}$,

$$(\mathcal{M}, s) \models A.$$

This proves the following

Fact 9.1

All equality axioms are predicate tautologies (valid) of $\mathcal{L}$.

This is why we still call logic with equality axioms added to it, a logic.

Henkin's Witnessing Expansion of $\mathcal{L}$

Now we are going to define notions that are fundamental to the Henkin's technique for reducing predicate logic to propositional logic. The first one is that of witnessing expansion of the language $\mathcal{L}$.

We construct an expansion of the language $\mathcal{L}$ by adding a set C of new constants to it, i.e. by adding a specially constructed the set C to the set $\mathbf{C}$ such that $C \cap \mathbf{C} = \emptyset$. The construction of the expansion is described below. The language such constructed is called witnessing expansion of the language $\mathcal{L}$.

Definition 9.13

For any predicate language $\mathcal{L} = \mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$, the language

$$\mathcal{L}(C) = \mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C} \cup C))$$

for the set C defined by (10.11) and $\mathcal{L}(C)$ defined by (9.9) and the construction described below is called a **witnessing expansion** *of $\mathcal{L}$. We write also*

$$\mathcal{L}(C) = \mathcal{L} \cup C.$$

Construction of the witnessing expansion of $\mathcal{L}$

We define the set C of new constants by constructing an infinite sequence

$$C_0, C_1, ..., C_n, ... \tag{9.6}$$

of sets of constants together with an infinite sequence

$$\mathcal{L}_0, \mathcal{L}_1, ..., \mathcal{L}_n, ... \tag{9.7}$$

We define sequences (11.25), (9.7) as follows. Let

$$C_0 = \emptyset, \quad \mathcal{L}_0 = \mathcal{L} \cup C_0 = \mathcal{L}.$$

We denote by

$$A[x]$$

the fact that the formula A has exactly one free variable and for each such a formula we introduce a distinct new constant denoted by

$$c_{A[x]}.$$

We define

$$C_1 = \{c_{A[x]} : \quad A[x] \ \in \mathcal{L}_0\}, \quad \mathcal{L}_1 = \mathcal{L} \cup C_1.$$

Assume that we have defined C_n and $\mathcal{L}_n$. We assign distinct new constant symbol $c_{A[x]}$ to each formula $A[x]$ of $\mathcal{L}_n$ which is not already a formula of $\mathcal{L}_{n-1}$ (i.e., if some constant from C_n appears in $A[x]$). We write it informally as $A[x] \in (\mathcal{L}_n - \mathcal{L}_{n-1})$. We define $C_{n+1} = C_n \cup \{c_{A[x]} : \quad A[x] \in (\mathcal{L}_n - \mathcal{L}_{n-1})\}$ and $\mathcal{L}_{n+1} = \mathcal{L} \cup C_{n+1}$. We put

$$C = \bigcup C_n \tag{9.8}$$

and

$$\mathcal{L}(C) = \mathcal{L} \cup C. \tag{9.9}$$

Definition 9.14 (Henkin Axioms)

The following sentences

H1 $\quad (\exists x A(x) \Rightarrow A(c_{A[x]}))$,

H2 $\quad (A(c_{\neg A[x]}) \Rightarrow \forall x A(x))$

are called **Henkin axioms** *and for any formula A, a constant $c_{A[x]} \in C$ as defined by (10.11) called a* **witnessing constant**.

The informal idea behind the Henkin axioms is the following.

The axiom **H1** says:
If $\exists x A(x)$ is true in a structure, choose an element a satisfying $A(x)$ and give it a new name $c_{A[x]}$.

The axiom **H2** says:
If $\forall x A(x)$ is false, choose a counterexample b and call it by a new name $c_{\neg A[x]}$.

Definition 9.15 (QuantifiersAxioms)

The following sentences

Q1 $(\forall x A(x) \Rightarrow A(t))$, *t is a closed term of* $\mathcal{L}(C)$;

Q2 $(A(t) \Rightarrow \exists x A(x))$, *t is a closed term of* $\mathcal{L}(C)$

are called **quantifiers axioms**.

Observe that the quantifiers axioms **Q1, Q2** obviously are predicate tautologies.

Definition 9.16 (Henkin Set)

Any set of sentences of $\mathcal{L}(C)$ which are either Henkin axioms (definition 9.14) or quantifiers axioms (definition 9.15) is called the **Henkin set** *and denoted by*

$$S_{Henkin}.$$

The *Henkin* is obviously not true in every $\mathcal{L}(C)$-structure, but we are going to show that every $\mathcal{L}$ -structure can be turned into an $\mathcal{L}(C)$-structure which is a **model** of S_{Henkin}. Before we do so we need to introduce two new notions.

Reduct and Expansion

Given two languages $\mathcal{L}$ and $\mathcal{L}^{'}$ such that $\mathcal{L} \subseteq \mathcal{L}^{'}$. Let $\mathcal{M}^{'} = [M, I^{'}]$ be a structure for $\mathcal{L}^{'}$. The structure

$$\mathcal{M} = [M, I^{'} \mid \mathcal{L}]$$

is called the *reduct* of $\mathcal{M}^{'}$ to the language $\mathcal{L}$ and $\mathcal{M}^{'}$ is called the *expansion* of $\mathcal{M}$ to the language $\mathcal{L}^{'}$.

Thus the reduct and the expansion $\mathcal{M}^{'}$ and $\mathcal{M}$ are the same except that $\mathcal{M}^{'}$ assigns meanings to the symbols in $\mathcal{L} - \mathcal{L}^{'}$.

Lemma 9.3

Let $\mathcal{M} = [M, I]$ be any structure for the language $\mathcal{L}$ and let $\mathcal{L}(C)$ be the witnessing expansion of $\mathcal{L}$. There is an expansion $\mathcal{M}^{'} = [M, I^{'}]$ of $\mathcal{M} = [M, I]$ such that $\mathcal{M}^{'}$ is a model of the set S_{Henkin}

Proof In order to define the expansion of $\mathcal{M}$ to $\mathcal{M}^{'}$ we have to define the interpretation $I^{'}$ for the symbols of the language $\mathcal{L}(C) = \mathcal{L} \cup C$, such that $I^{'} \mid \mathcal{L} = I$. This means that we have to define $c_{I'}$ for all $c \in C$. By the definition, $c_{I'} \in M$, so this also means that we have to assign the elements of M to all constants $c \in C$ in such a way that the resulting expansion is a model for all sentences from S_{Henkin}.

The quantifier axioms (definition 9.15) are predicate tautologies so they are going to be true regardless, so we have to worry only about the Henkin axioms (definition 9.14). Observe now that if the lemma 9.3 holds for the Henkin axiom **H1**, then it must hold for the axiom **H2**. Namely, let's consider the axiom **H2**:

$$(A(c_{\neg A[x]}) \Rightarrow \forall x A(x)).$$

Assume that $A(c_{\neg A[x]})$ is true in the expansion $\mathcal{M}^{'}$, i.e. that $\mathcal{M}^{'} \models A(c_{\neg A[x]})$ and that $\mathcal{M}^{'} \not\models \forall x A(x)$. This means that $\mathcal{M}^{'} \models \neg\forall x A(x)$ and by the de Morgan Laws, $\mathcal{M}^{'} \models \exists x \neg A(x)$. But we have assumed that $\mathcal{M}^{'}$ is a model for $H1$. In particular $\mathcal{M}^{'} \models (\exists x \neg A(x) \Rightarrow \neg A(c_{\neg A[x]}))$, and hence $\mathcal{M}^{'} \models \neg A(c_{\neg A[x]})$ and this contradicts the assumption that $\mathcal{M}^{'} \models A(c_{\neg A[x]})$. Thus if $\mathcal{M}^{'}$ is a model for all axioms of the type **H1**, it is also a model for all axioms of the type **H2**.

We define $c_{I'}$ for all $c \in C = \bigcup C_n$ by induction on n. Let $n = 1$ and $c_{A[x]} \in C_1$. By definition, $C_1 = \{c_{A[x]} : A[x] \in \mathcal{L}\}$. In this case we have that $\exists x A(x) \in \mathcal{L}$ and hence the notion $\mathcal{M} \models \exists x A(x)$ is well defined, as $\mathcal{M} = [M, I]$ is the structure for the language $\mathcal{L}$.

As we consider arbitrary structure $\mathcal{M}$, there are two possibilities: $\mathcal{M} \models \exists x A(x)$ or $\mathcal{M} \not\models \exists x A(x)$.

We define $c_{I'}$, for all $c \in C_1$ as follows.

If $\mathcal{M} \models \exists x A(x)$, then $(\mathcal{M}, v') \models A(x)$ for certain $v'(x) = a \in M$. We set $(c_{A[x]})_{I'} = a$. If $\mathcal{M} \not\models \exists x A(x)$, we set $(c_{A[x]})_{I'}$ arbitrarily.

This makes all the positive Henkin axioms about the $c_{A[x]} \in C_1$ true, i.e. $\mathcal{M} = (M, I) \models (\exists x A(x) \Rightarrow A(c_{A[x]}))$. But once $c_{A[x]} \in C_1$ are all interpreted in M, then the notion $\mathcal{M}^{'} \models A$ is defined for all formulas $A \in \mathcal{L} \cup C_1$. We carry the same argument and define $c_{I'}$, for all $c \in C_2$ and so on. The inductive step in the exactly the same way as the one above.

Definition 9.17 (Canonical structure)

Given a structure $\mathcal{M} = [M, I]$ for the language $\mathcal{L}$. The expansion $\mathcal{M}^{'} = [M, I^{'}]$ of $\mathcal{M} = [M, I]$ is called a **canonical structure** *for $\mathcal{L}(C)$ if all $a \in M$ are denoted by some $c \in C$. That is,*

$$M = \{c_{I'} : \ c \in C\}.$$

Now we are ready to state and proof a lemma 9.2 that provides the essential step in the proof of the completeness theorem for predicate logic.

Theorem 9.2 (The reduction to propositional logic)

Let $\mathcal{L} = \mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C})$ be a predicate language and let $\mathcal{L}(C) = \mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C} \cup C)$ be a witnessing expansion of $\mathcal{L}$.

For any set S of sentences of $\mathcal{L}$ the following conditions are equivalent.

(i) *S has a model, i.e. there is a structure $\mathcal{M} = [M, I]$ for the language $\mathcal{L}$ such that $\mathcal{M} \models A$ for all $A \in S$.*

(ii) *There is a* **canonical** *$\mathcal{L}(C)$ structure $\mathcal{M} = [M, I]$ which is a model for S, i.e. such that $\mathcal{M} \models A$ for all $A \in S$.*

(iii) *The set $S \cup S_{Henkin} \cup EQ$ is consistent in sense of propositional logic, where EQ denotes the equality axioms $E1 - E5$.*

Proof The implication $(ii) \rightarrow (i)$ is immediate. The implication $(i) \rightarrow (iii)$ follows from lemma 9.3. We have to prove only the implication $(iii) \rightarrow (ii)$.

Assume that the set $S \cup S_{Henkin} \cup EQ$ is consistent in sense of propositional logic and let v be a truth assignment to the prime sentences of $\mathcal{L}(C)$, such that $v^*(A) = T$ for all $A \in S \cup S_{Henkin} \cup EQ$. To prove the lemma, we construct a canonical $\mathcal{L}(C)$ structure $\mathcal{M} = [M, I]$ such that, for all sentences A of $\mathcal{L}(C)$,

$$\mathcal{M} \models A \text{ if and only if } v^*(A) = T. \tag{9.10}$$

The truth assignment v is a propositional model for the set S_{Henkin}, so v^* satisfies the following conditions:

$$v^*(\exists x A(x)) = T \text{ if and only if } v^*(A(c_{A[x]})) = T, \tag{9.11}$$

$$v^*(\forall x A(x)) = T \text{ if and only if } v^*(A(t)) = T, \tag{9.12}$$

for all closed terms t of $\mathcal{L}(C)$.

The conditions (9.11) and (9.12) allow us to construct the **canonical** $\mathcal{L}(C)$ model $\mathcal{M} = [M, I]$ out of the constants in C in the following way.

To define $\mathcal{M} = [M, I]$ we must (1.) specify the universe M of $\mathcal{M}$, (2.) define, for each n-ary predicate symbol $R \in \mathbf{P}$, the interpretation R_I as an n-argument relation in M, (3.) define, for each n-ary function symbol $f \in \mathbf{F}$, the interpretation $f_I : M^n \rightarrow M$, and (4.) define, for each constant symbol c of $\mathcal{L}(C)$, i.e. $c \in \mathbf{C} \cup C$, an element $c_I \in M$.

The construction of $\mathcal{M} = [M, I]$ must be such that the condition (9.10) holds for for all sentences A of $\mathcal{L}(C)$. This condition (9.10) tells us how to construct the definitions (1.) - (4.) above. Here are the definitions.

(1.) **Definition** of the universe M of $\mathcal{M}$.

In order to define the universe M we first define a relation $\approx$ on C by

$$c \approx d \ \text{ if and only if } \ v(c = d)) = T. \tag{9.13}$$

The equality axioms axioms guarantee that the relation (9.13) is equivalence relation on C, i.e. is reflexive, symmetric, and transitive. All axioms are predicate tautologies, so $v(c = d)) = T$ by axiom **E1** and $c \approx c$ holds for any $c \in C$.

Symmetry condition " if $c \approx d$, then $d \approx c$ " holds by axiom **E2**. Assume $c \approx d$, by definition $v(c = d)) = T$. By axiom **E2**

$$v^*((c = d \Rightarrow d = c)) = v(c = d) \Rightarrow v(d = c) = T,$$

i.e. $T \Rightarrow v(d = c) = T$. This is possible only if $v(d = c) = T$. This proves that $d \approx c$.

We prove transitivity in a similar way. Assume now that $c \approx d$ and $d \approx e$. We check to see that $c \approx e$. By the axiom **E3** we have that

$$v^*(((c = d \cap d = e) \Rightarrow c = e)) = T.$$

Since $v(c = d)) = T$ and $v(d = e)) = T$ by $c \approx d$ and $d \approx e$,

$$v^*((c = d \cap d = e) \Rightarrow c = e) = (T \cap T \Rightarrow c = e) = (T \Rightarrow c = e) = T,$$

we get that $v(c = e) = T$ and hence $d \approx e$.
We denote by $[c]$ the equivalence class of c and we define the universe M of $\mathcal{M}$ as

$$M = \{[c] : \ c \in C\}. \tag{9.14}$$

(2.) **Definition** of $R_I \subseteq M^n$.

Let M be given by (9.14). We define

$$([c_1], [c_2], \ \ldots, [c_n]) \in R_I \quad \text{if and only if} \quad v(R(c_1, c_2, \ \ldots, c_n)) = T. \tag{9.15}$$

We have to prove now that R_I is well defined by the condition (9.15). To do so we must check

$$\text{if } \ [c_1] = [d_1], [c_2] = [d_2], \ \ldots, [c_n] = [d_n] \ \text{ and } \ ([c_1], [c_2], \ \ldots, [c_n]) \in R_I,$$

$$\text{then} \quad ([d_1], [d_2], \ \ldots, [d_n]) \in R_I.$$

We have by the axiom **E4** that

$$v^*(((c_1 = d_1 \cap ... \cap c_n = d_n) \Rightarrow (R(c_1, ..., c_n) \Rightarrow R(d_1, ..., d_n)))) = T. \tag{9.16}$$

By the assumption $[c_1] = [d_1],\ \ldots, [c_n] = [d_n]$ we have that $v(c_1 = d_1) = T, \ldots, v(c_n = d_n) = T$. By the assumption $([c_1], [c_2],\ \ldots, [c_n]) \in R_I$, we have that $v(R(c_1, ..., c_n)) = T$. Hence the condition (9.16) becomes

$$(T \Rightarrow (T \Rightarrow v(R(d_1, ..., d_n)))) = T.$$

It holds only when $v(R(d_1, ..., d_n)) = T$ and by (9.15) we proved that

$$([d_1], [d_2],\ \ldots, [d_n]) \in R_I.$$

(3.) **Definition** of $f_I :\ M^n \to M$.

Let $c_1,\ c_2,\ \ldots,\ c_n \in C$ and $f \in \mathbf{F}$. We claim that there is $c \in C$ such that $f(c_1,\ c_2,\ \ldots,\ c_n) = c$ and $v(f(c_1,\ c_2,\ \ldots,\ c_n) = c) = T$.

For consider the formula $A(x)$ given by $f(c_1,\ c_2,\ \ldots,\ c_n) = x$. If $v^*(\exists x A(x)) = v^*(f(c_1,\ c_2,\ \ldots,\ c_n) = x) = T$, we want to prove $v^*(A(c_{A[x]})) = T$, i.e.

$$v(f(c_1,\ c_2,\ \ldots,\ c_n) = c_A) = T.$$

So suppose that $v(f(c_1,\ c_2,\ \ldots,\ c_n) = c_A) = F$. But one member of he Henkin set S_{Henkin} (definition 9.16) is the sentence $(A(f(c_1,\ c_2,\ \ldots,\ c_n)) \Rightarrow \exists x A(x))$ so we must have that $v^*(A(f(c_1,\ c_2,\ \ldots,\ c_n))) = F$. But this says that v assigns F to the atomic sentence $f(c_1,\ c_2,\ \ldots,\ c_n) = f(c_1,\ c_2,\ \ldots,\ c_n)$, i.e. By the axiom **E1** $v(c_i = c_i) = T$ for $i = 1, 2 \ldots n$ and by **E5**

$$(v^*(c_1 = c_1 \cap \ldots\ c_n = c_n) \Rightarrow v^*(f(c_1,\ \ldots,\ c_n) = f(c_1, \ldots, c_n))) = T.$$

This means that $T \Rightarrow F = T$ and this contradiction proves there is $c \in C$ such that $f(c_1,\ c_2,\ \ldots,\ c_n) = c$ and $v(f(c_1,\ c_2,\ \ldots,\ c_n) = c) = T$. We can hence define

$$f_I(([c_1], \ldots, [c_n]) = [c] \text{ for } c \text{ such that } v(f(c_1, \ldots,\ c_n) = c) = T. \tag{9.17}$$

The argument similar to the one used in (2.) proves that f_I is well defined.

(4.) **Definition** of $c_I \in M$.

For any $c \in C$ we take $c_I = [c]$. If $d \in \mathbf{C}$, then an argument similar to that used on (3.) shows that there is $c \in C$ such that $v(d = c) = T$, i.e. $d \approx c$, so we put $d_I = [c]$.

This completes the construction of the canonical structure $\mathcal{M} = [M, I]$ and guarantees that (9.10) holds for for all **atomic propositional** sentences (definition 9.2), i.e. we proved that

$$\mathcal{M} \models B \ \text{ if and only if } \ v^*(B) = T, \text{ for sentences } B \in \mathcal{P}.$$

To complete the proof of the Lemma 9.2 we prove that the property (9.10) holds for the canonical structure $\mathcal{M} = [M, I]$ defined above and all other sentences. We carry the proof by induction on length of formulas. The case of propositional connectives is trivial. For example, $\mathcal{M} \models (A \cap B)$ if and only if $\mathcal{M} \models A$ and $\mathcal{M} \models B)$ (follows directly from the satisfaction definition) if and only if $v^*(A) = T$ and $v^*(B) = T$ (by the induction hypothesis) if and only if $v^*(A \cap B) = T$. We proved

$$\mathcal{M} \models (A \cap B) \text{ if and only if } v^*(A \cap B) = T,$$

for all sentences A, B of $\mathcal{L}(C)$. The proof for all other connectives is similar.

We prove now the case of a sentence B of the form $\exists x A(x)$, i.e. we want to show that

$$\mathcal{M} \models \exists x A(x) \quad \text{if and only if} \quad v^*(\exists x A(x)) = T. \tag{9.18}$$

$v^*(\exists x A(x)) = T$. Then there is a c such that $v^*(A(c) = T$, so by induction hypothesis, $\mathcal{M} \models A(c)$ so $\mathcal{M} \models \exists x A(x)$.

On the other hand, if $v^*(\exists x A(x)) = F$, then by $S_{Henking}$ quantifier axiom **Q2** (definition 9.15) we have that $v^*(A(t)) = F$ for all closed terms t of $\mathcal{L}(C)$. In particular, for every $c \in C$ $v^*(A(c)) = F$. By induction hypothesis, $\mathcal{M} \models \neg A(c)$, for all $c \in C$. Since every element of M is denoted by some $c \in C$, $\mathcal{M} \models \neg\exists x A(x)$. Thus we proved (9.18).

The proof of the case of a sentence B of the form $\forall x A(x)$ is similar and is left to the reader.

The Reduction to Propositional Logic Theorem 9.2 provides not only a method of constructing models of theories out of symbols, but also gives us immediate proofs of the Compactness Theorem 9.3 for the predicate logic and Lowenheim-Skolem Theorem 9.4.

Theorem 9.3 (Compactness theorem for the predicate logic)

Let S be an y set of predicate formulas of $\mathcal{L}$.
The set S has a model if and only if any finite subset S_0 of S has a model.

Proof
Let S be a set of predicate formulas such that every finite subset S_0 of S has a model. We need to show that S has a model. By the implication $(iii) \rightarrow (i)$ of the Theorem 9.2 this is equivalent to proving that $S \cup S_{Henkin} \cup EQ$ is consistent in the sense of propositional logic. By the Compactness Theorem 9.1 for propositional logic of $\mathcal{L}$, it suffices to prove that for every finite subset $S_0 \subset S$, $S_0 \cup S_{Henkin} \cup EQ$ is consistent, which follows from the hypothesis and the implication $(i) \rightarrow (iii)$ of the Reduction to Propositional Logic Theorem 9.2.

Theorem 9.4 (Löwenheim-Skolem Theorem)

Let κ be an infinite cardinal and let Γ be a set of at most κ formulas of the first order language.

If the set S has a model, then there is a model $\mathcal{M} = [M, I]$ of S such that $cardM \leq \kappa$.

Proof Let $\mathcal{L}$ be a predicate language with the alphabet $\mathcal{A}$ such that $card(\mathcal{A}) \leq \kappa$. Obviously, $card(\mathcal{F}) \leq \kappa$. By the definition of the witnessing expansion $\mathcal{L}(C)$ of $\mathcal{L}$, $C = \bigcup_n C_n$ and for each n, $card(C_n) \leq \kappa$. So also $cardC \leq \kappa$. Thus any canonical structure for $\mathcal{L}(C)$ has $\leq \kappa$ elements. By the implication $(i) \rightarrow (ii)$ of the Reduction to Propositional Logic Lemma 9.2 there is a model of S (canonical structure) with $\leq \kappa$ elements.

9.2 Proof of Completeness Theorem

The proof of Gödel's completeness theorem given by Kurt Gdel in his doctoral dissertation of 1929 (and a rewritten version of the dissertation, published as an article in 1930) is not easy to read today; it uses concepts and formalism that are no longer used and terminology that is often obscure. The version given below attempts to represent all the steps in the proof and all the important ideas faithfully, while restating the proof in the modern language of mathematical logic. This outline should not be considered a rigorous proof of the theorem.

It was first proved by Kurt Gödel in 1929. It was then simplified in 1947, when Leon Henkin observed in his Ph.D. thesis that the hard part of the proof can be presented as the Model Existence Theorem (published in 1949). Henkin's proof was simplified by Gisbert Hasenjaeger in 1953. Other now classical proofs has been published by Rasiowa and Sikorski (1951 1952) using Boolean algebraic methods and by by Beth (1953), using topological methods. Still other proofs may be found in Hintikka(1955) and in Beth(1959).

Hilbert-style Proof System H

Language $\mathcal{L}$

The language $\mathcal{L}$ of the proof system **H** is a predicate (first order) language with equality (definition 9.12). We assume that the sets **P, F, C** are infinitely enumerable. We also assume that it has a full set of propositional connectives, i.e.

$$\mathcal{L} = \mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow\}}(\mathbf{P}, \mathbf{F}, \mathbf{C}). \tag{9.19}$$

Logical Axioms LA

The set LA of logical axioms consists of three groups of axioms: propositional axioms PA, equality axioms EA, and quantifiers axioms QA. we write it symbolically as

$$LA = \{PA,\ EA,\ QA\}.$$

For the set PA of *propositional axioms* we choose any complete set of axioms for propositional logic with a full set $\{\neg, \cap, \cap, \Rightarrow\}$ of propositional connectives.

In some formalizations, including the one in the *Handbook of Mathematical Logic, Barwise, ed.* (1977) we base our system **H** on, the authors just say for this group of axioms: "all tautologies". They of course mean all predicate formulas of $\mathcal{L}$ that are substitutions of propositional tautologies. This is done for the need of being able to use freely these predicate substitutions of propositional tautologies in the proof of completeness theorem for the proof system they formalize this way.

In this case these tautologies are listed as axioms of the system and hence are provable in it. This is a convenient approach, but also the one that makes such a proof system not to be finately axiomatizable.

We avoid the infinite axiomatization by choosing a proper finite set of predicate language version of propositional axioms that is known (proved already for propositional case) to be complete, i.e. the one in which all propositional tautologies are provable.

We choose, for name **H** (Hilbert) and historical sake, the set of Hilbert (1928) axioms from chapter 5.

.For the set EA of *equational axioms* we choose the same set (9.5) as in section 9.1.1 because they were used in the proof of Reduction to Propositional Logic Theorem 9.2 and we want to be able to carry this proof within the system **H**.

For the set QA of *quantifiers axioms* we choose the axioms such that the Henkin set S_{Henkin} axioms **Q1, Q2** are their particular cases, so again a proof of the Reduction to Propositional Logic Theorem 9.2 can be carried within **H**.

Rules of inference $\mathcal{R}$

There are three inference rules: Modus Ponens (MP) and two quantifiers rules $(G), (G1), (G2)$, called Generalization Rules.

We define the proof system **H** as follows.

$$\mathbf{H} = (\mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow\}}(\mathbf{P},\mathbf{F},\mathbf{C}),\ \mathcal{F},\ LA,\ \ \mathcal{R} = \{(MP),\ (G),(G1),\ (G2)\}), \quad (9.20)$$

where

$\mathcal{L} = \mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow\}}(\mathbf{P},\mathbf{F},\mathbf{C})$ is predicate (first order) language with equality (definition 9.12). We assume that the sets **P, F, C** are infinitely enumerable.

$\mathcal{F}$ is the set of all well formed formulas of $\mathcal{L}$.

LA is the set of logical axioms and

$$LA = \{PA, EA, QA\} \tag{9.21}$$

for PA, EA, QA defined as follows.

PA is the set of **propositional axioms** (Hilbert, 1928)

A1 $(A \Rightarrow A)$,

A2 $(A \Rightarrow (B \Rightarrow A))$,

A3 $((A \Rightarrow B) \Rightarrow ((B \Rightarrow C) \Rightarrow (A \Rightarrow C)))$,

A4 $((A \Rightarrow (A \Rightarrow B)) \Rightarrow (A \Rightarrow B))$,

A5 $((A \Rightarrow (B \Rightarrow C)) \Rightarrow (B \Rightarrow (A \Rightarrow C)))$,

A6 $((A \Rightarrow B) \Rightarrow ((B \Rightarrow C) \Rightarrow (A \Rightarrow C)))$,

A7 $((A \cap B) \Rightarrow A)$,

A8 $((A \cap B) \Rightarrow B)$,

A9 $((A \Rightarrow B) \Rightarrow ((A \Rightarrow C) \Rightarrow (A \Rightarrow (B \cap C)))$,

A10 $(A \Rightarrow (A \cup B))$,

A11 $(B \Rightarrow (A \cup B))$,

A12 $((A \Rightarrow C) \Rightarrow ((B \Rightarrow C) \Rightarrow ((A \cup B) \Rightarrow C)))$,

A13 $((A \Rightarrow B) \Rightarrow ((A \Rightarrow \neg B) \Rightarrow \neg A))$,

A14 $(\neg A \Rightarrow (A \Rightarrow B))$,

A15 $(A \cup \neg A)$,

for any $A, B, C \in \mathcal{F}$.

EA is the set of **equality axioms**.

E1 $u = u$,

E2 $(u = w \Rightarrow w = u)$,

E3 $((u_1 = u_2 \cap u_2 = u_3) \Rightarrow u_1 = u_3)$,

E4 $((u_1 = w_1 \cap ... \cap u_n = w_n) \Rightarrow (R(u_1, ..., u_n) \Rightarrow R(w_1, ..., w_n)))$,

E5 $((u_1 = w_1 \cap ... \cap u_n = w_n) \Rightarrow (t(u_1, ..., u_n) \Rightarrow t(w_1, ..., w_n)))$,

for any free variable or constant of $\mathcal{L}$, $R \in \mathbf{P}$, and $t \in \mathbf{T}$, where R is an arbitrary n-ary relation symbol of $\mathcal{L}$ and $t \in \mathbf{T}$ is an arbitrary n-ary term of $\mathcal{L}$.

QA is the set of **quantifiers axioms**.

Q1 $(\forall x A(x) \Rightarrow A(t))$,

Q2 $(A(t) \Rightarrow \exists x A(x))$,

where where t is a term, A(t) is a result of substitution of t for all free occurrences of x in $A(x)$, and t is *free for* x in A(x), i.e. no occurrence of a variable in t becomes a bound occurrence in A(t).

$\mathcal{R}$ is the set of **rules of inference**,

$$\mathcal{R} = \{(MP),\ (G),\ (G1),\ (G2)\},$$

where

(MP) is Modus Ponens rule

$$(MP)\ \ \frac{A\ ;\ (A \Rightarrow B)}{B},$$

for any formulas $A, B \in \mathcal{F}$.

$(G), (G1), (G2)$ are the following **quantifiers generalization rules**.

$$(G)\ \ \frac{A}{\forall x A},$$

where $A \in \mathcal{F}$. In particular we write

$$(G)\ \ \frac{A(x)}{\forall x A(x)}$$

for $A(x) \in \mathcal{F}$ and $x \in VAR$.

$$(G1)\ \ \frac{(B \Rightarrow A(x))}{(B \Rightarrow \forall x A(x))},$$

where $A(x), B \in \mathcal{F}$, $x \in VAR$, and B is such that x is not free in B.

$$(G2)\ \ \frac{(A(x) \Rightarrow B)}{(\exists x A(x) \Rightarrow B)},$$

where $A(x), B \in \mathcal{F}$, $x \in VAR$, and B is such that x is not free in B.

We define, as we do for any proof system, a notion of a *proof of a formula A from a set S* of formulas in **H** as a finite sequence of formulas $B_1, B_2, \ \ldots\ B_n$, with $B_n = A$, each of which is either a logical axiom of **H**, a member of S, or else follows from earlier formulas in the sequence by one of the inference rules from $\mathcal{R}$. We write it formally as follows.

Definition 9.18 (Proof from Γ)

Let $\Gamma \subseteq \mathcal{F}$ be any set of formulas of $\mathcal{L}$. A proof in **H** *of a formula $A \in \mathcal{F}$ from s set Γ of formulas is a sequence*

$$B_1, B_2, \ \ldots \ B_n$$

of formulas, such that

$$B_1 \in LA \cup \Gamma, \qquad B_n = A$$

and for each i, $1 < i \leq n$, either $B_i \in LA \cup \Gamma$ or B_i is a conclusion of some of the preceding expressions in the sequence $B_1, B_2, \ \ldots \ B_n$ by virtue of one of the rules of inference from $\mathcal{R}$.

We write

$$\Gamma \vdash_{\mathbf{H}} A$$

to denote that the formula A has a proof from Γ in **H** *and*

$$\Gamma \ \vdash \ A,$$

when the proof system **H** *is fixed.*

The case when $\Gamma = \emptyset$ is a special one. By the definition 9.18, $\emptyset \vdash_{\mathbf{H}} A$ means that in the proof of A only logical axioms LA are used. We hence write

$$\vdash_{\mathbf{H}} A$$

to denote that a formula A has a proof in **H**.

As we work with a fixed (and only one) proof system, we use the notation

$$\Gamma \ \vdash \ A \ \text{ and } \ \vdash \ A$$

to denote the proof of a formula A from a set Γ and proof of a formula A in **H**, respectively.

Any proof of the completeness theorem for a given proof system consists always of two parts. First we have show that *all formulas that have a proof in the system are tautologies.* This implication is called a soundness theorem, or soundness part of the completeness theorem.

The second implication says: *if a formula is a tautology then it has a proof in the proof system.* This alone is sometimes called a completeness theorem (on assumption that the system is sound). Traditionally it is called a completeness part of the completeness theorem.

We know that all axioms of **H** are predicate tautologies (proved in chapter 8 and all rules of inference from $\mathcal{R}$ are sound as the corresponding formulas were also proved in chapter 8 to be predicate tautologies and so the system **H** is sound, i.e. the following holds for **H**.

Theorem 9.5 (Soundness Theorem)

For every formula $A \in \mathcal{F}$ of the language $\mathcal{L}$ of the proof system **H***,*

$$\textit{if } \vdash A, \quad \textit{then} \quad \models A.$$

The soundness theorem proves that the proofs in the system "produce" only tautologies. We show here, as the next step that our proof system **H** "produces" not only tautologies, but that all tautologies are provable in it.

This is called a *completeness theorem* for classical predicate (first order logic, as it all is proven with respect to classical semantics. This is why it is called a completeness of predicate logic theorem.

The goal is now to prove the completeness part of the following.

Theorem 9.6 (Gödel Completeness of Predicate Logic)

For any formula A of the language $\mathcal{L}$ of the system **H***,*
A is provable in **H** *if and only if A is a predicate tautology (valid).*
We write it symbolically as

$$\vdash A \quad \textit{if and only if} \quad \models A. \tag{9.22}$$

We are going to prove the Gödel' s Theorem 9.6 as a particular case of Theorem 9.7 that follows. It is its more general, and more modern version. This version, as well as the method of proving it, was first introduced by Henkin in 1947. It became with its consequent improvements, as classical as the Gödel's own. It uses the notion of a *logical implication*, and some other notions. We introduce them below.

Definition 9.19 (Sentence, Closure)

Any formula of $\mathcal{L}$ without free variables is called a **sentence***.*
For any formula $A(x_1, \dots x_n)$, a sentence

$$\forall x_1 \forall x_2 \dots \forall x_n \; A(x_1, \dots x_n)$$

is called a **closure** *of $A(x_1, \dots x_n)$.*

Directly from the definition 10.16 have that the following hold.

Fact 9.2 *For any formula $A(x_1, \dots x_n)$,*

$$\models A(x_1, \dots x_n) \quad \textit{if and only if} \quad \models \forall x_1 \forall x_2 \dots \forall x_n \; A(x_1, \dots x_n).$$

Definition 9.20 (Logical Implication)

For any set $\Gamma \subseteq \mathcal{F}$ of formulas of $\mathcal{L}$ and any $A \in \mathcal{F}$, we say that the set Γ **logically implies** *the formula A and write it as $\Gamma \models A$ if and only if all models of Γ are models of A.*

Observe, that in order to prove that $\Gamma \models B$ we have to show that the implication

$$\text{if } \ \mathcal{M} \models \Gamma \ \text{ then } \ \mathcal{M} \models B$$

holds for all structures $\mathcal{M} = [U, I]$ for $\mathcal{L}$.

Lemma 9.4 *Let Γ be a set of sentences of $\mathcal{L}$, for any formula $A(x_1, \dots x_n)$ that is not a sentence,*

$$\Gamma \vdash A(x_1, \dots x_n) \quad \textit{if and only if} \quad \Gamma \models \forall x_1 \forall x_2 \dots \forall x_n \ A(x_1, \dots x_n).$$

Proof
Let $B_1, B_2, \ \dots \ B_n$ be a proof of $A(x_1, \dots x_n)$ from Γ and let $\mathcal{M}$ be a model of Γ. We use Fact 9.2 and prove by induction on n, that $\mathcal{M} \models \forall x_1 \forall x_2 \dots \forall x_n \ B_n(x_1, \dots x_n)$, and hence $\mathcal{M} \models \forall x_1 \forall x_2 \dots \forall x_n \ A(x_1, \dots x_n)$. The converse implication is obvious.

Fact 9.2 and Lemma 9.4 show that we need to consider only sentences (closed formulas) of $\mathcal{L}$, since a formula of $\mathcal{F}$ is a tautology if and only if its closure is a tautology and is provable from Γ if and only if its closure is provable from Γ. This justifies the following generalization of the original Gödel' s completeness of predicate logic Theorem 9.6.

Theorem 9.7 (Gödel Completeness Theorem)

Let Γ be any set of sentences and A any sentence of a language $\mathcal{L}$ of Hilbert proof system **H**.
A sentence A is provable from Γ in **H** *if and only if the set Γ* **logically implies** *A.*
We write it in symbols,

$$\Gamma \vdash A \quad \textit{if and only if} \quad \Gamma \models A. \tag{9.23}$$

Remark
We want to remind the readers that the Reduction Predicate Logic to Propositional Logic Section 9.1 is an integral and the first part of the proof the Gödel Completeness Theorem 9.7. We presented it separately for two reasons.

R1. The reduction method and theorems and their proofs are purely semantical in their nature and hence are independent of the proof system **H**.

R2. Because of **R1.** the reduction method can be used/adapted to a proof of completeness theorem of any other proof system one needs to prove the classical completeness theorem for. See section **??**.

In order to prove it we must formulate it properly so we need to introduce few new important and classical notion and prove some lemmas needed for the proof. The first is the notion of **consistency**.

There are **two definitions of consistency**; semantical and syntactical. The **semantical** one uses definition the notion of a model and says, in plain English: *a set of formulas is consistent if it has a model.*

The **syntactical** one uses the notion of provability and says: *a set of formulas is consistent if one can't prove a contradiction from it.*

We have used, in the Proof Two of the Completeness Theorem for propositional logic (chapter 5) the **syntactical** definition of consistency. We use here the following **semantical** definition.

Definition 9.21 (Consistent/Inconsistent)

A set $\Gamma \subseteq \mathcal{F}$ of formulas of $\mathcal{L}$ is **consistent** *if and only if it has a model, otherwise, it is* **inconsistent**.

Directly from the above definitions we have the following.

Lemma 9.5

For any set $\Gamma \subseteq \mathcal{F}$ of formulas of $\mathcal{L}$ and any $A \in \mathcal{F}$,

if $\Gamma \models A$, then the set $\Gamma \cup \{\neg A\}$ is **inconsistent**.

Proof
Assume $\Gamma \models A$ and $\Gamma \cup \{\neg A\}$ is consistent. By definition 9.21 there is a structure $\mathcal{M} = [U, I]$, such that $\mathcal{M} \models \Gamma$ and $\mathcal{M} \models \neg A$, i.e. $\mathcal{M} \not\models A$. This is a contradiction with $\Gamma \models A$.

Now we are going to prove the following Lemma 9.6 that is crucial, together with the Reduction to Propositional Logic Theorem 9.2 and the above Lemma 9.5 to the proof of the Completeness Theorem 9.7.

Lemma 9.6

Let Γ be any set of sentences of a language $\mathcal{L}$ of Hilbert proof system **H**.
The following conditions hold For any formulas $A, B \in \mathcal{F}$ of $\mathcal{L}$.

(i) If $\Gamma \vdash (A \Rightarrow B)$ and $\Gamma \vdash (\neg A \Rightarrow B)$, then $\Gamma \vdash B$.

(ii) If $\Gamma \vdash ((A \Rightarrow C) \Rightarrow B)$, then $\Gamma \vdash (\neg A \Rightarrow B)$ and $\Gamma \vdash (C \Rightarrow B)$.

(iii) If x does not appear in B and if $\Gamma \vdash ((\exists y A(y) \Rightarrow A(x)) \Rightarrow B)$, then $\Gamma \vdash B$.

(iv) If x does not appear in B and if $\Gamma \vdash ((A(x) \Rightarrow \forall y A(y)) \Rightarrow B)$, then $\Gamma \vdash B$.

Proof
(i) Notice that the formula $((A \Rightarrow B) \Rightarrow ((\neg A \Rightarrow B) \Rightarrow B))$ is a substitution

of a propositional tautology, hence by definition of **H**, is provable in it. By monotonicity,

$$\Gamma \vdash ((A \Rightarrow B) \Rightarrow ((\neg A \Rightarrow B) \Rightarrow B)).$$

By assuption $\Gamma \vdash (A \Rightarrow B)$ and Modus Ponens we get

$$\Gamma \vdash ((\neg A \Rightarrow B) \Rightarrow B).$$

By assuption $S\Gamma \vdash (\neg A \Rightarrow B)$ and Modus Ponens we get $\Gamma \vdash B$.

(ii) The formulas (1) $(((A \Rightarrow B) \Rightarrow (\neg A \Rightarrow B)))$ and (2) $(((A \Rightarrow B) \Rightarrow B) \Rightarrow (C \Rightarrow B))$ are substitution of a propositional tautologies, hence are provable in **H**. Assume $\Gamma \vdash ((A \Rightarrow C) \Rightarrow B)$. By monotonicity and (1) we get $\Gamma \vdash (\neg A \Rightarrow B)$ and by (2) we get $\vdash (C \Rightarrow B)$.

(iii) Assume $\Gamma \vdash ((\exists y A(y) \Rightarrow A(x)) \Rightarrow B)$. Observe that it is a particular case of assumption $\Gamma \vdash ((A \Rightarrow C) \Rightarrow B)$ in (ii), for $A = \exists y A(y),\ C = A(x), B = B$. Hence by (ii) we have that $\Gamma \vdash (\neg\exists y A(y) \Rightarrow B)$ and $\Gamma \vdash (A(x) \Rightarrow B)$.

Apply Generalization Rule G2 to $\Gamma \vdash (A(x) \Rightarrow B)$ and we have $\Gamma \vdash (\exists y A(y) \Rightarrow B.)$ Then by (i) applied to $\Gamma \vdash (\exists y A(y) \Rightarrow B)$ and $\Gamma \vdash (\neg\exists y A(y) \Rightarrow B)$ we get $\Gamma \vdash B$.

The proof of (iv) is similar to (iii), but uses the Generalization Rule G1.This **ends the proof** of the lemma.

Now we are ready to conduct the proof of the Completeness Theorem for **H**. There are two versions. Theorem 9.7 that is Gödel original formulation and the one we used in previous chapters of the book. It follows from theorem **??**). We put them both together as follows.

Theorem 9.8 (H Completeness)

Let Γ be any set of sentences and A any sentence of a language $\mathcal{L}$ of Hilbert proof system **H**.

$$\Gamma \vdash A \quad \textit{if and only if} \quad \Gamma \models A. \tag{9.24}$$

In particular, for any formula A of $\mathcal{L}$,

$$\vdash A \quad \textit{if and only if} \quad \models A. \tag{9.25}$$

Proof
We first prove the completeness part (9.24), i.e. we prove the implication

$$\text{if } \ \Gamma \models A, \ \text{then } \ \Gamma \vdash A. \tag{9.26}$$

Suppose that $\Gamma \models A$, i.e. we assume that all $\mathcal{L}$ models of Γ are models of A. By Lemma 9.5 the set $\Gamma \cup \{\neg A\}$ is inconsistent.

Let $\mathcal{M} \models \Gamma$. We construct, as a next step, a witnessing expansion language $\mathcal{L}(C)$ of $\mathcal{L}$ (definition 9.13). By the Reduction to Propositional Logic Theorem 9.2, the set $\Gamma \cup S_{Henkin} \cup EQ$ is consistent in a sense of propositional logic in $\mathcal{L}$. The set S_{Henkin} is a Henkin Set (definition 9.16) and **EQ** are equality axioms (9.5) that are also the equality axioms EQ of **H**.

By the Compactness Theorem 9.1 for propositional logic of $\mathcal{L}$ there is a finite set $S_0 \subseteq \Gamma \cup S_{Henkin} \cup EQ$ such that $S_0 \cup \{\neg A\}$ is inconsistent in the sense of propositional logic.

We list all elements of S_0 in a sequence

$$A_1,\ A_2, \ldots, A_n,\ B_1,\ B_2, \ldots, B_m \tag{9.27}$$

as follows. The sequence $A_1,\ A_2, \ldots, A_n$ consists of those elements of S_0 which are either in $\Gamma \cup EQ$ or else are quantifiers axioms (definition 9.15) that are particular cases of the quantifiers axioms QA of **H**. We list them in any order.

The sequence $B_1,\ B_2, \ldots, B_m$ consists of elements of S_0 which are Henkin Axioms (definition 9.14) but listed carefully as to be described as follows. Observe that by definition 9.13,

$$\mathcal{L}(C) = \bigcup_{n \in N} \mathcal{L}_n, \text{ for } \mathcal{L} = \mathcal{L}_0 \subseteq \mathcal{L}_1 \subseteq \ \ldots.$$

We define the *rank* of $A \in \mathcal{L}(C)$ to be the least n, such that $A \in \mathcal{L}_n$.

Now we choose for B_1 a Henkin Axiom in S_0 of the *maximum rank.*
We choose for B_1 a Henkin Axiom in $S_0 - \{B_1\}$ of the *maximum rank.*
We choose for B_2 a Henkin Axiom in $S_0 - \{B_1, B_2\}$ of the *maximum rank*, etc.
The point of choosing the formulas B_i's in this way is to make sure that the witnessing constant about which B_i speaks, does not appear in $B_{i+1},\ B_{i+2},\ \ldots, B_m$.
For example, if B_1 is

$$(\exists x C(x) \Rightarrow C(c_{C[x]})),$$

then $C[x]$ does not appear in any of the other $B_2,\ \ldots, B_m$, by the maximality condition on B_1.

We know that that $S_0 \cup \{\neg A\}$ is inconsistent in the sense of propositional logic, i.e. it does not have a (propositional) model. This means that $v^*(\neg A) \neq T$ for all v and so $v^*(A) = T$ for all v. Hence a sentence

$$(A_1 \Rightarrow (A_2 \Rightarrow \ldots (A_n \Rightarrow (B_1 \Rightarrow \ldots (B_m \Rightarrow A))..)$$

is a propositional tautology.
We now replace each witnessing constant in this sentence by a distinct new variable and write the result as

$$({A_1}' \Rightarrow ({A_2}' \Rightarrow \ldots ({A_n}' \Rightarrow ({B_1}' \Rightarrow \ldots ({B_m}' \Rightarrow A))..)$$

. We have $A' = A$ since A has no witnessing constant in it. The result is still a tautology and hence is provable in **H** from propositional axioms PA and Modus Ponens. By monotonicity

$$S_0 \vdash (A_1{}' \Rightarrow (A_2{}' \Rightarrow \ldots (A_n{}' \Rightarrow (B_1{}' \Rightarrow \ldots (B_m{}' \Rightarrow A))..). \tag{9.28}$$

Each of $A_1{}', A_2{}', \ldots, A_n{}'$ is either a quantifiers axiom from QA of **H** or else in S_0, so

$$S_0 \vdash A_i{}' \ \text{ for all } \ 1 \leq i \leq n.$$

We apply Modus Ponens to the above and (9.28) n times and get

$$S_0 \vdash (B_1{}' \Rightarrow (B_2{}' \Rightarrow \ldots (B_m{}' \Rightarrow A))..). \tag{9.29}$$

For example, if $B_1{}'$ is $(\exists x C(x) \Rightarrow C(x))$, we have by (9.29)

$$S_0 \vdash ((\exists x C(x) \Rightarrow C(x)) \Rightarrow B). \tag{9.30}$$

for $B = (B_2{}' \Rightarrow \ldots (B_m{}' \Rightarrow A))..)$. By the Reduction to Propositional Logic Theorem 9.2 part (iii), we get $S_0 \vdash B$, i.e.

$$S_0 \vdash (B_2{}' \Rightarrow \ldots (B_m{}' \Rightarrow A))..). \tag{9.31}$$

If, for example, $B_2{}'$ is $(D(x) \Rightarrow \forall x D(x))$, we have by (9.30)

$$S_0 \vdash ((\exists x C(x) \Rightarrow C(x)) \Rightarrow D). \tag{9.32}$$

for $D = (B_3{}' \Rightarrow \ldots (B_m{}' \Rightarrow A))..)$. By the Reduction to Propositional Logic Theorem 9.2 part (iv), we get $S_0 \vdash D$, i.e.

$$S_0 \vdash (B_3{}' \Rightarrow \ldots (B_m{}' \Rightarrow A))..). \tag{9.33}$$

. We hence apply parts (iii) ad (iv) of Theorem 9.2 to successively remove all $B_1{}', ..., B_m{}'$ and obtain the proof of A from S_0.
This **ends the proof** that $\Gamma \vdash A$ and hence the proof of the completeness part of (9.24).

The soundness part of of (9.24), i.e. the implication

$$\text{if } \ \Gamma \vdash A, \ \text{ then } \ \Gamma \models A,$$

holds for any sentence A of $\mathcal{L}$ directly by Fact 9.2, Lemma 9.4, and Theorem 9.5.

The Theorem 9.6, as expressed by (9.25) follows from Fact 9.2, Lemma 9.4 as a case of (9.24) for $\Gamma = \emptyset$.

This **ends the proof** of Theorem 9.8 as well as Theorem 9.7, and the proof of the original Gödel Completeness of Predicate Logic Theorem 9.6.

9.3 Deduction Theorem

In mathematical arguments, one often assumes a statement A on the assumption (hypothesis) of some other statement B and then concludes that we have proved the implication "if A, then B". This reasoning is justified by the following theorem, called a Deduction Theorem. It was first formulated and proved for a certain Hilbert proof system S for the classical propositional logic by Herbrand in 1930 in a form stated below.

Theorem 9.9 (Deduction Theorem) *(Herbrand,1930)*

For any formulas A, B of the language of a propositional proof system S,

$$\textit{if} \quad A \vdash_S B, \quad \textit{then} \ \vdash_S (A \Rightarrow B).$$

In chapter 5 we formulated and proved the following, more general version of the Herbrand Theorem 9.10 for a very simple (two logical axioms and Modus Ponens) propositional proof system $H1$.

Theorem 9.10 (Deduction Theorem)

For any subset Γ of the set of formulas $\mathcal{F}$ of H_1 and for any formulas $A, B \in \mathcal{F}$,

$$\Gamma,\ A \vdash_{H_1} B \ \ if\ and\ only\ if \ \ \Gamma \vdash_{H_1} (A \Rightarrow B).$$

In particular,

$$A \vdash_{H_1} B \ \ if\ and\ only\ if \ \ \vdash_{H_1} (A \Rightarrow B).$$

A natural question arises: does deduction theorem holds for the predicate logic in general and for its proof system **H** we defined here?.

The Theorem 9.10 cannot be carried *directly* to the predicate logic, but it nevertheless holds with *some modifications.* Here is where the problem lays.

Fact 9.3 *Given the proof system (9.20), i.e.*
$\boldsymbol{H} = (\mathcal{L}(\mathbf{P}, \mathbf{F}, \mathbf{C}),\ \mathcal{F},\ LA,\ \ \mathcal{R} = \{(MP),\ (G), (G1),\ (G2)\})$.
For any formula $A(x) \in \mathcal{F}$,
$A(x) \vdash \forall x A(x)$, but it is not always the case that $\vdash (A(x) \Rightarrow \forall x A(x))$.

Proof
Obviously, $A(x) \vdash \forall x A(x)$ by Generalization rule (G). Let now $A(x)$ be an atomic formula $P(x)$. By the Completeness Theorem 9.6, $\vdash (P(x) \Rightarrow \forall x P(x))$ if and only if $\models (P(x) \Rightarrow \forall x P(x))$. Consider a structure $\mathcal{M} = [M, I]$, where M contains at least two elements c and d. We define $P_I \subseteq M$ as a property that holds only for c, i.e. $P_I = \{c\}$. Take any assignment of $\mathcal{L}$ in $\mathcal{M}$, i.e.

$s : VAR \longrightarrow M$. Then $(\mathcal{M}, s) \models P(x)$ only when $s(x) = c$ for all $x \in VAR$. $\mathcal{M} = [M, I]$ is a counter model for $(P(x) \Rightarrow \forall x P(x))$, as we found s such $(\mathcal{M}, s) \models P(x)$ and obviously $(\mathcal{M}, s) \not\models \forall x P(x)$. This proves that Deduction Theorem fails for $A(x)$ being an atomic formula $P(x)$.

The Fact 9.3 shows that the problem is with application of the generalization rule (G) to the formula $A \in \Gamma$. To handle this we introduce, after Mendelson(1987) the following notion.

Definition 9.22

Let A be one of formulas in Γ and let

$$B_1, B_2, ..., B_n \tag{9.34}$$

a deduction (proof of B_n from Γ, together with justification *at each step.*

We say that the formula B_i **depends upon** *A in the proof (9.34)*
if and only if
(1) B_i is A and the justification *for B_i is $B_i \in \Gamma$*
or
(2) B_i is justified *as direct consequence by MP or (G) of some preceding formulas in the sequence (9.34), where at least one of these preceding formulas* **depends upon** *A.*

Here is a deduction

$$B_1, B_2, \ldots, B_5 \tag{9.35}$$

showing that

$$A,\ (\forall x A \Rightarrow C).$$

B_1 A, Hyp

B_2 $\forall x A$, $B_1, (G)$

B_3 $(\forall x A \Rightarrow C)$, Hyp

B_4 C, MP on B_2, B_3

B_5 $\forall x C$. (G)

Observe that the formulas A, C may, or may not have x as a free variable.

Example 9.2

In the derivation (9.35)
B_1 depends upon A,
B_2 depends upon A,
B_3 depends upon $(\forall x A \Rightarrow C)$,
B_4 depends upon A and $(\forall x A \Rightarrow C)$,
B_5 depends upon A and $(\forall x A \Rightarrow C)$.

Lemma 9.7

If B does not depend upon A in a deduction showing that $\Gamma, A \vdash B$, then $\Gamma \vdash B$.

Proof
Let $B_1, B_2, \ldots, B_n = B$ be a deduction of B from Γ, A in which B does not depend upon A. we prove by Induction that $\Gamma \vdash B$. Assume that Lemma 9.7 holds for all deductions of the length less than n. If $B \in \Gamma$ or $B \in LA$, then $\Gamma \vdash B$. If B is a direct consequence of two preceding formulas, then, since B does not depend upon A , neither do theses preceding formulas. By inductive hypothesis, theses preceding formulas have a proof from Γ alone. Hence so does B.

Now we are ready to formulate and prove the Deduction Theorem 9.11 for predicate logic.

Theorem 9.11 (Deduction Theorem)

For any formulas A, B of the language of proof system **H** *the following holds.*

(1) Assume that in some deduction showing that

$$\Gamma, A \vdash B,$$

no application *of the generalization rule (G) to a formula that depends upon A* has *as its quantified variable a free variable of A. Then*

$$\Gamma \vdash (A \Rightarrow B).$$

(2) If $\Gamma \vdash (A \Rightarrow B)$, then $\Gamma, A \vdash B$.

Proof
The proof extends the proof of the Deduction Theorem for propositional logic from chapter 5. We adopt the propositional proof (for a different proof system) to the system **H** and adding the predicate case. For the sake of clarity and independence we write now the whole proof in all details.
(1) Assume that $\Gamma, A \vdash B$, i.e. that we have a formal proof

$$B_1, B_2, ..., B_n \tag{9.36}$$

of B from the set of formulas $\Gamma \cup \{A\}$. In order to prove that $\Gamma \vdash (A \Rightarrow B)$ we will prove the following a little bit stronger statement **S**.

$$\mathbf{S}: \quad \Gamma \vdash (A \Rightarrow B_i) \quad \text{for all } B_i \ \ (1 \leq i \leq n) \ \text{ in the proof (9.36) of } B.$$

Hence, in particular case, when $i = n$, we will obtain that also

$$\Gamma \vdash (A \Rightarrow B).$$

The proof of **S** is conducted by induction on i ($1 \leq i \leq n$).

Base Step $i = 1$.
When $i = 1$, it means that the formal proof (5.6) contains only one element B_1. By the definition of the formal proof from $\Gamma \cup \{A\}$, we have that $B_1 \in LA$, or $B_1 \in \Gamma$, or $B_1 = A$, i.e.

$$B_1 \in LA \cup \Gamma \cup \{A\}.$$

Here we have two cases.

Case 1. $B_1 \in LA \cup \Gamma$.
Observe that the formula is a particular case of A2 of **H**. By assumption $B_1 \in LA \cup \Gamma$, hence we get the required proof of $(A \Rightarrow B_1)$ from Γ by the following application of the Modus Ponens rule

$$(MP)\ \frac{B_1\ ;\ (B_1 \Rightarrow (A \Rightarrow B_1))}{(A \Rightarrow B_1)}.$$

Case 2. $B_1 = A$.
When $B_1 = A$, then to prove $\Gamma \vdash (A \Rightarrow B)$ means to prove $\Gamma \vdash (A \Rightarrow A)$. But $(A \Rightarrow A) \in LA$ (axiom A21of **H**), i.e. $\vdash (A \Rightarrow A)$. By the monotonicity of the consequence we have that $\Gamma \vdash (A \Rightarrow A)$. The above cases conclude the proof of the Base case $i = 1$.

Inductive step
Assume that $\Gamma \vdash (A \Rightarrow B_k)$ for all $k < i$, we will show that using this fact we can conclude that also $\Gamma \vdash (A \Rightarrow B_i)$.

Consider a formula B_i in the sequence 9.36. By the definition, $B_i \in LA \cup \Gamma \cup \{A\}$ or B_i follows by MP from certain B_j, B_m such that $j < m < i$. We have to consider again two cases.

Case 1. $B_i \in LA \cup \Gamma \cup \{A\}$.
The proof of $(A \Rightarrow B_i)$ from Γ in this case is obtained from the proof of the Base Step for $i = 1$ by replacement B_1 by B_i and will be omitted here as a straightforward repetition.

Case 2. B_i is a conclusion of MP.
If B_i is a conclusion of MP, then we must have two formulas B_j, B_m in the sequence 9.36 such that $j < i, m < i,\ j \neq m$ and

$$(MP)\ \frac{B_j\ ;\ B_m}{B_i}.$$

By the inductive assumption, the formulas B_j, B_m are such that

$$\Gamma \vdash (A \Rightarrow B_j) \tag{9.37}$$

and

$$\Gamma \vdash (A \Rightarrow B_m). \tag{9.38}$$

Moreover, by the definition of the Modus Ponens rule, the formula B_m has to have a form $(B_j \Rightarrow B_i)$, i.e. $B_m = (B_j \Rightarrow B_i)$, and the the inductive assumption (9.38) can be re-written as follows.

$$\Gamma \vdash (A \Rightarrow (B_j \Rightarrow B_i)), \quad for\ j < i. \tag{9.39}$$

Observe now that the formula

$$((A \Rightarrow (B_j \Rightarrow B_i)) \Rightarrow ((A \Rightarrow B_j) \Rightarrow (A \Rightarrow B_i)))$$

is a substitution of the axiom A3 of **H** and hence has a proof in **H**. By the monotonicity,

$$\Gamma \vdash ((A \Rightarrow (B_j \Rightarrow B_i)) \Rightarrow ((A \Rightarrow B_j) \Rightarrow (A \Rightarrow B_i))). \tag{9.40}$$

Applying the rule MP to formulas (9.40) and (9.39,) i.e. performing the following

$$(MP)\ \frac{(A \Rightarrow (B_j \Rightarrow B_i))\ ;\ ((A \Rightarrow (B_j \Rightarrow B_i)) \Rightarrow ((A \Rightarrow B_j) \Rightarrow (A \Rightarrow B_i)))}{((A \Rightarrow B_j) \Rightarrow (A \Rightarrow B_i))}$$

we get that also

$$\Gamma \vdash ((A \Rightarrow B_j) \Rightarrow (A \Rightarrow B_i)). \tag{9.41}$$

Applying again the rule MP to formulas 9.37 and 9.41, i.e. performing the following

$$(MP)\ \frac{(A \Rightarrow B_j)\ ;\ ((A \Rightarrow B_j) \Rightarrow (A \Rightarrow B_i))}{(A \Rightarrow B_i)}$$

we get that

$$\Gamma \vdash (A \Rightarrow B_i).$$

Finally, suppose that there is some $j < i$ such that B_i is $\forall x B_j$. By hypothesis $\Gamma \vdash B_j$ and either (i) B_j does not depend upon A or (ii) x is not free variable in A.
We have two cases (i) and (ii) to consider.

(i) If B_j does not depend upon A , then by Lemma 9.7 $\Gamma \vdash B_j$ and, consequently, by the generalization rule (G), $\Gamma \vdash \forall x B_j$. Thus $\Gamma \vdash B_i$.

Now, by hypothesis $\Gamma \vdash B_j$ and by axiom A2, $\vdash (B_i \Rightarrow (A \Rightarrow B_i))$. Applying MP we get $\Gamma \vdash A \Rightarrow B_i)$.

(ii) If x is not free variable in A, then, by Completeness Theorem 9.6 and $\models (\forall x(A \Rightarrow B_j) \Rightarrow (A \Rightarrow \forall x B_j))$ we have that $\vdash (\forall x(A \Rightarrow B_j) \Rightarrow (A \Rightarrow \forall x B_j))$
.

Since $\Gamma \vdash A \Rightarrow B_i)$, we get by the generalization rule (G), $\Gamma \vdash \forall x(A \Rightarrow B_j)$, and so, by MP, $\Gamma \vdash A \Rightarrow \forall x B_j)$; that is $\Gamma \vdash A \Rightarrow B_i)$.
This completes the induction and the case (1) holds for $i = n$.

(2) The proof of the implication

$$\text{if } \ \Gamma \vdash (A \Rightarrow B) \ \text{ then } \ \Gamma, A \vdash B$$

is straightforward. Assume that $\Gamma \vdash (A \Rightarrow B)$, hence by monotonicitywe have also that $\Gamma, A \ \vdash (A \Rightarrow B)$. Obviously, $\Gamma, A \ \vdash A$. Applying Modus Ponens to the above, we get the proof of B from $\{\Gamma, A\}$ i.e. we have proved that $\Gamma, \ A \vdash B$. This **ends the proof** of the Deduction Theorem for **H**.

9.4 Some other Axiomatizations

We present here some of most known, and historically important axiomatizations of classical predicate logic, i.e. the following Hilbert style proof systems.

1. Hilbert and Ackermann (1928)

D. Hilbert and W. Ackermann, *Grundzügen der Theoretischen Logik (Principles of Theoretical Logic)*, Springer - Verlag, 1928. The book grew from the courses on logic and foundations of mathematics Hilbert gave in years 1917-1922. He received help in writeup from Barnays and the material was put into the book by Ackermann and Hilbert. It was conceived as an introduction to mathematical logic and was followed by D. Hilbert and P. Bernays, *Grundzügen der Mathematik I,II.* Springer -Verlag, 1934, 1939.

Hilbert and Ackermann formulated and asked a question of the completeness for their deductive (proof) system. It was answered affirmatively by Kurt Gödel in 1929 with proof of his Completeness Theorem 9.6.

We define the Hilbert and Ackermann system **HA** following a pattern established for the **H** system (9.20). The original language use by Hilbert and Ackermann contained only negation $\neg$ and disjunction $\cup$ and so do we.

$$\mathbf{HA} = (\mathcal{L}_{\{\neg,\cup\}}(\mathbf{P}, \mathbf{F}, \mathbf{C}), \ \mathcal{F}, \ LA, \ \ \mathcal{R} = \{(MP), \ (SB), \ (G1), \ (G2)\}), \quad (9.42)$$

where the set LA of logical axioms is as follows.

Propositional Axioms

A1 $(\neg(A \cup A) \cup A)$, A2 $(\neg A \cup (A \cup B))$,

A3 $(\neg(A \cup B) \cup (B \cup A))$, A4 $(\neg(\neg B \cup C) \cup (\neg(A \cup B) \cup (A \cup C)))$, for any $A, B, C, \in \mathcal{F}$.

Quantifiers Axioms

Q1 $(\neg\forall x A(x) \cup A(x))$, Q2 $(\neg A(x) \cup \exists x A(x))$,

Q2 $(\neg A(x) \cup \exists x A(x))$,

for any $A(x) \in \mathcal{F}$.

Rules of Inference $\mathcal{R}$

(MP) is the Modus Ponens rule. It has, in the language $\mathcal{L}_{\{\neg,\cup\}}$, a form

$$(MP) \quad \frac{A \;;\quad (\neg A \cup B)}{B}.$$

. (SB) is a **substitution rule**

$$(SB) \quad \frac{A(x_1, x_2, \ \dots \ x_n)}{A(t_1, t_2, \ \dots \ t_n)},$$

where $A(x_1, x_2, \ \dots \ x_n) \in \mathcal{F}$ and $t_1, t_2, \ \dots \ t_n \in \mathbf{T}$.

$(G1), (G2)$ are **quantifiers generalization rules**.

$$(G1) \quad \frac{(\neg B \cup A(x))}{(\neg B \cup \forall x A(x))}, \qquad (G2) \quad \frac{(\neg A(x) \cup B)}{(\neg \exists x A(x) \cup B)},$$

where $A(x), B \in \mathcal{F}$ and B is such that x is not free in B.

The **HA** system is usually written now with the use of implication, i.e. as based on a language $\mathcal{L} = \mathcal{L}_{\{\neg,\Rightarrow\}}(\mathbf{P}, \mathbf{F}, \mathbf{C})$, i.e. as a proof system

$$\mathbf{HAI} = (\mathcal{L}_{\{\neg,\Rightarrow\}}(\mathbf{P}, \mathbf{F}, \mathbf{C}),\ \mathcal{F},\ LA,\ \mathcal{R} = \{(MP),\ (SB),\ (G1),\ (G2)\}), \quad (9.43)$$

where the set LA of logical axioms is as follows.

Propositional Axioms

A1 $((A \cup A) \Rightarrow A)$, A2 $(A \Rightarrow (A \cup B))$,

A3 $((A \cup B) \Rightarrow (B \cup A))$, A4 $((\neg B \cup C) \Rightarrow ((A \cup B) \Rightarrow (A \cup C)))$,

for any $A, B, C, \in \mathcal{F}$.

Quantifiers Axioms

Q1 $(\forall x A(x) \Rightarrow A(x))$, Q2 $(A(x) \Rightarrow \exists x A(x))$,

for any $A(x) \in \mathcal{F}$.

Rules of Inference $\mathcal{R}$

(MP) is Modus Ponens rule

$$(MP)\ \ \frac{A\ ;\ (A \Rightarrow B)}{B},$$

for any formulas $A, B \in \mathcal{F}$.

(SB) is a **substitution rule**

$$(SB)\ \ \frac{A(x_1, x_2, \ \dots\ x_n)}{A(t_1, t_2, \ \dots\ t_n)},$$

where $A(x_1, x_2, \ \dots\ x_n) \in \mathcal{F}$ and $t_1, t_2, \ \dots\ t_n \in \mathbf{T}$.

$(G1), (G2)$ are **quantifiers generalization rules**.

$$(G1)\ \ \frac{(B \Rightarrow A(x))}{(B \Rightarrow \forall x A(x))}, \qquad (G2)\ \ \frac{(A(x) \Rightarrow B)}{(\exists x A(x) \Rightarrow B)},$$

where $A(x), B \in \mathcal{F}$ and B is such that x is not free in B.

The form of the quantifiers axioms Q1, Q2, and quantifiers generalization rule (Q2) is due to Bernays.

2. Mendelson (1987)

Here is the first order logic proof system **HM** as introduced in the Elliott Mendelson's book *Introduction to Mathematical Logic*, hence the name. (1987). It is an generalization to the predicate language of the proof system H_2 for propositional logic defined and studied in Chapter 5.

$$\mathbf{HM} = (\mathcal{L}_{\{\neg, \cup\}}(\mathbf{P}, \mathbf{F}, \mathbf{C}),\ \mathcal{F},\ LA,\ \ \mathcal{R} = \{(MP),\ (G)\}). \tag{9.44}$$

Propositional Axioms

A1 $(A \Rightarrow (B \Rightarrow A))$,

A2 $((A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C)))$,

A3 $((\neg B \Rightarrow \neg A) \Rightarrow ((\neg B \Rightarrow A) \Rightarrow B)))$,

for any $A, B, C, \in \mathcal{F}$.

Quantifiers Axioms

Q1 $(\forall xA(x) \Rightarrow A(t))$,

where where t is a term, A(t) is a result of substitution of t for all free occurrences of x in $A(x)$, and t is *free for* x in A(x), i.e. no occurrence of a variable in t becomes a bound occurrence in A(t).

Q2 $(\forall x(B \Rightarrow A(x)) \Rightarrow (B \Rightarrow \forall xA(x)))$, where $A(x), B \in \mathcal{F}$ and B is such that x is not free in B.

Rules of Inference $\mathcal{R}$

(MP) is the Modus Ponens rule

$$(MP)\ \ \frac{A\ ;\ (A \Rightarrow B)}{B},$$

for any formulas $A, B \in \mathcal{F}$.

(G) is the **generalization rule**

$$(G)\ \ \frac{A(x)}{\forall xA(x)},$$

where $A(x) \in \mathcal{F}$ and $x \in VAR$.

Rasiowa-Sikorski (1950)

Rasiowa and Sikorski are the authors of the first algebraic proof of the Gödel completeness theorem ever given in 1950. Other algebraic proofs were later given by Rieger, Beth, Łos in 1951, and Scott in 1954.

Here is their original axiomatization.

$$RS = (\mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow\}}(\mathbf{P},\mathbf{F},\mathbf{C}),\ \mathcal{F},\ LA,\ \ \mathcal{R}). \tag{9.45}$$

Propositional Axioms

A1 $((A \Rightarrow B) \Rightarrow ((B \Rightarrow C) \Rightarrow (A \Rightarrow C)))$,

A2 $(A \Rightarrow (A \cup B))$,

A3 $(B \Rightarrow (A \cup B))$,

A4 $((A \Rightarrow C) \Rightarrow ((B \Rightarrow C) \Rightarrow ((A \cup B) \Rightarrow C)))$,

A5 $((A \cap B) \Rightarrow A)$,

A6 $((A \cap B) \Rightarrow B)$,

A7 $((C \Rightarrow A) \Rightarrow ((C \Rightarrow B) \Rightarrow (C \Rightarrow (A \cap B)))$, A8 $((A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \cap B) \Rightarrow C))$,

A9 $(((A \cap B) \Rightarrow C) \Rightarrow (A \Rightarrow (B \Rightarrow C))$,

A10 $(A \cap \neg A) \Rightarrow B)$,

A11 $((A \Rightarrow (A \cap \neg A)) \Rightarrow \neg A)$,

A12 $(A \cup \neg A)$,

for any $A, B, C \in \mathcal{F}$.

Rules of Inference $\mathcal{R} = \{(MP), (SB), (Q1), (Q2), (Q3), (Q4)\}$.

(MP) is **Modus Ponens** rule

$$(MP) \quad \frac{A \ ; \ (A \Rightarrow B)}{B},$$

for any formulas $A, B \in \mathcal{F}$.

(SB) is a **substitution rule**

$$(SB) \quad \frac{A(x_1, x_2, \ \ldots \ x_n)}{A(t_1, t_2, \ \ldots \ t_n)},$$

where $A(x_1, x_2, \ \ldots \ x_n) \in \mathcal{F}$ and $t_1, t_2, \ \ldots \ t_n \in \mathbf{T}$.

$(G1), (G2)$ are the following **quantifiers introduction rules**.

$$(G1) \quad \frac{(B \Rightarrow A(x))}{(B \Rightarrow \forall x A(x))}, \qquad (G2) \quad \frac{(A(x) \Rightarrow B)}{(\exists x A(x) \Rightarrow B)},$$

where $A(x), B \in \mathcal{F}$ and B is such that x is not free in B.

$(G3), (G3)$ are the following **quantifiers elimination rules**.

$$(G3) \quad \frac{(B \Rightarrow \forall x A(x))}{(B \Rightarrow A(x))}, \qquad (G4) \quad \frac{\exists x (A(x) \Rightarrow B)}{A(x) \Rightarrow B)},$$

where $A(x), B \in \mathcal{F}$ and B is such that x is not free in B.

The algebraic logic starts from purely logical considerations, abstracts from them, places them into a general algebraic contest, and makes use of other branches of mathematics such as topology, set theory, and functional analysis. For example, Rasiowa and Sikorski algebraic generalization of the completeness theorem for classical predicate logic is the following.

Theorem 9.12 (Rasiowa, Sikorski 1950)

For every formula A of the classical predicate calculus $S = \{\mathcal{L}, \mathcal{C}\}$ the following conditions are equivalent

i *A is derivable in RS;*

ii *A is valid in every realization of $\mathcal{L}$;*

iii *A is valid in every realization of $\mathcal{L}$ in any complete Boolean algebra;*

iv *A is valid in every realization of $\mathcal{L}$ in the field $B(X)$ of all subsets of any set $X \neq \emptyset$;*

v *A is valid in every semantic realization of $\mathcal{L}$ in any enumerable set;*

vi *there exists a non-degenerate Boolean algebra $\mathcal{A}$ and an infinite set J such that A is valid in every realization of $\mathcal{L}$ in J and $\mathcal{A}$;*

vii *$A_R(\mathbf{i}) = V$ for the canonical realization R of $\mathcal{L}$ in the Lindenbaum-Tarski algebra $\mathcal{LT}$ of S and the identity valuation $\mathbf{i}$;*

viii *A is a predicate tautology.*

9.5 Homework Problems

1. Prove that for any equality axioms (9.5) A and for every structure $\mathcal{M} = [M, I]$ and every $s : VAR \longrightarrow M$, $(\mathcal{M}, s) \models A$.
2. Let **H** be the proof system defined (9.20). Prove the following.

   (i) QA axioms Q1 and Q2 of **H** are predicate tautologies.

   (ii) The rules of inference (G), (G1), (G2) of **H** are sound.
3. A proof system S is *strongly sound* if for any rule of inference r of S, a conjunction of all premisses of r is logically equivalent with its conclusion.

   Show that the proof system **H** is not strongly sound.
4. Prove soundness theorem for Hilbert Ackerman system **HA** (9.42).
5. Given two proof systems S and K we say that S and K are equivalent and write it as $S \equiv K$ if they have the same sets of of theorems.
   Prove that **HA** $\equiv$ **HAI** for **HA** defined by (9.42) and **HAI** defined by (9.43)
6. We know that the Medndelson proof system **HM** defined by (9.44) is complete. Prove that **HM** $\equiv$ **H**, where **H** be the proof system defined (9.20).

7. Let **RSE** be a proof system obtained from **RS** system defined by (9.45) by changing the language $\mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow\}}(\mathbf{P},\mathbf{F},\mathbf{C})$ of **RS** to the language with equality (definition 9.12) and adding Eguality Axioms (9.5) to the set LA of logocal axioms of **RS**. Prove Completeness Theorem 9.7 for **RSE**.

8. Prove Deduction Theorem 9.11 for Mendelson (1973) formalization.

9. In the proof of Deduction Theorem 9.11 for the proof system **H** we used gthe completeness of **H**. Write a proof of the Deduction Theorem 9.11 for **H** without use of its completeness.

Chapter 10

Predicate Automated Proof Systems Completeness of Classical Predicate Logic

We define and discuss here a Rasiowa and Sikorski Gentzen style proof system **QRS** for classical predicate logic. The propositional version of it, the **RS** proof system, was studied in detail in chapter 6. These both proof systems admit a *constructive proof* of completeness theorem. We adopt Rasiowa, Sikorski (1961) technique of construction a counter model determined by a decomposition tree to prove **QRS** completeness theorem 10.4. The proof, presented in section 10.3, is a generalization of the completeness proofs of **RS** and other Gentzen style propositional systems presented in details in chapter 6. We refer the reader to this chapter as it provides a good introduction to the subject.

The other Gentzen type predicate proof system, including the original Gentzen proof systems **LK, LI** for classical and intuitionistic predicate logics are obtained from their propositional versions discussed in detail in chapter 6. It can be done in a similar way as a generalization of the propositional **RS** the predicate **QRS** system presented here. We leave these generalizations as an exercises for the reader. That includes also the predicate language version of Gentzen proof of cut elimination theorem, Hauptzatz (1935). The Hauptzatz proof for the predicate classical **LK** and intuitionistic **LI** systems is easily obtained from the propositional proof included in chapter6.

There are of course other types of automated proof systems based on different methods of deduction.
There is a *Natural Deduction* mentioned by Gentzen in his Hauptzatz paper in

1935 and later fully developed by Dag Prawitz (1965). It is now called Prawitz, or Gentzen-Prawitz Natural Deduction.
There is a *Semantic Tableaux* deduction method invented by Evert Beth (1955). It was conequently simplified and further developed by Raymond Smullyan (1968). It is now often called Smullyan *Semantic Tableaux.*
Finally, there is a *Resolution.* The resolution method can be traced back to Davis and Putnam (1960). Their work is still known as Davis-Putnam method.The difficulties of their method were eliminated by John Alan Robinson (1965) and developed into what we call now Robinson Resolution, or just a Resolution.
There are many excellent textbooks covering each of these methods. We recommend Melvin Fitting book *First-order logic and automated theorem proving*(2nd ed.). Springer-Verlag(1996) as the one that not only covers all of them but also discusses their relationships.

The Resolution proof system for propositional or predicate logic operates on a set of *clauses* as a basic expressions and uses a *resolution rule* as the only rule of inference. In section 10.4 we define and prove correctness of effective *procedures* of converting any formula A into a corresponding set of clauses in both propositional and predicate cases. The correctness of propositional case is established by theorem 10.5, of predicate case by theorem 10.6. In the first step of the predicate procedure we define a process of *elimination of quantifiers* from the original language. It is called Skolemization of the language and is presented in section 10.4.1. The correctness of the Skolemization is established by Skolem theorem 10.11. In the second step of the procedure we show how convert a quantifiers free formula into logically equivalent set of clauses. It is presented with a proof of correctness (theorem 10.13) in section 10.4.2.

10.1 QRS Proof System

We define components and semantics of the proof system **QRS** as follows.
Language $\mathcal{L}$

We adopt a predicate (first order) language

$$\mathcal{L} = \mathcal{L}_{\{\cap,\cup,\Rightarrow,\neg\}}(\mathbf{P},\mathbf{F},\mathbf{C}) \tag{10.1}$$

for **P, F, C** countably infinite sets of predicate, functional, and constant symbols respectively.
Let $\mathcal{F}$ denote a set of formulas of $\mathcal{L}$. The rules of inference of our system **QRS** operate on *finite sequences of formulas*, i.e. elements of $\mathcal{F}^*$ so we define the set of expressions of of **QRS** as follows. **Expressions $\mathcal{E}$**

We adopt as the set of expressions $\mathcal{E}$ of **RS** the set $\mathcal{F}^*$, i.e.

$$\mathcal{E} = \mathcal{F}^*.$$

We will denote the expressions of **QRS**, i.e. the finite sequences of formulas by

$$\Gamma, \Delta, \Sigma, \text{with indices if necessary.}$$

In order to define the axioms LA and the set of rules of inference of **QRS** we need to bring back some notions and to introduce some definitions.

An **atomic formula** of the predicate language $\mathcal{L}$ defined by (10.1) is any element of $\mathcal{A}^*$ (finite strings over the alphabet of $\mathcal{L}$) of the form

$$R(t_1, t_2, ..., t_n)$$

where $R \in \mathbf{P}, \#R = n$ and $t_1, t_2, ..., t_n \in \mathbf{T}$.

The set of all **atomic formulas** is denoted by $A\mathcal{F}$ and is defined as

$$A\mathcal{F} = \{R(t_1, t_2, ..., t_n) \in \mathcal{A}^* : \quad R \in \mathbf{P},\ t_1, t_2, ..., t_n \in \mathbf{T},\ n \geq 1\}. \qquad (10.2)$$

We use symbols $R, Q, P, ...$ with indices if necessary to denote the **atomic formulas**.

Literals
We form a special subset $LT \subseteq \mathcal{F}$ of formulas, called a set of all *literals*, which is defined as follows.

$$LT = \{R : \ R \in A\mathcal{F}\} \cup \{\neg R : \ R \in A\mathcal{F}\}. \qquad (10.3)$$

The atomic formulas (10.2) are called **positive literals** and the elements of the second set of the above union (10.3), i.e. the negations of the atomic formulas are called **negative literals**.

Indecomposable, Decomposable Formulas
A formula $A \in \mathcal{F}$ is *indecomposable* if and only if it is atomic or a negation of an atomic formula, i.e. an *literal.* Otherwise A is *decomposable.*

Now we form *finite sequences* out of formulas (and, as a special case, out of literals). We need to distinguish the sequences formed out of literals from the sequences formed out of other formulas, so we adopt the following definition and notaions.

Indecomposable, Decomposable Sequences
A sequence Γ is *indecomposable* if and only if is formed out of indecomposable formulas only. Otherwise is *decomposable.*
We denote **indecomposable sequences** by by

$$\Gamma^{'},\ \Delta^{'},\ \Sigma^{'}, \ldots \ \text{ with indices if necessary.} \qquad (10.4)$$

By definition, $\Gamma^{'},\ \Delta^{'},\ \Sigma^{'} \ldots$ are finite sequences (empty included) formed out of **literals**, i.e $\Gamma^{'},\ \Delta^{'},\ \Sigma^{'}\Gamma^{'},\ \Delta^{'},\ \Sigma^{'} \in LT^*$.

We denote by

$$\Gamma,\ \Delta,\ \Sigma, \ldots \quad \text{with indices if necessary,} \tag{10.5}$$

the elements of $\mathcal{F}^*$, i.e. we denote $\Gamma,\ \Delta,\ \Sigma$ finite sequences (empty included) formed out of elements of $\mathcal{F}$.

Logical Axioms LA

As the *logical axiom* of **QRS** we adopt any sequence of formulas which contains a and its negation, i.e any sequence of the form

$$\Gamma_1,\ A,\ \Gamma_2,\ \neg A'\ \Gamma_3 \quad \text{or} \quad \Gamma_1,\ \neg A,\ \Gamma_2,\ A,\ \Gamma_3 \tag{10.6}$$

for any literal $A \in LT$ and any sequences $\Gamma_1, \Gamma_2, \Gamma_3 \in \mathcal{F}^*$ of formulas.

$$\textbf{Rules of inference } \mathcal{R} \tag{10.7}$$

Group 1: Propositional Rules

Disjunction rules

$$(\cup)\ \frac{\Gamma^{'}, A, B, \Delta}{\Gamma', (A \cup B), \Delta}, \qquad (\neg\cup)\ \frac{\Gamma^{'}, \neg A, \Delta \quad : \quad \Gamma^{'}, \neg B, \Delta}{\Gamma', \neg(A \cup B), \Delta}$$

Conjunction rules

$$(\cap)\ \frac{\Gamma^{'}, A, \Delta \quad ; \quad \Gamma^{'}, B, \Delta}{\Gamma', (A \cap B), \Delta}, \qquad (\neg\cap)\ \frac{\Gamma^{'}, \neg A, \neg B, \Delta}{\Gamma', \neg(A \cap B), \Delta}$$

Implication rules

$$(\Rightarrow)\ \frac{\Gamma^{'}, \neg A, B, \Delta}{\Gamma', (A \Rightarrow B), \Delta}, \qquad (\neg\ \Rightarrow)\ \frac{\Gamma^{'}, A, \Delta \quad : \quad \Gamma^{'}, \neg B, \Delta}{\Gamma', \neg(A \Rightarrow B), \Delta}$$

Negation rule

$$(\neg\ \neg)\ \frac{\Gamma^{'}, A, \Delta}{\Gamma', \neg\neg A, \Delta}$$

where $\Gamma^{'} \in LT^*, \Delta \in \mathcal{F}^*, A, B \in \mathcal{F}$.

Group 2: Quantifiers Rules

(∃)

$$\frac{\Gamma^{'}, A(t), \Delta, \exists x A(x)}{\Gamma', \exists x A(x), \Delta}$$

where t is an arbitrary term.

(∀)

$$\frac{\Gamma^{'}, A(y), \Delta}{\Gamma', \forall x A(x), \Delta}$$

where y is a free individual variable which does not appear in any formula in the conclusion, i.e. in the sequence $\Gamma^{'}, \forall x A(x), \Delta$.

(¬∀)

$$\frac{\Gamma^{'}, \exists x \neg A(x), \Delta}{\Gamma', \neg \forall x A(x), \Delta}$$

(¬∃)

$$\frac{\Gamma^{'}, \forall x \neg A(x), \Delta}{\Gamma', \neg \exists x A(x), \Delta}$$

$\Gamma^{'} \in \mathcal{LT}^*, \Delta \in \mathcal{F}^*, A, B \in \mathcal{F}$.

Note that $A(t), A(y)$ denotes a formula obtained from $A(x)$ by writing t, y, respectively, in place of all occurrences of x in A. The variable y in (∀) is called the *eigenvariable*. The condition: *where y is a free individual variable which does not appear in any formula in the conclusion* is called the *eigenvariable condition.*

All occurrences of y in $A(y)$ of the rule (∀) are fully indicated.

The Proof System QRS

Formally we define the proof system **QRS** as follows.

$$\textbf{QRS} = (\mathcal{L}_{\{\neg,\Rightarrow,\cup,\cap\}},\ \mathcal{E},\ \ LA,\ \ \mathcal{R}), \tag{10.8}$$

where $\mathcal{E} = \{\Gamma :\ \ \Gamma \in \mathcal{F}^*\}$, LA contains logical axioms of the system defined by (10.6), $\mathcal{R}$ is the set of rules of inference:

$$\mathcal{R} = \{(\cup),\ (\neg\cup), (\cap), (\neg\cap), (\Rightarrow), (\neg \Rightarrow), (\neg\neg), (\neg\forall), (\neg\exists), (\forall), (\exists))\}$$

defined by (10.7).

By a **formal proof** of a sequence Γ in the proof system **RS** we understand any sequence

$$\Gamma_1,\ \Gamma_2, ...\ \Gamma_n \tag{10.9}$$

of sequences of formulas (elements of $\mathcal{F}^*$, such that

1. $\Gamma_1 \in LA,\ \ \Gamma_n = \Gamma$, and

2. for all i $(1 \leq i \leq n)$ $\Gamma_i \in AL$, or Γ_i is a conclusion of one of the inference rules of **QRS** with all its premisses placed in the sequence $\Gamma_1,\ \Gamma_2,\ ....\ \Gamma_{i-1}$.

We write, as usual,

$$\vdash_{QRS} \Gamma$$

to denote that Γ has a formal proof in **QRS**.

As the proofs in **QRS** are sequences (definition of the formal proof) of sequences of formulas (definition of expressions $\mathcal{E}$) we will not use "," to separate the steps of the proof, and write the formal proof as $\Gamma_1;\ \Gamma_2;\ ....\ \Gamma_n$.

We write, however, the formal proofs in **QRS** as we did the propositional case (chapter 6), in a form of trees rather then in a form of sequences, ie. in a form of a tree, where *leafs* of the tree are axioms, *nodes* are sequences such that each sequence on the tree follows from the ones immediately preceding it by one of the rules. The *root* is a theorem. We picture, and write our tree-proofs with the node on the top, and leafs on the very bottom, instead of more common way, where the leafs are on the top and root is on the bottom of the tree. We adopt hence the following definition.

Definition 10.1 (Proof Tree)

By a proof tree, or **QRS***-proof of* Γ *we understand a tree* $\mathbf{T}_\Gamma$ *of sequences satisfying the following conditions:*

1. The topmost sequence, i.e the root of $\mathbf{T}_\Gamma$ *is* Γ*,*

2. all leafs are axioms,

3. the nodes are sequences such that each sequence on the tree follows from the ones immediately preceding it by one of the rules of inference (10.7).

We picture, and write our proof trees with the node on the top, and leafs on the very bottom, instead of more common way, where the leafs are on the top and root is on the bottom of the tree.

In particular cases, as in the propositional case, we will write our proof- trees indicating additionally the name of the inference rule used at each step of the proof. For example, if in a proof of a formula A from axioms (10.6) we use subsequently the rules

$$(\cap),\ (\cup),\ (\forall),\ (\cap),\ (\neg\neg),\ (\forall),\ (\Rightarrow)$$

we represent the proof as the following tree denoted by $\mathbf{T}_A$.

$\mathbf{T}_A$

Formula A

$\mid (\Rightarrow)$

conclusion of $(\forall)$

$\mid (\forall)$

conclusion of $(\neg\neg)$

$\mid (\neg\neg)$

conclusion of $(\cap)$

$\bigwedge(\cap)$

conclusion of $(\forall)$ *conclusion of* $(\cup)$

$\mid (\forall)$ $\mid (\cup)$

axiom *conclusion of* $(\cap)$

$\bigwedge(\cap)$

axiom *axiom*

Remark that the derivation trees don't represent a different *definition* of a formal proof. This remains the same in the Gentzen - style systems. Trees represent a certain *visualization* for those proofs and any formal proof in any system can be represented in a tree form.

10.2 QRS Decomposition Trees

The main advantage of the Gentzen proof systems lies not in a way we generate proofs in them, but in the way we can *search* for proofs in them. That such proof searches happens to be deterministic and automatic. We conduct such search by treating inference rules as *decomposition rules* (see chapter 6) and building *decomposition trees.*A general principle of building a decomposition tree is the following.

Decomposition Tree $\mathbf{T}_\Gamma$
For each $\Gamma \in \mathcal{F}^*$, a decomposition tree $\mathbf{T}_\Gamma$ is a tree build as follows.
Step 1. The sequence Γ is the **root** of $\mathbf{T}_\Gamma$ and for any node Δ of the tree we follow the steps bellow.

Step 2. If Δ is indecomposable or an axiom, then Δ becomes a **leaf** of the tree.
Step 3. If Δ is decomposable, then we traverse Δ from left to right to identify the first **decomposable formula** B and identify inference rule treated as decomposition rule determined uniquely by B. We put its left and right premisses as the left and right leaves, respectively.
Step 4. We repeat steps 2 and 3 until we obtain only leaves or infinite branch.

In particular case when when Γ has only one element, namely a a formula $A \in \mathcal{F}$, we define we call it a decomposition tree of A and denote by $\mathcal{T}_A$.

Here is a detailed definition of the decomposition tree for **QRS**.

QRS Decomposition Tree Definition (10.10)

Given a formula $A \in \mathcal{F}$, we define its decomposition tree $\mathbf{T}_A$ as follows.

Observe that the inference rules of **QRS** are divided in two groups: propositional connectives rules and quantifiers rules. The propositional connectives rules are: $(\cup), (\neg\cup), (\cap), (\neg\cap), (\Rightarrow), (\neg \Rightarrow)$, and $(\neg\neg)$. The quantifiers rules are: $(\forall), (\exists), (\neg\forall)$ and $(\neg\exists)$.

We define the decomposition tree in the case of the propositional rules and the rules $(\neg\forall)$, $(\neg\exists)$ in the exactly the same way as in the propositional case (chapter 6).

The case of the rules $(\forall)$ and $(\exists)$ is more complicated, as the rules contain the specific conditions under which they are applicable.

To define the way of decomposing the sequences of the form $\Gamma^{'}, \forall x A(x), \Delta$ or $\Gamma^{'}, \exists x A(x), \Delta$, i.e. to deal with the rules $(\forall)$ and $(\exists)$ in a way which would preserve the property of the *uniqueness* of the decomposition tree, we assume that all terms form a one-to one sequence

$$t_1, t_2, ..., t_n, ... \tag{10.11}$$

Observe, that by the definition, all free variables are terms, hence all free variables appear in the sequence 10.11. Let Γ be a sequence on the tree with $\forall$ as a main connective, i.e. Γ is of the form $\Gamma^{'}, \forall x A(x), \Delta$. We write a sequence $\Gamma^{'}, A(x), \Delta$ below it on the tree, as its child, where the variable x has to fulfill the following condition.

Condition 10.1 $(\forall)$

x is the first free variable in the sequence 10.11 such that x does not appear in any formula in $\Gamma^{'}, \forall x A(x), \Delta$.

Observe, that the condition 10.1 corresponds to the restriction put on the application of the rule $(\forall)$.

If the main connective of Γ, i.e. the main connective of the first formula in Γ which is not an literal, is $(\exists)$. In this case Γ is of the form $\Gamma^{'}, \exists x A(x), \Delta$, we write a sequence $\Gamma^{'}, A(t), \Delta, \exists x A(x)$ as its child, where the term t has to fulfill the following condition.

Condition 10.2 $(\exists)$

t is the first term in the sequence 10.11 such that the formula A(t) does not appear in any sequence which is placed above $\Gamma^{'}, A(t), \Delta, \exists x A(x)$ on the tree.

The fact that the sequence 10.11 is one- to - one and the fact that, by the conditions 10.1 and 10.2, we always chose the first appropriate term (variable) from this sequence, guarantee that the decomposition process is also unique in the case of the quantifiers rules $(\forall)$ and $(\exists)$.

From all above, and we conclude the following.

Theorem 10.1

For any formula $A \in \mathcal{F}$,

(i) *the decomposition tree $\mathbf{T}_A$ is unique.*

(ii) *Moreover, the following conditions hold.*

1. If $\mathbf{T}_A$ is **finite** *and all its leaves are axioms, then*

$$\vdash_{QRS} \; A$$

and $\mathbf{T}_A$ is a tree-proof of A in **QRS**.

2. If $\mathbf{T}_A$ is **finite** *and contains a non-axiom leaf, or $\mathbf{T}_A$ is* **infinite**, *then*

$$\nvdash_{QRS} A.$$

10.2.1 Examples of Decomposition Trees

In all the examples below, the formulas $A(x), B(x)$ represent any formula. But there is no indication about their particular components, so they are treated as **indecomposable formulas**.

Example 10.1

The decomposition tree $\mathcal{T}_A$ of the de Morgan Law

$$(\neg \forall x A(x) \Rightarrow \exists x \neg A(x))$$

is the following.

$$\mathbf{T}_A$$

$$(\neg\forall xA(x) \Rightarrow \exists x\neg A(x))$$

$$\mid (\Rightarrow)$$

$$\neg\neg\forall xA(x), \exists x\neg A(x)$$

$$\mid (\neg\neg)$$

$$\forall xA(x), \exists x\neg A(x)$$

$$\mid (\forall)$$

$$A(x_1), \exists x\neg A(x)$$

where x_1 is a first free variable in the sequence 10.11 such that x_1 does not appear in $\forall xA(x), \exists x\neg A(x)$

$$\mid (\exists)$$

$$A(x_1), \neg A(x_1), \exists x\neg A(x)$$

where x_1 is the first term (variables are terms) in the sequence 10.11 such that $\neg A(x_1)$ does not appear on a tree above $A(x_1), \neg A(x_1), \exists x\neg A(x)$
Axiom

The above tree $\mathbf{T}_A$ ended with an axiom, so it represents a proof of

$$(\neg\forall xA(x) \Rightarrow \exists x\neg A(x))$$

in **QRS**, i.e. we proved that

$$\vdash_{QRS} (\neg\forall xA(x) \Rightarrow \exists x\neg A(x)).$$

Example 10.2

The decomposition tree $\mathbf{T}_A$ of

$$(\forall xA(x) \Rightarrow \exists xA(x))$$

is the following.

$$\mathbf{T}_A$$

$$(\forall xA(x) \Rightarrow \exists xA(x))$$

$$\mid (\Rightarrow)$$

$$\neg\forall xA(x), \exists xA(x)$$
$$\mid (\neg\forall)$$
$$\neg\forall xA(x), \exists xA(x)$$
$$\exists x\neg A(x), \exists xA(x)$$
$$\mid (\exists)$$
$$\neg A(t_1), \exists xA(x), \exists x\neg A(x)$$

where t_1 is the first term in the sequence 10.11, such that $\neg A(t_1)$ does not appear on the tree above $\neg A(t_1), \exists xA(x), \exists x\neg A(x)$

$$\mid (\exists)$$
$$\neg A(t_1), A(t_1), \exists x\neg A(x), \exists xA(x)$$

where t_1 is the first term in the sequence 10.11, such that $A(t_1)$ does not appear on the tree above $\neg A(t_1), A(t_1), \exists x\neg A(x), \exists xA(x)$

Axiom

The above tree also ended with the axiom, hence we proved that

$$\vdash_(QRS)\ (\forall xA(x) \Rightarrow \exists xA(x)).$$

Example 10.3

The decomposition tree $\mathbf{T}_A$ *of*

$$(\exists xA(x) \Rightarrow \forall xA(x))$$

is the following.

$$\mathbf{T}_A$$
$$(\exists xA(x) \Rightarrow \forall xA(x))$$
$$\mid (\Rightarrow)$$
$$\neg\exists xA(x), \forall xA(x)$$
$$\mid (\neg\exists)$$
$$\forall x\neg A(x), \forall xA(x)$$
$$\mid (\forall)$$
$$\neg A(x_1), \forall xA(x)$$

where x_1 is a first free variable in 10.11 such that x_1 does not appear in $\forall x\neg A(x), \forall xA(x)$

$$\mid (\forall)$$
$$\neg A(x_1), A(x_2)$$

where x_2 is a first free variable in 10.11 such that x_2 does not appear in $\neg A(x_1), \forall xA(x)$, the sequence 10.11 is one-to- one, hence $x_1 \neq x_2$

Non - axiom

The decomposition tree, for any formula A is *unique*, so we conclude from the fact that the above tree has a non-axiom branch that

$$\not\vdash_{QRS} (\exists x A(x) \Rightarrow \forall x A(x)).$$

Remark when constructing the following tree $\mathbf{T}_A$ for the formula $\exists x A(x)$ in example 10.4 below we adopt on the right branch of the a tree in the the short-hand notation instead of the repeating a similar reasoning performed on the left branch.

Example 10.4

The decomposition tree $\mathbf{T}_A$ of the formula $\exists x A(x)$ is the following.

$$\mathbf{T}_A$$

$$\exists x A(x)$$

$$\mid (\exists)$$

$$A(t_1), \exists x A(x)$$

where t_1 is the first term in the sequence 10.11, such that $A(t_1)$ does not appear on the tree above $A(t_1), \exists x A(x)$

$$\mid (\exists)$$

$$A(t_1), A(t_2), \exists x A(x)$$

where t_2 is the first term in the sequence 10.11, such that $A(t_2)$ does not appear on the tree above $A(t_1), A(t_2), \exists x A(x)$, i.e. $t_2 \neq t_1$

$$\mid (\exists)$$

$$A(t_1), A(t_2), A(t_3), \exists x A(x)$$

where t_3 is the first term in the sequence 10.11, such that $A(t_3)$ does not appear on the tree above $A(t_1), A(t_2), A(t_3), \exists x A(x)$, i.e. $t_3 \neq t_2 \neq t_1$

$$\mid (\exists)$$

$$A(t_1), A(t_2), A(t_3), A(t_4), \exists x A(x)$$

$$\mid (\exists)$$

.....

$$\mid (\exists)$$

.....

$$infinite\ branch$$

Obviously, the above decomposition tree is infinite, what proves that

$$\nvdash_{QRS} \exists x A(x).$$

We will find now the proof of the distributivity law

$$(\exists x(A(x) \cap B(x)) \Rightarrow (\exists x A(x) \cap \exists x B(x)))$$

and show that we can't prove in **QRS** the inverse implication

$$((\exists x A(x) \cap \exists x B(x)) \Rightarrow \exists x(A(x) \cap B(x))).$$

Remark when constructing the following trees $\mathbf{T}_A$ in examples 10.5, 10.6 adopt, as we did in the previous example 10.4, the shorthand notation when the reasoning is similar to the one presented in the example 10.4.

Example 10.5

The decomposition tree $_A$ *of the first formula*

$$(\exists x(A(x) \cap B(x)) \Rightarrow (\exists x A(x) \cap \exists x B(x)))$$

is the following.

$$\mathbf{T}_A$$

$$(\exists x(A(x) \cap B(x)) \Rightarrow (\exists x A(x) \cap \exists x B(x)))$$

$$\mid (\Rightarrow)$$

$$\neg \exists x(A(x) \cap B(x)), (\exists x A(x) \cap \exists x B(x))$$

$$\mid (\neg \exists)$$

$$\forall x \neg(A(x) \cap B(x)), (\exists x A(x) \cap \exists x B(x))$$

$$\mid (\forall)$$

$$\neg(A(x_1) \cap B(x_1)), (\exists x A(x) \cap \exists x B(x))$$

where x_1 *is a first free variable in the sequence 10.11 such that* x_1 *does not appear in* $\forall x \neg(A(x) \cap B(x)), (\exists x A(x) \cap \exists x B(x))$

$$\mid (\neg \cap)$$

$$\neg A(x_1), \neg B(x_1), (\exists x A(x) \cap \exists x B(x))$$

$$\bigwedge (\cap)$$

$$\neg A(x_1), \neg B(x_1), \exists x A(x)$$
$$\mid (\exists)$$
$$\neg A(x_1), \neg B(x_1), A(t_1), \exists x A(x)$$

where t_1 is the first term in the sequence 10.11, such that $A(t_1)$ does not appear on the tree above $\neg A(x_1), \neg B(x_1), A(t_1), \exists x A(x)$ Observe, that it is possible that $t_1 = x_1$, as $A(x_1)$ does not appear on the tree above. By the definition of the sequence 10.11, x_1 is placed somewhere in it, i.e. $x_1 = t_i$, for certain $i \geq 1$. It means that after i applications of the step $(\exists)$ in the decomposition tree, we will get a step:

$$\mid (\exists)$$
$$\neg A(x_1), \neg B(x_1), ...A(x_1), \exists x A(x)$$

$$\neg A(x_1), \neg B(x_1), \exists x B(x)$$
$$\mid (\exists)$$
$$\neg A(x_1), \neg B(x_1), B(t_1), \exists x B(x)$$
$$\mid (\exists)$$
$$...$$
$$\mid (\exists)$$
$$\neg A(x_1), \neg B(x_1), ...B(x_1), \exists x B(x)$$

All leaves of the above tree $\mathbf{T}_A$ are axioms, what means that we proved

$$\vdash_{QRS} (\exists x(A(x) \cap B(x)) \Rightarrow (\exists x A(x) \cap \exists x B(x))).$$

We construct now, as the last example, a decomposition tree $\mathbf{T}_A$ of the formula $((\exists x A(x) \cap \exists x B(x)) \Rightarrow \exists x(A(x) \cap B(x)))$.

Example 10.6

The decomposition tree of the formula

$$((\exists x A(x) \cap \exists x B(x)) \Rightarrow \exists x(A(x) \cap B(x)))$$

is the following.

$$\mathbf{T}_A$$

$$((\exists x A(x) \cap \exists x B(x)) \Rightarrow \exists x(A(x) \cap B(x)))$$
$$\mid (\Rightarrow)$$
$$\neg(\exists x A(x) \cap \exists x B(x)) \exists x(A(x) \cap B(x))$$
$$\mid (\neg \cap)$$
$$\neg \exists x A(x), \neg \exists x B(x), \exists x(A(x) \cap B(x))$$
$$\mid (\neg \exists)$$
$$\forall x \neg A(x), \neg \exists x B(x), \exists x(A(x) \cap B(x))$$

$$\mid (\forall)$$

$$\neg A(x_1), \neg\exists x B(x), \exists x(A(x) \cap B(x))$$

$$\mid (\neg\exists)$$

$$\neg A(x_1), \forall x \neg B(x), \exists x(A(x) \cap B(x))$$

$$\mid (\forall)$$

$$\neg A(x_1), \neg B(x_2), \exists x(A(x) \cap B(x))$$

By the reasoning similar to the reasonings in the previous examples we get that $x_1 \neq x_2$

$$\mid (\exists)$$

$$\neg A(x_1), \neg B(x_2), (A(t_1) \cap B(t_1)), \exists x(A(x) \cap B(x))$$

where t_1 is the first term in the sequence 10.11, such that $(A(t_1) \cap B(t_1))$ does not appear on the tree above $\neg A(x_1), \neg B(x_2), (A(t_1) \cap B(t_1)), \exists x(A(x) \cap B(x))$ Observe, that it is possible that $t_1 = x_1$, as $(A(x_1) \cap B(x_1))$ does not appear on the tree above. By the definition of the sequence 10.11, x_1 is placed somewhere in it, i.e. $x_1 = t_i$, for certain $i \geq 1$. For simplicity, we assume that $t_1 = x_1$ and get the sequence:

$$\neg A(x_1), \neg B(x_2), (A(x_1) \cap B(x_1)), \exists x(A(x) \cap B(x))$$

$$\bigwedge (\cap)$$

$\neg A(x_1), \neg B(x_2),$
$A(x_1), \exists x(A(x) \cap B(x))$
$Axiom$

$\neg A(x_1), \neg B(x_2),$
$B(x_1), \exists x(A(x) \cap B(x))$
$\mid (\exists)$
$\neg A(x_1), \neg B(x_2), B(x_1),$
$(A(x_2) \cap B(x_2)), \exists x(A(x) \cap B(x))$

where $x_2 = t_2$ ($x_1 \neq x_2$) is the first term in the sequence 10.11, such that $(A(x_2) \cap B(x_2))$ does not appear on the tree above $\neg A(x_1), \neg B(x_2), (B(x_1), (A(x_2) \cap B(x_2)), \exists x(A(x) \cap B(x))$. We assume that $t_2 = x_2$ for the reason of simplicity.

$\bigwedge(\cap)$

$\neg A(x_1),$
$\neg B(x_2),$
$B(x_1), A(x_2),$
$\exists x(A(x) \cap B(x))$
$\mid (\exists)$
$...$
$\bigwedge(\cap)$
$...$
$\mid (\exists)$
$...$
$\mid (\exists)$
$infinite\ branch$

$\neg A(x_1),$
$\neg B(x_2),$
$B(x_1), B(x_2),$
$\exists x(A(x) \cap B(x))$
$Axiom$

The above decomposition tree $\mathbf{T}_A$ contains an infinite branch what means that

$$\nvdash_{QRS} \ ((\exists x A(x) \cap \exists x B(x)) \Rightarrow \exists x(A(x) \cap B(x))).$$

10.3 Proof of QRS Completeness

Our main goal is to prove the Completeness Theorem for **QRS**. The proof of completeness theorem presented here is due to Rasiowa and Sikorski (1961), as is the proof system **QRS**. We adopted their proof to propositional case in chapter 6.The completeness proofs, in the propositional case and in predicate case, are constructive as they are based on a direct construction of a counter model for any unprovable formula. The construction of the counter model for the unprovable formula A uses the decomposition tree $\mathbf{T}_A$. We call such constructed counter model a counter model determined by the tree $\mathbf{T}_A$. Rasiowa-Sikorski type of constructive proofs of counter models determined by the tree decomposition trees relay heavily of the notion of a *strong soundness*. We define it here (definition 10.8), adopting chapter 4 general definition to our case.

Given a first order language $\mathcal{L}$ (10.1) with the set VAR of variables and the set $\mathcal{F}$ of formulas. We define, after chapter 8 a notion of a model and a counter-model of a formula A of $\mathcal{L}$ and then extend it to the the set $\mathcal{F}^*$ establishing the **semantics** for **QRS**.

Definition 10.2 (Model)

A structure $\mathcal{M} = [M, I]$ is called a model of $A \in \mathcal{F}$ if and only if

$$(\mathcal{M}, v) \models A$$

for all assignments $v : VAR \longrightarrow M$.
We denote it by

$$\mathcal{M} \models A.$$

M is called the universe of the model, I the interpretation.

Definition 10.3 (Counter - Model)

A structure $\mathcal{M} = [M, I]$ is called a counter- model of $A \in \mathcal{F}$ if and only if there is $v : VAR \longrightarrow M$, such that

$$(\mathcal{M}, v) \not\models A.$$

We denote it by

$$\mathcal{M} \not\models A.$$

The definition of the first order logic tautology is the following.

Definition 10.4 (Tautology)

For any $A \in \mathcal{F}$, A is called a (predicate) tautology and denoted by

$$\models A$$

if and only if all structures $\mathcal{M} = [M, I]$ are models of A, i.e.

$$\models A \quad \text{if and only if} \quad \mathcal{M} \models A$$

for all structures $\mathcal{M} = [M, I]$ for $\mathcal{L}$.

Directly from the above definition we get the following, simple fact.

Fact 10.1 (Counter Model)

For any $A \in \mathcal{F}$, A is not a tautology ($\not\models A$) if and only if there is a counter - model $\mathcal{M} = [M, I]$ of A, i.e. we can define M, I, and v such that $([M, I], v) \not\models A$.

Definition 10.5

For any sequence $\Gamma \in \mathcal{F}^*$, by

$$\delta_\Gamma$$

we understand any disjunction of all formulas of Γ.

Definition 10.6 (QRS Semantics)

A structure $\mathcal{M} = [M, I]$ for $\mathcal{L}$ is called a **model** *of a $\Gamma \in \mathcal{F}^*$ and denoted by*

$$\mathcal{M} \models \Gamma$$

if and only if

$$\mathcal{M} \models \delta_\Gamma.$$

The sequence Γ is a predicate tautology if and only if the formula δ_Γ is a predicate tautology, i.e.

$$\models \Gamma \quad \textit{if and only if} \quad \models \delta_\Gamma.$$

Our goal now is to prove the completeness theorem for **QRS**.The correctness of the proof we present depends on the strong soundness of the rules of inference of rules of inference defined as follows.

Definition 10.7 (Strongly Sound Rules)

Given a predicate language (10.1) proof system $S = (\mathcal{L}, \mathcal{E}, LA, \mathcal{R})$ An inference rule $r \in \mathcal{R}$ of the form

$$(r) \qquad \frac{P_1 \ ; \ P_2 \ ; \ .... \ ; \ P_m}{C}$$

is **strongly sound** *if the following condition holds for and structure $\mathcal{M} = [M, I]$ for $\mathcal{L}$.*

$$\mathcal{M} \models \{P_1, P_2, .P_m\} \quad \textit{if and only if} \quad \mathcal{M} \models C. \tag{10.12}$$

We say it less formally that a rule (r) is **strongly sound** if the conjunction of its premisses is logically equivalent with the conclusion, i.e.

$$P_1 \cap P_2 \cap \ \dots \ \cap P_m \equiv C. \tag{10.13}$$

Definition 10.8 (Strongly Sound System)

A predicate language (10.1) proof system $S = (\mathcal{L}, \mathcal{E}, LA, \mathcal{R})$ is **strongly sound** *if and only if all logical axioms LA are tautologies and all its rules of inference $r \in \mathcal{R}$ are* **strongly sound***.*

Theorem 10.2 (Strong Soundness)

The proof system **QRS** *(10.8) is* **strongly sound***.*

Proof
We have already proved in chapter 6 strong soundness of the propositional rules. The quantifiers rule are strongly sound by straightforward verification and is left as an exercise.

The strong soundness property is stronger then soundness property, hence also the following holds.

Theorem 10.3 (Soundness Theorem)

For any $\Gamma \in \mathcal{F}^*$,

$$\textit{if} \quad \vdash_{QRS} \Gamma, \quad \textit{then} \quad \models \Gamma.$$

In particular, for any formula $A \in \mathcal{F}$,

$$\textit{if} \quad \vdash_{QRS} A, \quad \textit{then} \quad \models A.$$

Theorem 10.4 (Completeness Theorem)

For any $\Gamma \in \mathcal{F}^*$,

$$\vdash_{QRS} \Gamma \quad \textit{if and only if} \quad \models \Gamma.$$

In particular, for any formula $A \in \mathcal{F}$,

$$\vdash_{QRS} A \quad \textit{if and only if} \quad \models A.$$

Proof
We have to prove the inverse implication to the soundness theorem 10.3. We need to prove the formula A case only because the case of a sequence Γ can be reduced to the formula case. Namely, the disjunction of all formulas in Γ. I.e. we prove the implication:

$$\text{if} \quad \models A, \quad \text{then} \quad \vdash_{QRS} A.$$

We do it, as in the propositional case, by proving the opposite implication

$$\text{if} \quad \nvdash_{QRS} A \quad \text{then} \quad \not\models A.$$

This means that we want prove that for any formula A, unprovability of A in **QRS** ($\nvdash_{QRS} A$), allows us to define its counter- model. The counter- model is determined, as in the propositional case, by the decomposition tree $\mathbf{T}_A$. By theorem 10.1 each formula A, generates its unique decomposition tree $\mathcal{T}_A$ and A has a proof only if this tree is finite and all its end sequences (leaves) are axioms. Moreover, it says that we have two cases to consider:

(C1) the tree $\mathbf{T}_A$ is **finite** and contains a leaf which is not axiom, or

(C2) the tree $\mathbf{T}_A$ is **infinite**.

We will show how to construct a counter- model for A in both cases: a counter-model determined by a non-axiom leaf of the decomposition tree $\mathbf{T}_A$, or a counter- model determined by an infinite branch of $\mathbf{T}_A$.

Proof in case **(C1)**: $\mathbf{T}_A$ is **finite** and contains a non- axiom leaf.

Before describing a general method of constructing the counter-model determined by the decomposition tree $\mathcal{T}_A$ we describe it, as an example, for a case of a formula

$$(\exists x A(x) \Rightarrow \forall x A(x)),$$

and its **particular case**

$$(\exists x(P(x) \cap R(x,y)) \Rightarrow \forall x(P(x) \cap R(x,y))) \tag{10.14}$$

for P, R one and two argument predicate symbols, respectively.

We construct the counter model for the formula (10.14) as follows.

First we build its decomposition tree:

$$\mathbf{T}_A$$

$$(\exists x(P(x) \cap R(x,y)) \Rightarrow \forall x(P(x) \cap R(x,y)))$$

$$\mid (\Rightarrow)$$

$$\neg\exists x(P(x) \cap R(x,y)), \forall x(P(x) \cap R(x,y))$$

$$\mid (\neg\exists)$$

$$\forall x\neg(P(x) \cap R(x,y)), \forall x(P(x) \cap R(x,y))$$

$$\mid (\forall)$$

$$\neg(P(x_1) \cap R(x_1,y)), \forall x(P(x) \cap R(x,y))$$

where x_1 is a first free variable in 10.11 such that x_1 does not appear in $\forall x\neg(P(x) \cap R(x,y)), \forall x(P(x) \cap R(x,y))$

$$\mid (\neg\cap)$$

$$\neg P(x_1), \neg R(x_1,y), \forall x(P(x) \cap R(x,y))$$

$$\mid (\forall)$$

$$\neg P(x_1), \neg R(x_1, y), (P(x_2) \cap R(x_2, y))$$

where x_2 is a first free variable in the sequence 10.11 such that x_2 does not appear in $\neg P(x_1), \neg R(x_1, y), \forall x(P(x) \cap R(x, y))$, the sequence 10.11 is one-to- one, hence $x_1 \neq x_2$

$$\bigwedge(\cap)$$

$$\neg P(x_1), \neg R(x_1, y), P(x_2) \qquad \neg P(x_1), \neg R(x_1, y), R(x_2, y)$$

$x_1 \neq x_2$, Non-axiom $\qquad$ $x_1 \neq x_2$, Non-axiom

There are two non-axiom leaves. In order to define a counter-model for (10.14) determined by the tree $\mathbf{T}_A$ we need to chose only one of them. Let's choose the leaf

$$L_A = \neg P(x_1), \neg R(x_1, y), P(x_2). \tag{10.15}$$

We use the **non-axiom leaf** L_A to define a structure $\mathcal{M} = [M, I]$ and an assignment v, such that

$$(\mathcal{M}, v) \not\models A.$$

Such defined $\mathcal{M}$ is called a **counter - model** determined by the tree $\mathbf{T}_A$.

We take a the universe of $\mathcal{M}$ the set $\mathbf{T}$ of all terms of our language $\mathcal{L}$, i.e. we put

$$M = \mathbf{T}.$$

We define the interpretation I as follows. For any predicate symbol $Q \in \mathbf{P}, \#Q = n$ we put that $Q_I(t_1, \dots t_n)$ is **true** (holds) for terms $t_1, \dots t_n$ if and only if the negation $\neg Q_I(t_1, \dots t_n)$ of the formula $Q(t_1, \dots t_n)$ appears on the leaf L_A and $Q_I(t_1, \dots t_n)$ is **false** (does not hold) for terms $t_1, \dots t_n$ otherwise.

For any functional symbol $f \in \mathbf{F}, \#f = n$ we put $f_I(t_1, \dots t_n) = f(t_1, \dots t_n)$.

It is easy to see that in particular case of our non-axiom leaf (10.15)

$$L_A = \neg P(x_1),\ \neg R(x_1, y),\ P(x_2)$$

$P_I(x_1)$ is true for x_1, and not true for x_2. $R_I(x_1, y)$ is true (holds) holds for x_1 any for any $y \in VAR$.

We define the assignment $v : VAR \longrightarrow T$ as *identity*, i.e., we put $v(x) = x$ for any $x \in VAR$.

Obviously, for such defined structure $[M, I]$ and the assignment v we have that

$$([\mathbf{T}, I],\ v) \models P(x_1), \quad ([\mathbf{T}, I],\ v) \models R(x_1, y), \quad \text{and} \quad ([\mathbf{T}, I],\ v) \not\models P(x_2).$$

We hence obtain that

$$([\mathbf{T}, I],\ v) \not\models \neg P(x_1), \neg R(x_1, y), P(x_2).$$

This proves that such defined structure $[\mathbf{T}, I]$ is a counter model for a non-axiom leaf (10.15). By the strong soundness of QRS) (theorem 10.2) the structure $\mathcal{M} = [\mathbf{T}, I]$ is also a counter- model for the formula (10.14), i.e. we proved that

$$\not\models (\exists x(P(x) \cap R(x, y)) \Rightarrow \forall x(P(x) \cap R(x, y))).$$

C1: General Method
Let A be any formula such that $\nvdash_{QRS} A$.

Let $\mathcal{T}_A$ be a decomposition tree of A. By the fact that $\nvdash_{QRS}$ and **C1**, the tree $\mathcal{T}_A$ is finite and has a *non axiom leaf*

$$L_A \subseteq LT^*. \tag{10.16}$$

By definition, the leaf L_A contains only atomic formulas and negations of atomic formulas.

We use the **non-axiom leaf** L_A (10.16) to define a structure $\mathcal{M} = [M, I]$ an assignment $v : VAR \longrightarrow M$, such that $(\mathcal{M}, v) \not\models A$. Such defined structure $\mathcal{M}$ is called a **counter - model** determined by the tree $\mathbf{T}_A$.

Structure $\mathcal{M}$ Definition (10.17)

Given L_A. We define a structure

$$\mathcal{M} = [M, I] \tag{10.18}$$

and an assignment $v : VAR \longrightarrow M$ as follows.

1. We take a the universe of $\mathcal{M}$ the set $\mathbf{T}$ of all terms of our language $\mathcal{L}$' i.e. we put

$$M = \mathbf{T}.$$

2. For any predicate symbol $Q \in \mathbf{P}, \#Q = n$,

$$Q_I \subseteq \mathbf{T}^n$$

is such that $Q_I(t_1, \dots t_n)$ holds (is true) for terms $t_1, \dots t_n$ if and only if the negation $\neg Q(t_1, \dots t_n)$ of the formula $Q(t_1, \dots t_n)$ appears on the leaf L_A and $Q_I(t_1, \dots t_n)$ does not hold (is false) for terms $t_1, \dots t_n$ otherwise.

3. For any constant $c \in \mathbf{C}$, we put $c_I = c$, for any variable x, $x_I = x$.
For any functional symbol $f \in \mathbf{F}, \#f = n$,

$$f_I : \mathbf{T}^n \longrightarrow \mathbf{T}$$

is identity function, i.e. we put

$$f_I(t_1, \dots t_n) = f(t_1, \dots t_n)$$

for all $t_1, \dots t_n \in \mathbf{T}$.

4. We define the assignment $v : VAR \longrightarrow \mathbf{T}$ as *identity*, i.e. we put for all $x \in VAR$ $v(x) = x$.

Obviously, for such defined structure $[\mathbf{T}, I]$ and the assignment v we have that

$$([\mathbf{T}, I],\ v) \not\models\ P \text{ if formula } P \text{ appears in } \ L_A,$$

$$([\mathbf{T}, I],\ v) \models\ P \text{ if formula } \neg P \text{ appears in } \ L_A.$$

This proves that the structure $\mathcal{M} = [\mathbf{T}, I]$ and assignment v defined by (10.18) are such that

$$([\mathbf{T}, I], v) \not\models\ L_A.$$

By the **strong soundness** (theorem 10.2) of **QRS**

$$(([\mathbf{T}, I], v) \not\models\ A.$$

This proves $\mathcal{M} \not\models\ A$ and we proved that

$$\not\models\ A.$$

This ends the proof of the case **C1**.

Proof in case **(C2)**: $\mathbf{T}_A$ is **infinite**.

The case of the **infinite tree** is similar, even if a little bit more complicated. Observe first that the rule $(\exists)$ is the t rule of inference (decomposition) which can "produces" an infinite branch. We first show how to construct the counter-model in the case of the simplest application of this rule, i.e. in the case of the formula

$$\exists x P(x)$$

where P is an one argument relational symbol. All other cases are similar to this one. The infinite branch $\mathcal{B}_A$ in this case consists of all elements of the decomposition tree:

$$\mathbf{T}_A$$

$$\exists x P(x)$$

$$\mid (\exists)$$

$$P(t_1), \exists x P(x)$$

where t_1 is the first term in the sequence 10.11, such that $P(t_1)$ does not appear on the tree above $P(t_1), \exists x P(x)$

$$\mid (\exists)$$

$$P(t_1), P(t_2), \exists x P(x)$$

where t_2 is the first term in the sequence 10.11, such that $P(t_2)$ does not appear on the tree above $P(t_1), P(t_2), \exists x P(x)$, i.e. $t_2 \neq t_1$

$$\mid (\exists)$$

$$P(t_1), P(t_2), P(t_3), \exists x P(x)$$

where t_3 is the first term in the sequence 10.11, such that $P(t_3)$ does not appear on the tree above $P(t_1), P(_2), P(t_3), \exists x P(x)$, i.e. $t_3 \neq t_2 \neq t_1$

$$\mid (\exists)$$

$$P(t_1), P(t_2), P(t_3), P(t_4), \exists x P(x)$$

$$\mid (\exists)$$

.....

$$\mid (\exists)$$

.....

The infinite branch of $\mathbf{T}_A$, written from the top, in oder of appearance of formulas is

$$\mathcal{B}_A = \{\exists x P(x),\ P(t_1),\ A(t_2),\ P(t_2),\ P(t_4), ...\}$$

where $t_1, t_2, ...$ is a one - to one sequence (10.11) of all elements of the set $\mathbf{T}$ of all terms.

This means that the infinite branch $\mathcal{B}$ contains with the formula $\exists x P(x)$ all its instances $P(t)$, for all terms $t \in \mathbf{T}$.

We define the structure $\mathcal{M} = [M, I]$ and valuation v following the definition 10.17. We take as the universe M the set $\mathbf{T}$ of all terms, and now in our case we define P_I as follows: $P_I(t)$ holds if $\neg P(t) \in \mathcal{B}_A$ and $P_I(t)$ does not hold if $P(t) \in \mathcal{B}_A$.

It is easy to see that for any formula $P(t) \in \mathcal{B}$,

$$([T, I], v) \not\models P(t).$$

But the $A(t) \in \mathcal{B}$ are all instances $\exists x A(x)$, hence

$$([T, I], v) \not\models \exists x A(x).$$

C1: General Method
Let A be any formula such that $\nvdash_{QRS} A$.

Let $\mathcal{T}_A$ be an infinite decomposition tree of a formula A. Let $\mathcal{B}_A$ the infinite branch of $\mathbf{T}_A$, written from the top, in oder of appearance of sequences $\Gamma \in \mathcal{F}^*$ on it, where $\Gamma_0 = A$.

$$\mathcal{B}_A = \{\Gamma_0, \ \ \Gamma_1, \ \ \Gamma_2, \ \ \ldots \ \ \Gamma_i, \ \ \Gamma_{i+1}, \ \ \ldots\} \tag{10.19}$$

We define a set $L\mathcal{F} \subseteq \mathcal{F}$ of all *indecomposable* formulas appearing in at least one $\Gamma_i, \ i \leq j$, i.e.

$$L\mathcal{F} = \{B \in LT : \ \text{ there is } \ \Gamma_i \in \mathcal{B}_A, \text{ such that } B \text{ is in } \Gamma_i\}. \tag{10.20}$$

Note, that the following holds.
(1) if $i \leq i'$ and an indecomposable formula appears in Γ_i, then it also appears in $\Gamma_{i'}$.
(2) Since none of Γ_i, is an axiom (10.6), for every atomic formula (10.2) $P \in A\mathcal{F}$, at most one of the formulas P and $\neg P$ is in $L\mathcal{F}$ (10.20).

Counter Model Definition
Let $\mathbf{T}$ be the set of all terms. We define the structure $\mathcal{M} = [\mathbf{T}, I]$ and valuation v in the set $\mathbf{T}$ as in the definition 10.17, with the interpretation of predicates $Q \in \mathbf{Q}$ defined as follows.

For any predicate symbol $Q \in \mathbf{P}, \#Q = n$, $Q_I \subseteq \mathbf{T}^n$ is such that:

(1) $Q_I(t_1, \ldots t_n)$ does not hold (is false) for terms $t_1, \ldots t_n$ if and only if

$$Q_I(t_1, \ldots t_n) \in L\mathcal{F}$$

and
(2) $Q_I(t_1, \ldots t_n)$ does holds (is true) for terms $t_1, \ldots t_n$ if and only if

$$Q_I(t_1, \ldots t_n) \notin L\mathcal{F}.$$

This proves that the structure $\mathcal{M} = [\mathbf{T}, I]$ is such that

$$\mathcal{M} \not\models \ L\mathcal{F}. \tag{10.21}$$

To prove that $\not\models \ A$ it suffices that

$$\mathcal{M} \not\models \ A. \tag{10.22}$$

For this purpose we first introduce, for any formula $A \in \mathcal{F}$, an inductive definition of the order *ord* A of the formula A.
(1) If $A \in A\mathcal{F}$, then *ord* $A = 1$.
(2) If *ord* $A = n$, then *ord* $\neg A = n + 1$. (3) If *ord* $A \leq n$ and *ord* $B \leq n$, then *ord* $(A \cup B) =$ *ord* $(A \cap B) =$ *ord* $(A \Rightarrow B) = n + 1$.
(4) If *ord* $A(x) = n$, then *ord* $\exists x A(x) =$ *ord* $\forall x A(x) = n + 1$.

We conduct the proof of (10.23) by contradiction. Suppose that (10.23) does not hold, i.e. assume that

$$\mathcal{M} \models A. \tag{10.23}$$

Consider now a set $M\mathcal{F}$ of all formulas B appearing in one of the sequences Γ_i of the branch $\mathcal{B}_A$, such that

$$\mathcal{M} \models B. \tag{10.24}$$

We write the the set $M\mathcal{F}$ formally as follows.

$$M\mathcal{F} = \{B \in \mathcal{F} : \text{ for some } \Gamma_i \in \mathcal{B}_A,\ B \text{ is in } \Gamma_i \text{ and } \mathcal{M} \models B\}. \tag{10.25}$$

Observe that by assumption (10.23) and the definition (10.25), the formula A is in $M\mathcal{F}$ and hence $M\mathcal{F} \neq \emptyset$.

Let B' be a formula in $M\mathcal{F}$ such that $ord\ B' \leq ord\ B$ for every $B \in M\mathcal{F}$. There exists $\Gamma_i \in\in \mathcal{B}_A$ that is of the form Γ', B', Δ with an indecomposable Γ'.
We have that B' can not be of the form

$$\neg\exists x A(x) \quad \text{or} \quad \neg\forall x A(x) \tag{10.26}$$

for if (refn-Q) is in $M\mathcal{F}$, then also formula $\forall x\neg A(x)$ or $\exists x\neg A(x)$ is in $M\mathcal{F}$ and the orders of the two formulas are equal.

We carry the same order argument and show that B' can not be of the form

$$(A \cup B), \neg(A \cup B), (A \cap B), \neg(A \cap B), (A \Rightarrow B), \neg(A \Rightarrow B), \neg\neg A, \forall x A(x). \tag{10.27}$$

The formula B' can't be of the form

$$\exists x B(x) \tag{10.28}$$

since then there exists term t and j such that $i \leq j$, $B'(t)$ appears in Γ_j and the formula $B(t)$ satisfies (10.24). Thus $B(t) \in M\mathcal{F}$ and $ordB(t) < ordB'$. This contradicts the definition of B'.

Since B' is not of the form (10.26), (10.27), (10.28), B' is indecomposable. Thus $B' \in L\mathcal{F}$ (10.20), and consequently by (10.21),

$$\mathcal{M} \not\models B'.$$

On the other hand B' by definition is in the set $M\mathcal{F}$ and hence is one o the formulas satisfying (10.24), i.e.

$$\mathcal{M} \not\models B'.$$

This contradiction proves that (10.23) $\mathcal{M} \not\models A$ and hence we proved

$$\not\models A.$$

This **ends** the proof of the Completeness Theorem 10.4 for **QRS**.

10.4 Skolemization and Clauses

The resolution proof system for propositional and predicate logic operates on a set of **clauses** as a basic expressions and uses a **resolution rule** as the only rule of inference.

The goal of this part is to define an effective *process of transformation* of any formula A of a predicate language $\mathcal{L} = \mathcal{L}_{\{\neg,\cup,\cap,\Rightarrow\}}(\mathbf{P}, \mathbf{F}, \mathbf{C})$ into a certain corresponding set of clauses $\mathbf{C}_A$. This is done in two stages.
S1. We convert any formula A of $\mathcal{L}$ into an *open* formula A^* of a language $\mathcal{L}^*$ by a process of *elimination of quantifiers* from the original $\mathcal{L}$. The method is due to T. Skolem (1920) and is called Skolemization. The resulting formula A^* is equisatisfiable with A: it is satisfiable if and only if the original one is satisfiable (Skolem Theorem 10.11).

The stage S1. is performed as the first step in a Resolution based automated theorem prover and is described in section 10.4.1.

S2. We define a proof system $\mathbf{QRS}^*$ based on the language $\mathcal{L}^*$ and use it transform any formula A^* of $\mathcal{L}^*$ into an logically equivalent set of clauses $\mathbf{C}_{A^*}$ (theorem 10.13).

The final result of stages S1. and S1 is the set $\mathbf{C}_A$ of clauses corresponding to the formula A, called a *clausal form* of A (theorem 10.6.

The *transformation process* for any propositional formula A into its *logically equivalent* set $\mathbf{C}_A$ of clauses follows directly from the use of the propositional system **RS** (theorem 10.5).

.

Definition 10.9 (Clauses)

Given a language $\mathcal{L}$, propositional or predicate.

1. A **literal** *as an atomic, or a negation of an atomic formula of $\mathcal{L}$. We denote by LT the set of all literals of $\mathcal{L}$.*

2. A **clause** *$\mathcal{C}$ is a* **finite set** *of literals.Empty clause is denoted by $\{\}$.*

3. We denote by $\mathbf{C}$ any finite set of all clauses.

$$\mathbf{C} = \{\mathcal{C}_1,\ \mathcal{C}_2,\ \dots\ \mathcal{C}_n\},$$

for any $n \geq 0$.

Definition 10.10

Given a propositional or predicate language L, and a sequence $\Gamma \in LT^*$. A **clause determined** *by Γ is a set form out of all elements of the sequence Γ We we denote it by $\mathcal{C}_\Gamma$.*

Example 10.7

In particular,
1. if $\Gamma_1 = a, a, \neg b, c, \neg b, c$ *and* $\Gamma_2 = \neg b, c, a$*, then* $\mathcal{C}_{\Gamma_1} = \mathcal{C}_{\Gamma_2}\{a, c, \neg b\}$.

2. If $\Gamma_1 = \neg P(x_1), \neg R(x_1, y), P(x_2), \neg P(x_1), \neg R(x_1, y), P(x_2)$ *and*
$\Gamma_2 = \neg P(x_1), \neg R(x_1, y), P(x_2)$*, then* $\mathcal{C}_{\Gamma_1} = \mathcal{C}_{\Gamma_2}\{\neg P(x_1), \neg R(x_1, y), P(x_2)\}$.

The semantics for clauses is basically the same as for the sequences. We define it as follows.

Definition 10.11 *Given a propositional or predicate language L. For any clause* $\mathcal{C}$*, write* $\delta_{\mathcal{C}}$ *for a* disjunction *of all literals in* $\mathcal{C}$.

Definition 10.12 (Clauses Semantics)

Let $\mathcal{M} = [M, I]$ *be a structure for a predicate language* $\mathcal{L}$*, or a truth assignment* v *in case of* $\mathcal{L}$ *propositional.*

$\mathcal{M}$ *is called a* **model** *for a clause* $\mathcal{C}$ *(predicate or propositional), (*$\mathcal{M} \models \mathcal{C}$*) if and only if*

$$\mathcal{M} \models \delta_{\mathcal{C}}.$$

$\mathcal{M}$ *is called a* **model** *for a set* $\mathbf{C}$ *of clauses (*$\mathcal{M} \models \mathbf{C}$*) if and only if*

$$\mathcal{M} \models \delta_{\mathcal{C}} \quad \text{for all clauses } \mathcal{C} \in \mathbf{C}.$$

Definition 10.13 (Equivalence)

A formula A of a language $\mathcal{L}$ *is* **equivalent** *with a set set* $\mathbf{C}$ *of clauses* $(A \equiv \mathbf{C})$ *if and only if* $A \equiv \sigma_{\mathbf{C}}$*, where* $\sigma_{\mathbf{C}}$ *is a* conjunction *of all formulas* $\delta_{\mathcal{C}}$ *for all clauses* $\mathcal{C} \in \mathbf{C}$.

Theorem 10.5 (Formula-Clauses Equivalency)

For any formula A of a propositional language $\mathcal{L}$*, there is an effective procedure of generating a corresponding set* $\mathbf{C}_A$ *of clauses such that*

$$A \equiv \mathbf{C}_A \tag{10.29}$$

Proof
Let $\mathcal{L} = \mathcal{L}_{\{\neg, \cup, \cap, \Rightarrow\}}$. Given $A \in \mathcal{F}$, we use the **RS** system (chapter 6) to build the decomposition tree $\mathbf{T}_A$. We form clauses out of the leaves of the tree $\mathbf{T}_A$, i.e. for every leaf L we create a clause $\mathcal{C}_L$ determined by L (definition 10.10). We put

$$\mathbf{C}_A = \{\mathcal{C}_L : \quad L \text{ is a leaf of } \quad \mathbf{T}_A\}.$$

Directly from the strong soundness (10.13) of rules of inference of **RS** and the definition 10.13 we get $A \equiv \mathbf{C}_A$. This ends the **proof** for the propositional case.

Consider a decomposition tree of a formula $(((a \Rightarrow b) \cap \neg c) \cup (a \Rightarrow c))$

$$\mathbf{T}_A$$

$$\begin{tikzcd}[row sep=small]
& (((a \Rightarrow b) \cap \neg c) \cup (a \Rightarrow c)) \arrow[d, dash, "(\cup)"] & \\
& ((a \Rightarrow b) \cap \neg c), (a \Rightarrow c) \arrow[dl, dash] \arrow[dr, dash, "(\cap)"] & \\
(a \Rightarrow b), (a \Rightarrow c) \arrow[d, dash, "(\Rightarrow)"] & & \neg c, (a \Rightarrow c) \arrow[d, dash, "(\Rightarrow)"] \\
\neg a, b, (a \Rightarrow c) \arrow[d, dash, "(\Rightarrow)"] & & \neg c, \neg a, c \\
\neg a, b, \neg a, c & &
\end{tikzcd}$$

Example 10.8

For the formula $(((a \Rightarrow b) \cap \neg c) \cup (a \Rightarrow c))$ and the tree $\mathbf{T}_A$, the leaves are $L_1 = \neg a, b, \neg a, c$ and $\mathcal{C}_{L_1} = \{\neg a, b, c\}$ and $L_2 = \neg c, \neg a, c$ and $\mathcal{C}_{L_2} = \{\neg c, \neg a, c\}$. The set of clauses is

$$\mathbf{C}_A = \{\{\neg a, b, c\},\ \{\neg c, \neg a, c\}\}.$$

By theorem 10.5, $A \equiv \mathbf{C}_A$. Semantically it means, by definition 10.13,

$$A \equiv (((\neg a \cup b) \cup c) \cap ((\neg c \cup \neg a) \cup c)).$$

Theorem 10.6 (Clausal Form)

For any formula A of a predicate language $\mathcal{L}$, there is an effective procedure of generating an open formula A^* of a quantifiers free *language $\mathcal{L}^*$ and a set $\mathbf{C}_{A^*}$ of clauses such that*

$$A^* \ \equiv \ \mathbf{C}_{A^*}. \tag{10.30}$$

The set $\mathbf{C}_{A^*}$ of clauses of $\mathcal{L}^*$ with the property (10.30) is called a **clausal form** *of the formula A of $\mathcal{L}$.*

Proof

Given a formula A of a language $\mathcal{L}$. The open formula A^* of the quantifiers free language $\mathcal{L}^*$ is obtained by the Skolemization process. The effectiveness and correctness of the process follows from **PNF** theorem 10.10 and **Skolem** theorem 10.11 described in section 10.4.1.

As the next step, we define (section 10.4.2) a proof system $\mathbf{QRS}^*$(10.43) based on the quantifiers free language $\mathcal{L}^*$. The system $\mathbf{QRS}^*$ is a version of the system **QRS** (10.8) restricted to its Propositional Rules. At this point we carry the proof in a similar way to the proof in the propositional case (theorem 10.5). Namely, for any formula A^* of $\mathcal{L}^*$ obtained from A of $\mathcal{L}$ we construct its the decomposition tree $\mathbf{T}_{A^*}$. We form clauses out of the leaves of the tree $\mathbf{T}_{A^*}$, i.e. for every leaf L we create a clause $\mathcal{C}_L$ determined by L and we put

$$\mathbf{C}_{A^*} = \{\mathcal{C}_L : \quad L \text{ is a leaf of } \quad \mathbf{T}_{A^*}\}.$$

This is the **clausal form** of the formula A of $\mathcal{L}$ by theorem 10.13 proved in section 10.4.2. To complete the proof we need now to develop results of the section 10.4.1 and the section 10.4.2.

10.4.1 Prenex Normal Forms and Skolemization

We remind the following important notion.

Term t is free for x in $A(x)$. Let $A(x) \in \mathcal{F}$ and t be a term, $A(t)$ be a result of substituting t for all free occurrences of x in $A(x)$.

We say that t **is free for** x **in** $A(x)$, if no occurrence of a variable in t becomes a bound occurrence in $A(t)$.

In particular, if $A(x), A(x_1, x_2, ..., x_n) \in \mathcal{F}$ and $t, t_1, t_2, ..., t_n \in \mathbf{T}$, then

$$A(x/t), \quad A(x_1/t_1, x_2/t_2, ..., x_n/t_n)$$

or, more simply just

$$A(t), \quad A(t_1, t_2, ..., t_n)$$

denotes the result of replacing all occurrences of the free variables $x, x_1, x_2, ..., x_n$, by the terms $t, t_1, t_2, ..., t_n$, respectively, assuming that $t, t_1, t_2, ..., t_n$ are free for x, $x_1, x_2, ..., x_n$, respectively, in A.

The assumption that t **is free for** x **in** $A(x)$ while substituting t for x, is important because otherwise we would distort the meaning of $A(t)$. This is illustrated by the following example.

Example 10.9

Let $t = y$ and $A(x)$ be

$$\exists y(x \neq y).$$

Obviously t is not free for y in A. The substitution of t for x produces a formula $A(t)$ of the form

$$\exists y(y \neq y),$$

which has a different meaning than $\exists y(x \neq y)$.

Here are more examples illustrating the notion: *t is free for x in $A(x)$*.

Example 10.10

Let $A(x)$ be a formula

$$(\forall y P(x,y) \cap Q(x,z))$$

and t be a term $f(x,z)$, i.e. $t = f(x,z)$.

None of the occurrences of the variables x, z of t is bound in $A(t)$, hence we say that $t = f(x,z)$ is **free for** *x in $(\forall y P(x,y) \cap Q(x,z))$.*

Substituting t on a place of x in $A(x)$ we obtain a formula $A(t)$ of the form

$$(\forall y P(f(x,z),y) \cap Q(f(x,z),z)).$$

Example 10.11

Let $A(x)$ be a formula

$$(\forall y P(x,y) \cap Q(x,z))$$

The term $t = f(y,z)$ is **not free** *for x in $A(x)$ because substituting $t = f(y,z)$ on a place of x in $A(x)$ we obtain now a formula $A(t)$ of the form*

$$(\forall y P(f(y,z),y) \cap Q(f(y,z),z))$$

which contain a bound occurrence of the variable y of t ($\forall y P(f(y,z),y)$).

The other occurrence $(Q(f(y,z),z))$ of y is free, but it is not sufficient, as for term to be free for x, all occurrences of its variables has to be free in $A(t)$.

Another important notion we will use here is the following notion of *similarity of formulas.*

Intuitively, we say that $A(x)$ and $A(y)$ are similar if and only if $A(x)$ and $A(y)$ are the same except that $A(x)$ has free occurrences of x in exactly those places where $A(y)$ has free occurrences of y.

Example 10.12

The formulas $\exists z(P(x,z) \Rightarrow Q(x))$ and $\exists z(P(y,z) \Rightarrow Q(y))$ are similar.

The formal definition of this notion follows.

Definition 10.14 (Similarity)

Let x and y be two different variables. We say that the formulas $A(x)$ and $A(x/y)$ are **similar** *and denote it by*

$$A(x) \sim A(x/y)$$

if and only if y is free for x in $A(x)$ and $A(x)$ **has no** *free occurrences of y.*

Example 10.13

The formulas $A(x)$*:* $\exists z(P(x,z) \Rightarrow Q(x,y))$ *and* $A(x/y)$*:* $\exists z(P(y,z) \Rightarrow Q(y,y))$ *are* **not similar***;* y *is free for* x *in* $A(x)$*, but the formula* $A(x/y)$ **has** *a free occurrence of* y*.*

Example 10.14

The formulas $A(x)$*:* $\exists z(P(x,z) \Rightarrow Q(x,y))$ *and* $A(x/w)$*:* $\exists z(P(w,z) \Rightarrow Q(w,y)$ **are similar***;* w *is free for* x *in* $A(x)$ *and the formula* $A(x/w)$ *has no free occurrence of* w*.*

Directly from the definition we get the following.

Lemma 10.1

For any formula $A(x) \in \mathcal{F}$*, if* $A(x)$ *and* $A(x/y)$ *are similar* $A(x) \sim A(y)$*, then*

$$\forall x A(x) \equiv \forall y A(y),$$

$$\exists x A(x) \equiv \exists y A(y).$$

We prove, by the induction on the number of connectives and quantifiers in a formula A the following.

Theorem 10.7 (Replacement Theorem)

For any formulas $A, B \in \mathcal{F}$*, if* B *is a sub-formula of* A*, if* A^* *is the result of replacing zero or more occurrences of* B *in* A *by a formula* C*, and* $B \equiv C$*, then* $A \equiv A^*$*.*

Directly from lemma 10.1 and replacement theorem 10.7 we get that the following theorem holds.

Theorem 10.8 (Change of Bound Variables)

For any formula $A(x), A(y), B \in \mathcal{F}$*, if* $A(x)$ *and* $A(x/y)$ *are similar, i.e.* $A(x) \sim A(y)$*, and the formula* $\forall x A(x)$ *or* $\exists x A(x)$ *is a sub-formula of* B*, and* B^* *is the result of replacing zero or more occurrences of* $A(x)$ *in* B *by a formula* $\forall y A(y)$ *or* $\exists y A(y)$*, then* $B \equiv B^*$*.*

Definition 10.15 (Naming Variables Apart)

We say that a formula B *has its variables* **named apart** *if no two quantifiers in* B *bind the same variable and no bound variable is also free.*

We can now use theorem 10.8 to prove its more general version.

Theorem 10.9 (Naming Variables Apart) *Every formula $A \in \mathcal{F}$ is logically equivalent to one in which all variables are named apart.*

We use the above theorems plus the equational laws for quantifiers (10.31) to prove, as a next step a so called a Prenex Form Theorem 10.10.
In order to do so we first we define an important notion of **prenex normal form** of a formula.

Definition 10.16 (Closure of a Formula)

By a **closure** *of a formula A we mean a* **closed** *formula A' obtained from A prefixing in universal quantifiers all those variables that a free in A; i.e. if $A(x_1, \ldots, x_n)$ then $A' \equiv A$ is*

$$\forall x_1 \forall x_2 .... \forall x_n A(x_1, x_2, ..., x_n)$$

Example 10.15

Let A be a formula $(P(x,y) \Rightarrow \neg\exists z\ R(x,y,z))$, its **closure** $A' \equiv A$ *is* $\forall x \forall y (P(x,y) \Rightarrow \neg\exists z\ R(x,y,z))$.

Definition 10.17 (Prenex Normal Form)

Any formula A of the form

$$Q_1 x_1 Q_2 x_2 .... Q_n x_n B$$

where each Q_i is a universal or existential quantifier, i.e. for all $1 \leq i \leq n$, $Q_i \in \{\exists, \forall\}$, $x_i \neq x_j$ for $i \neq j$, and B **contains no quantifiers***, is said to be in prenex normal form* **(PNF)**.
We include the case $n = 0$ when there are no quantifiers at all.

We assume that the formula A in **PNF** is always **closed**. If it is not closed we form its **closure** (definition 10.16) instead. We prove that, for every formula A, we can effectively construct a formula B that is in the prenex normal form **PNF** and $A \equiv B$.

Theorem 10.10 (PNF Theorem)

There is an effective procedure for transforming any formula $A \in \mathcal{F}$ into a logically equivalent formula A' in the prenex normal form **PNF**.

Proof
We use theorems 10.7, 10.8, 10.9, theorem 10.15, and the following logical equivalences (10.31) proved in chapter 2.

Equational Laws of Quantifiers (10.31)

$$\forall x(A(x) \cup B) \equiv (\forall x A(x) \cup B) \tag{10.32}$$

$$\forall x(A(x) \cap B) \equiv (\forall x A(x) \cap B) \tag{10.33}$$

$$\exists x(A(x) \cup B) \equiv (\exists x A(x) \cup B) \tag{10.34}$$

$$\exists x(A(x) \cap B) \equiv (\exists x A(x) \cap B) \tag{10.35}$$

$$\forall x(A(x) \Rightarrow B) \equiv (\exists x A(x) \Rightarrow B) \tag{10.36}$$

$$\exists x(A(x) \Rightarrow B) \equiv (\forall x A(x) \Rightarrow B) \tag{10.37}$$

$$\forall x(B \Rightarrow A(x)) \equiv (B \Rightarrow \forall x A(x)) \tag{10.38}$$

$$\exists x(B \Rightarrow A(x)) \equiv (B \Rightarrow \exists x A(x)) \tag{10.39}$$

where B is a formula such that B *does not contain any free occurrence* of x.

The formal procedure is defined by induction on the number k of occurrences of connectives and quantifiers in A. We show now how it works in some particular cases.

Exercise 10.1

Find a prenex normal form **PNF** *of a formula A:* $(\forall x(P(x) \Rightarrow \exists x Q(x))$.

Solution

We find **PNF** in the following steps.

Step 1: Rename Variables Apart

By the theorem 10.8 we can make all **bound variables** in A different, i.e. we transform A into an equivalent formula A'

$$\forall x(P(x) \Rightarrow \exists y Q(y)).$$

Step 2: Pull out Quantifiers

We apply the equational law

$$(C \Rightarrow \exists y Q(y)) \equiv \exists y \ (C \Rightarrow Q(y))$$

to the sub-formula $B : (P(x) \Rightarrow \exists y Q(y))$ of A' for $C = P(x)$, as P(x) does not contain the variable y. We get its equivalent formula $B^* : \exists y(P(x) \Rightarrow Q(y))$. We substitute now B^* on place of B in A' and get a formula

$$A'' : \quad \forall x \exists y(P(x) \Rightarrow Q(y))$$

such that $A'' \equiv A' \equiv A$.

A'' is a required prenex normal form **PNF** for A.

Exercise 10.2 *Find a prenex normal form* **PNF** *formula A' for the formula A:*

$$(\exists x \forall y \ R(x,y) \Rightarrow \forall y \exists x \ R(x,y))$$

Solution
Step 1: Rename Variables Apart
Take a sub- formula $B(x,y) : \ \forall y \exists x \ R(x,y)$ of A, get $B(x/z, y/w) : \forall z \exists w \ R(z,w)$ and replace B(x,y) by $B(x/z, y/w)$ in A and get

$$(\exists x \forall y \ R(x,y) \Rightarrow \forall z \exists w \ R(z,w))$$

Step 2: Pull out quantifiers
We use corresponding equational laws of quantifiers (11.46), 11.47) to pull out first quantifiers $\exists x \forall y$ and get the following

$$A' : \quad \forall x \exists y((R(x,y) \Rightarrow \ \forall z \exists w \ R(z,w))),$$

such that $A' \equiv A$. Now we pull quantifiers $\forall z \exists w$ in $(R(x,y) \Rightarrow \ \forall z \exists w \ R(z,w))$ and get the prenex normal form **PNF** formula

$$A'' : \quad \forall x \exists y \forall z \exists w \ ((R(x,y) \Rightarrow \ R(z,w))),$$

such that $A'' \equiv A' \equiv A$.

Observe we can also perform a different **Step 2** by pulling first the quantifiers $\forall z \exists w$ and then quantifiers $\forall x \exists y$ and obtain **another PNF** A''' for A:

$$A''' : \quad \forall z \exists w \forall x \exists y \ (R(x,y) \Rightarrow \ R(z,w)).$$

We will show now how any formula A in its prenex normal form **PNF** we can transformed it into a corresponding **open formula** A^*.

The open formula A^* belongs to a richer language then the initial language to which the formula A belongs. The transformation process **adds** new constants, called **Skolem constants**, and new function symbols, called **Skolem function symbols** to the initial language.

The whole process is called the skolemisation of the initial language $\mathcal{L}$, the such build extension of the initial language is called a **Skolem extension** of $\mathcal{L}$,.

Skolem Procedure of Elimination of Quantifiers (10.40)

Given a formula A of the language $\mathcal{L} = \mathcal{L}_{\{\neg,\cup,\cap,\Rightarrow\}}(\mathbf{P}, \mathbf{F}, \mathbf{C})$ in its prenex normal form **PNF**, i.e.

$$A = Q_1 x_1 Q_2 x_2 \dots Q_n x_n B(x_1, x_2, \dots x_n) \tag{10.41}$$

where each Q_i is a universal or existential quantifier, i.e. for all $1 \leq i \leq n$, $Q_i \in \{\exists, \forall\}$, $x_i \neq x_j$ for $i \neq j$, and $B(x_1, x_2, \dots x_n)$ contains no quantifiers.

We describe now a **procedure** of elimination of all quantifiers from the formula A (10.41) and hence transforming it into a corresponding **open formula** A^*.

We assume that the formula $A = Q_1 x_1 Q_2 x_2 \dots Q_n x_n B(x_1, x_2, \dots x_n)$ is **closed**. If it is not closed we form its **closure** instead.

We considerer 3 cases.
Case 1
All quantifiers Q_i for $1 \leq i \leq n$ are **universal**, i.e. the **closed formula** A is

$$\forall x_1 \forall x_2 \dots \forall x_n B(x_1, x_2, \dots, x_n)$$

We replace the formula A by the **open formula** A^*:

$$B(x_1, x_2, \dots., x_n).$$

Case 2
All quantifiers Q_i for $1 \leq i \leq n$ are **existential**, i.e. the **closed formula** A is

$$\exists x_1 \exists x_2 .... \exists x_n B(x_1, x_2, \dots x_n)$$

We replace the formula A by the **open formula** A^*:

$$B(c_1, c_2, \dots., c_n)$$

where $c_1, c_2, \dots., c_n$ and **new** individual constants, all different, **added** to our original language $\mathcal{L}$. We call such constants added to the language **Skolem constants**

Case 3
The quantifiers are **mixed** . We assume that A is **closed**. If it is not closed we

form its **closure** instead. We eliminate quantifiers one by one and step by step depending on first, and consecutive quantifiers.

Given a **closed PNF** formula A

$$Q_1x_1Q_2x_2 \dots Q_nx_nB(x_1, x_2, \dots x_n)$$

Step 1 Elimination of Q_1x_1
We have two possibilities for the first quantifier Q_1x_1, namely **P1** Q_1x_1 is **universal** or **P2** Q_1x_1 is **existential**.

Consider **P1**
First quantifier in A is universal, i. e. A is

$$\forall x_1Q_2x_2 \dots Q_nx_nB(x_1, x_2, \dots x_n)$$

We **replace** A by a formula A_1 :

$$Q_2x_2 \dots Q_nx_nB(x_1, x_2, \dots x_n)$$

We have **eliminated** the quantifier Q_1 in this case.

Consider **P2**
First quantifier in A is **existential**, i. e. A is

$$\exists x_1Q_2x_2 \dots Q_nx_nB(x_1, x_2, \dots x_n)$$

We **replace** A by a formula A_1 :

$$Q_2x_2 \dots Q_nx_nB(b_1, x_2, \dots x_n)$$

where b_1 is a **new** constant symbol **added** to our original language $\mathcal{L}$. We call such constant symbol added to the language **Skolem constant** symbol.
We have **eliminated** the quantifier Q_1 in this case. We have covered all cases and this **ends** the **Step 1**.

Step 2 Elimination of Q_2x_2.

Consider now the **PNF** formula A_1 from **Step1- P1**

$$Q_2x_2 \dots Q_nx_nB(x_1, x_2, \dots x_n)$$

Remark that the formula A_1 might not be closed.

We have again two possibilities for elimination of the quantifier Q_2x_2, namely **P1** Q_2x_2 is **universal** or **P2** Q_2x_2 is **existential**.

Consider **P1**
First quantifier in A_1 is **universal**, i.e. A_1 is

$$\forall x_2 Q_3 x_3 \ldots Q_n x_n B(x_1, x_2, x_3, \ldots x_n)$$

We replace A_1 by the following A_2

$$Q_3 x_3 \ldots Q_n x_n B(x_1, x_2, x_3, \ldots x_n)$$

We have **eliminated** the quantifier Q_2 in this case.

Consider **P2**
First quantifier in A_1 is **existential**, i.e. A_1 is

$$\exists x_2 Q_3 x_3 \ldots Q_n x_n B(x_1, x_2, x_3, \ldots x_n)$$

Observe that now the variable x_1 is a **free** variable in $B(x_1, x_2, x_3, \ldots x_n)$ and hence in A_1.
We replace A_1 by the following A_2

$$Q_3 x_3 \ldots Q_n x_n B(x_1, f(x_1), x_3, \ldots x_n)$$

where f is a **new** one argument functional symbol **added** to our original language $\mathcal{L}$. We call such functional symbols added to the original language **Skolem** functional symbols.
We have **eliminated** the quantifier Q_2 in this case.

Consider now the **PNF** formula A_1 from **Step1 - P2**

$$Q_2 x_2 Q_3 x_3 \ldots Q_n x_n B(b_1, x_2, \ldots x_n)$$

Again we have two cases.

Consider **P1**
First quantifier in A_1 is **universal**, i.e. A_1 is

$$\forall x_2 Q_3 x_3 \ldots Q_n x_n B(b_1, x_2, x_3, \ldots x_n)$$

We replace A_1 by the following A_2

$$Q_3 x_3 \ldots Q_n x_n B(b_1, x_2, x_3, \ldots x_n)$$

We have **eliminated the quantifier** Q_2 in this case.
Consider **P2**
First quantifier in A_1 is **existential**, i.e. A_1 is

$$\exists x_2 Q_3 x_3 \ldots Q_n x_n B(b_1, x_2, x_3, \ldots x_n)$$

We replace A_1 by A_2

$$Q_3x_3 \dots Q_nx_nB(b_1, b_2, x_3, \dots x_n)$$

where $b_2 \neq b_1$ is a new Skolem constant symbol **added** to our original language $\mathcal{L}$.
We have **eliminated** the quantifier Q_2 in this case. We have covered all cases and this **ends** the **Step 2**. **Step 3** Elimination of Q_3x_3

Let's now consider, as an **example** formula A_2 from **Step 2; P1** i.e. the formula

$$Q_3x_3 \dots Q_nx_nB(x_1, x_2, x_3, \dots x_n)$$

We have again 2 choices to consider, but will describe only the following.

P2 First quantifier in A_2 is **existential**, i. e. A_2 is

$$\exists x_2Q_4x_4 \dots Q_nx_nB(x_1, x_2, x_3, x_4, \dots x_n)$$

Observe that now the variables x_1, x_2 are **free** variables in $B(x_1, x_2, x_3, \dots x_n)$ and hence in A_2.

We replace A_2 by the following A_3

$$Q_4x_3 \dots Q_nx_nB(x_1, x_2, g(x_1, x_2), x_4 \dots x_n)$$

where g is a **new** two argument functional symbol **added** to our original language $\mathcal{L}$.
We have **eliminated** the quantifier Q_3 in this case.

Step i
At each **Step i**, for $1 \leq i \leq n$), we build a **binary tree** of possibilities:
P1 Q_ix_i is **universal** or **P2** Q_ix_i is **existential** and as result we obtain a formula A_i with one less quantifier. The elimination process builds a sequence of formulas

$$A,\ A_1,\ A_2,\ \dots,\ A_n = A^*$$

where the formula A belongs to our original language

$$\mathcal{L} = \mathcal{L}_{\{\neg,\cup,\cap,\Rightarrow\}}(\mathbf{P}, \mathbf{F}, \mathbf{C}),$$

the formula A^* belongs to its **Skolem extension** language (10.42) defined as follows.

Definition 10.18

The language $\mathcal{L}^*$ obtained from $\mathcal{L}$ by the quantifiers elimination procedure (10.40) is is called a **Skolem extension** *of $\mathcal{L}$.*

$$\mathcal{L}^* = \mathcal{L}_{\{\neg,\cup,\cap,\Rightarrow\}}(\mathbf{P}, \mathbf{F} \cup S\mathbf{F},\ \mathbf{C} \cup S\mathbf{C}). \tag{10.42}$$

Observe that in the elimination process (10.40) a **universal quantifier** introduces free variables in the formula $B(x_1, x_2, \dots x_n)$. The **elimination** of an existential quantifier that follows universal quantifiers introduces a **new** functional symbol with number of arguments equal the number of universal quantifiers preceding it.

The resulting is an **open** formula A^* of **Skolem extension** language $\mathcal{L}^*$. By **PNF** theorem 10.10, for any formula A of $\mathcal{L}$ its **PNF** formula (10.41) exists and is logically equivalent with A. We hence introduce the following definition.

Definition 10.19 (Skolemization)

Given a formula A of $\mathcal{L}$.
A formula A^* of the Skolem extension language $\mathcal{L}^*$ (10.42) obtained from a **PNF** *form of A by the Skolem Procedure (10.40) is called a* **Skolem form** *of the formula A and the process obtaining it is called a* **Skolemization** *of A.*

Exercise 10.3 *Let A be a* **PNF** *formula*

$$\forall y_1 \exists y_2 \forall y_3 \exists y_4 \; B(y_1, y_2, y_3, y_4, y_4).$$

Find the **Skolem form** *of A (the formula $B(y_1, y_2, y_3, y_4, y_4)$ is quantifiers free).*

Solution
We eliminate $\forall y_1$ and get a formula A_1

$$\exists y_2 \forall y_3 \exists y_4 \; B(y_1, y_2, y_3, y_4).$$

We eliminate $\exists y_2$ by replacing y_2 by $h(y_1)$ where h is a **new** one argument functional symbol **added** to our original language $\mathcal{L}$.
We get a formula A_2

$$\forall y_3 \exists y_4 \; B(y_1, h(y_1), y_3, y_4).$$

We eliminate $\forall y_3$ and get a formula A_3

$$\exists y_4 \; B(y_1, h(y_1), y_3, y_4).$$

We eliminate $\exists y_4$ by replacing y_4 by $f(y_1, y_3)$, where f is a **new** two argument functional symbol **added** to our original language $\mathcal{L}$.
We get a formula A_4 that is our resulting **open** formula A^*

$$B(y_1, h(y_1), y_3, f(y_1, y_3)).$$

Exercise 10.4

Let now A be a **PNF** *formula*

$$\exists y_1 \forall y_2 \forall y_3 \exists y_4 \exists y_5 \forall y_6 \; B(y_1, y_2, y_3, y_4, y_4, y_5, y_6)$$

Find the **Skolem form** *of A (the formula $B(y_1, y_2, y_3, y_4, y_4, y_5)$ is quantifiers free).*

Solution
We eliminate $\exists y_1$ and get a formula A_1

$$\forall y_2 \forall y_3 \exists y_4 \exists y_5 \forall y_6 \ B(b_1, y_2, y_3, y_4, y_4, y_5, y_6)$$

where b_1 is a **new** constant symbol **added** to our original language $\mathcal{L}$.
We eliminate $\forall y_2, forall y_3$ and get a formulas A_2, A_3; here is the formula A_3

$$\exists y_4 \exists y_5 \forall y_6 \ B(b_1, y_2, y_3, y_4, y_4, y_5, y_6)$$

We eliminate $\exists y_4$ and get a formula A_4

$$\exists y_5 \forall y_6 \ B(b_1, y_2, y_3, g(y_2, y_3), y_5, y_6)$$

where g is a **new** two argument functional symbol **added** to our original language $\mathcal{L}$.
We eliminate $\exists y_5$ and get a formula A_5

$$\forall y_6 \ B(b_1, y_2, y_3, g(y_2, y_3), h(y_2, y_3), y_6)$$

where h is a **new** two argument functional symbol **added** to our original language $\mathcal{L}$.
We eliminate $\forall y_6$ and get a formula A_6 that is the resulting **open** formula A^*

$$B(b_1, y_2, y_3, g(y_2, y_3), h(y_2, y_3), y_6).$$

The **correctness** of the Skolemization process is established by the Skolem theorem 10.11. It states informally that the formula A^* obtained from a formula A via the Skolemization is satisfiable if and only if the original one is satisfiable. We define this notion formally as follows.

.

Definition 10.20 (Equisatisfiable)

For any formulas A of $\mathcal{L}$ and B of the Skolem extension $\mathcal{L}^*$ (10.42) of $\mathcal{L}$, we say that A and B are **equisatisfiable** *if and only if the following conditions are satisfied.*

1. *Any structure $\mathcal{M}$ of $\mathcal{L}$ can be extended to a structure $\mathcal{M}^*$ of $\mathcal{L}^*$ and following implication holds.*
$$\text{If } \mathcal{M} \models A, \text{ then } \mathcal{M}^* \models B.$$

2. *Any structure $\mathcal{M}^*$ of $\mathcal{L}^*$ can be restricted to a structure $\mathcal{M}$ of $\mathcal{L}$ and following implication holds.*
$$\text{If } \mathcal{M}^* \models B, \text{ then } \mathcal{M} \models A.$$

Theorem 10.11 (Skolem Theorem)

*Let $\mathcal{L}^*$ be the Skolem extension (10.42) of a language $\mathcal{L}$.
Any formula A of $\mathcal{L}$ and its* Skolem form *A^* of $\mathcal{L}^*$ are* **equisatisfiable**.

10.4.2 Clausal Form of Formulas

Let $\mathcal{L}^*$ be the Skolem extension of $\mathcal{L}$, i.e. $\mathcal{L}^*$ does not contain quantifiers. We define a proof system $\mathbf{QRS}^*$ as an open formulas language version of **QRS** that includes only its Group 1: Propositional Rules of (10.7).

We denote the set of formulas of $\mathcal{L}^*$ by $O\mathcal{F}$ to stress the fact that all its formulas are *open* and define $\mathbf{QRS}^*$ formally as follows.

$$\mathbf{QRS}^* = (\mathcal{L}^*,\ \mathcal{E},\ \ LA,\ \ \mathcal{R}), \tag{10.43}$$

where $\mathcal{E} = \{\Gamma :\ \ \Gamma \in O\mathcal{F}^*\}$, LA is defined by (10.6), and $\mathcal{R}$ contains Group 1: Propositional Rules (10.7):

$$(\cup)\ \frac{\Gamma^{'}, A, B, \Delta}{\Gamma', (A \cup B), \Delta}, \qquad (\neg\cup)\ \frac{\Gamma^{'}, \neg A, \Delta \quad : \quad \Gamma^{'}, \neg B, \Delta}{\Gamma', \neg(A \cup B), \Delta}$$

$$(\cap)\ \frac{\Gamma^{'}, A, \Delta \quad ; \quad \Gamma^{'}, B, \Delta}{\Gamma', (A \cap B), \Delta}, \qquad (\neg\cap)\ \frac{\Gamma^{'}, \neg A, \neg B, \Delta}{\Gamma', \neg(A \cap B), \Delta}$$

$$(\Rightarrow)\ \frac{\Gamma^{'}, \neg A, B, \Delta}{\Gamma', (A \Rightarrow B), \Delta}, \qquad (\neg\ \Rightarrow)\ \frac{\Gamma^{'}, A, \Delta \quad : \quad \Gamma^{'}, \neg B, \Delta}{\Gamma', \neg(A \Rightarrow B), \Delta}$$

$$(\neg\,\neg)\ \frac{\Gamma^{'}, A, \Delta}{\Gamma', \neg\neg A, \Delta}$$

where $\Gamma^{'} \in LT^*, \Delta \in O\mathcal{F}^*, A, B \in O\mathcal{F}$.

For any formula $A \in O\mathcal{F}$ we define, as we did in chapter 6 its decomposition tree $\mathbf{T}_A$ as follows.

Decomposition tree $\mathbf{T}_A$
Step 1. The formula A is the **root** of $\mathbf{T}_A$ and for any node Δ of the tree we follow the steps bellow.
Step 2. If Δ in *indecomposable*, then Δ becomes a *leaf* of the tree.
Step 3. If Δ is decomposable, then we traverse Δ from left to right to identify the first *decomposable formula B*. In case of a one premisses rule we put is premise as a *leaf;* in case of a two premisses rule we put its left and right premisses as the left and right *leaves*, respectively.

Step 4. We repeat steps 2 and 3 until we obtain only leaves.

We adopt the definition 10.12 to $\mathbf{QRS}^*$ and the language $\mathcal{L}^*$.

Definition 10.21 (Semantics)

For any sequence Γ of formulas of $\mathcal{L}^*$, any structure $\mathcal{M} = [M, I]$ for $\mathcal{L}^*$,

$$\mathcal{M} \models \Gamma \quad \textit{if and only if} \quad \mathcal{M} \models \delta_\Gamma,$$

where δ_Γ denotes a disjunction *of all formulas in Γ.*

The semantics for clauses is basically the same as for the sequences. We define it, after definition 10.5, as follows.

Definition 10.22 (Clauses Semantics)

For any clause $\mathcal{C}$ of the language $\mathcal{L}^*$ (definition 10.9), $\delta_\mathcal{C}$ denotes a disjunction *of all literals in $\mathcal{C}$.*
For any finite set of clauses $\mathbf{C}$ of $\mathcal{L}^*$, any structure $\mathcal{M} = [M, I]$ for $\mathcal{L}^*$, and any $\mathcal{C} \in \mathbf{C}$,

1. $\mathcal{M} \models \mathcal{C}$ *if and only if* $\mathcal{M} \models \delta_\mathcal{C}$.
2. $\mathcal{M} \models \mathbf{C}$ *if and only if* $\mathcal{M} \models \delta_\mathcal{C}$ *for all* $\mathcal{C} \in \mathbf{C}$.
3. $(A \equiv \mathbf{C})$ *if and only if* $A \equiv \sigma_\mathbf{C}$,
where $\sigma_\mathbf{C}$ is a conjunction *of all formulas $\delta_\mathcal{C}$ for all clauses $\mathcal{C} \in \mathbf{C}$.*

Obviously, all rules of $\mathbf{QRS}^*$ are **strongly sound** (definition 10.7) and theorem 10.2 holds for $\mathbf{QRS}^*$, i.e. we have the following.

Theorem 10.12 (Strong Soundness)

The proof system $\mathbf{QRS}^*$ is **strongly sound**.

We are going to prove now that any formula A of $\mathcal{L}^*$ can be transformed into in logically equivalent set of clauses.

Theorem 10.13 (Formula-Clauses Equivalency)

For any formula A of $\mathcal{L}^*$, there is an effective procedure of generating a set of clauses $\mathbf{C}_A$ of $\mathcal{L}^*$ such that

$$A \equiv \mathbf{C}_A \tag{10.44}$$

Proof
Given $A \in O\mathcal{F}$. Here is the two steps procedure. S1. We construct (finite and unique) decomposition tree $\mathbf{T}_A$. S2. We form clauses out of the leaves of the tree $\mathbf{T}_A$, i.e. for every leaf L we create a clause $\mathcal{C}_L$ determined by L (definition 10.10) and we put

$$\mathbf{C}_A = \{\mathcal{C}_L : \quad L \text{ is a leaf of } \quad \mathbf{T}_A\}.$$

Directly from the strong soundness of rules of inference of $\mathbf{QRS}^*$ (theorem 10.12) and the semantics for clauses definition 10.22 we get that

$$A \equiv \mathbf{C}_A.$$

Exercise 10.5

Find he set $\mathbf{C}_A$ *of clauses for the following formula* A.

$$(((P(b,f(x)) \Rightarrow Q(x)) \cup \neg R(z)) \cup (P(b,f(x)) \cap R(z))))$$

Solution
S1. We construct the decomposition tree for A as follows

$$\mathbf{T}_A$$

$$\begin{array}{c}
(((P(b,f(x)) \Rightarrow Q(x)) \cup \neg R(z)) \cup (P(b,f(x)) \cap R(z))) \\
\mid (\cup) \\
(((P(b,f(x)) \Rightarrow Q(x)) \cup \neg R(z)), (P(b,f(x)) \cap R(z)) \\
\mid (\cup) \\
(P(b,f(x)) \Rightarrow Q(x)), \neg R(z), (P(b,f(x)) \cap R(z)) \\
\mid (\Rightarrow) \\
\neg P(b,f(x)), Q(x), \neg R(z), (P(b,f(x)) \cap R(z)) \\
\bigwedge(\cap)
\end{array}$$

$$\neg P(b,f(x)), Q(x), \neg R(z), P(b,f(x)) \qquad \neg P(b,f(x)), Q(x), \neg R(z), R(z)$$

S2. The leaves of $\mathbf{T}_A$ are

$L_1 = \neg P(b,f(x)),\ Q(x),\ \neg R(z),\ P(b,f(x))$ and
$L_2 = \neg P(b,f(x)),\ Q(x),\ \neg R(z),\ R(z)$.

The corresponding clauses are
$\mathcal{C}_1 = \{\neg P(b,f(x)), Q(x), \neg R(z), P(b,f(x))\}$ and
$\mathcal{C}_2 = \{\neg P(b,f(x)), Q(x), \neg R(z), R(z)\}$.

The set of clauses is

$\mathbf{C}_A = \{\{\neg P(b,f(x)), Q(x), \neg R(z), P(b,f(x)\}, \{\neg P(b,f(x)), Q(x), \neg R(z), R(z)\}$.

Definition 10.23 *Clausal Form Given a formula* A *of the language* $\mathcal{L}$ *and its Skolem form* A^* *of* $\mathcal{L}^*$. *The set* $\mathbf{C}_{A^*}$ *of clauses such that*

$$A^* \ \equiv\ \mathbf{C}_{A^*}$$

s called a **clausal form** *of the formula* A *of* $\mathcal{L}$.

Exercise 10.6 *Find the clausal form of a formula* A:

$$(\exists x \forall y\ (R(x,y) \cup \neg P(x)) \Rightarrow \forall y \exists x\ \neg R(x,y)).$$

Solution
Step 1: We rename variables apart in A and get a formula A':

$$(\exists x \forall y\ (R(x,y) \cup \neg P(x)) \Rightarrow \forall z \exists w\ \neg R(z,w)).$$

Step 2: We use Equational Laws of Quantifiers (11.46), (11.47)t o pull out $\exists x$ and $\forall y$ and get a formula A'':

$$(\forall x \exists y\ ((R(x,y) \cup \neg P(x)) \Rightarrow \forall z \exists w\ \neg R(z,w)).$$

Step 3: We use Equational Laws of Quantifiers (11.46), (11.47)t o pull out $\exists x$ and $\forall y$ and get a formula A''':

$$(\forall x \exists y\ ((R(x,y) \cup \neg P(x)) \Rightarrow \forall z \exists w\ \neg R(z,w)).$$

Step 4: We use Equational Laws of Quantifiers (10.38), (10.39)t o pull out $\exists z$ and $\forall w$ from the sub formula $((R(x,y) \cup \neg P(x)) \Rightarrow \forall z \exists w\ \neg R(z,w))$ and get a formula A'''' This is the prenex normal form **PNF** of A.

$$(\forall x \exists y \forall z \exists w\ ((R(x,y) \cup \neg P(x)) \Rightarrow\ \neg R(z,w)). \tag{10.45}$$

Step 5: We perform the Skolemization Procedure (10.40) to (10.45). Observe (10.45) that the formula is of the form of the formula of exercise 10.3. We follow the exercise and eliminate $\forall x$ and get a formula A_1

$$\exists y \forall z \exists w\ ((R(x,y) \cup \neg P(x)) \Rightarrow\ \neg R(z,w)).$$

We eliminate $\exists y$ by replacing y by $h(x)$ where h is a **new** one argument functional symbol **added** to our original language $\mathcal{L}$.
We get a formula A_2

$$\forall z \exists w\ ((R(x,h(x)) \cup \neg P(x)) \Rightarrow\ \neg R(z,w)).$$

We eliminate $\forall z$ and get a formula A_3

$$\exists w\ ((R(x,h(x)) \cup \neg P(x)) \Rightarrow\ \neg R(z,w)).$$

We eliminate $\exists w$ by replacing w by $f(x,z)$, where f is a **new** two argument functional symbol **added** to our original language $\mathcal{L}$.
We get a formula A_4 that is our resulting **open** formula A^*

$$A^*:\quad ((R(x,h(x)) \cup \neg P(x)) \Rightarrow\ \neg R(z,(x,z))). \tag{10.46}$$

Step 6: We build the decomposition tree $\mathbf{T}_{A^*}$ for (10.46).

$$\mathbf{T}_{A^*}$$

$$((R(x,h(x)) \cup \neg P(x)) \Rightarrow\ \neg R(z,f(x,z)))$$

$$\mid (\Rightarrow)$$

$$\neg(R(x,h(x)) \cup \neg P(x)),\ \neg R(z,f(x,z))$$

$$\bigwedge(\neg\cup)$$

$$\neg R(x,h(x)), \neg R(z,f(x,z) \qquad\qquad \neg\neg P(x),\ \neg R(z,f(x,z))$$

$$\mid(\neg\neg)$$

$$P(x),\ \neg R(z,f(x,z))$$

Step 7: The leaves of $\mathbf{T}_{A^*}$ are

$L_1 = \neg R(x,h(x)), \neg R(z,f(x,z)$ and $L_2 = P(x),\ \neg R(z,f(x,z))$.

The corresponding clauses are
$\mathcal{C}_1 = \{\neg R(x,h(x)), \neg R(z,f(x,z)\}$ and
$\mathcal{C}_2 = \{P(x),\ \neg R(z,f(x,z))\}$.

Step 8: The **clausal form** of the formula A

$$(\exists x \forall y\ (R(x,y) \cup \neg P(x)) \Rightarrow \forall y \exists x\ \neg R(x,y))$$

is the set of clauses

$$\mathbf{C}_{A^*} = \{\ \{\neg R(x,h(x)), \neg R(z,f(x,z)\},\quad \{P(x),\ \neg R(z,f(x,z))\}\ \}.$$

10.5 Homework Problems

1. Given a predicate (first order) language (10.1), i.e. $\mathcal{L} = \mathcal{L}_{\{\cap,\cup,\Rightarrow,\neg\}}(\mathbf{P},\mathbf{F},\mathbf{C})$. Let **QRS** de a proof system (10.8). For any formulas A, B of $\mathcal{L}$, we define:

   $$\vdash_{QRS} A \equiv B \quad \text{it and only if} \quad \vdash_{QRS} (A \Rightarrow B) \ \text{ and } \ \vdash_{QRS} (B \Rightarrow A).$$

   Show that for any formulas $A(x), B(x)$ with a free variable x the following holds.
   Remider: 1. you treat $A(x), B(x)$ as atomic foprmulas, 2.you must transform formulas with restricted domain quantifiers into proper formulas of $\mathcal{L}$.

   (i) $\vdash_{QRS}\ forallx\ (A(x) \cap B(x))\ \equiv\ (\forall x A(x) \cap \forall x B(x))$

   (ii) $\vdash_{QRS}\ \exists x\ (A(x) \cup B(x))\ \equiv\ (\exists x A(x) \cup \exists x B(x))$.

   (iii) $\vdash_{QRS}\ \neg\forall_{B(x)}\ A(x) \equiv \exists_{B(x)}$.

   (iv) $\vdash_{QRS}\ \neg\exists_{B(x)}\ A(x) \equiv \forall_{B(x)}\neg A(x)$.

   (v) $\vdash_{QRS}\ \neg\forall x A(x) \equiv \exists x \neg A(x)$.

   (vi) $\vdash_{QRS}\ \neg\exists x A(x) \equiv \forall x \neg A(x)$.

   (vii) $\vdash_{QRS}\ (\forall x(B(x) \Rightarrow A(x)) \Rightarrow (\exists x\ B(x) \Rightarrow \exists x\ (B(x) \cap A(x))))$

2. Show that for any formulas $A(x), B$ B where B does not contain any free occurrence of x the following holds.

   (i) $\vdash_{QRS} \forall x(A(x) \cap B) \equiv (\forall x A(x) \cap B)$.

   (ii) $\vdash_{QRS} \forall x(A(x) \cup B) \equiv (\forall x A(x) \cup B)$.

   (iii) $\vdash_{QRS} \exists x(A(x) \Rightarrow B) \equiv (\forall x A(x)$.

   (iv) $\vdash_{QRS} \exists x(A(x) \Rightarrow B) \equiv (\forall x A(x)$.

3. Prove that following formulas are not provable in **QRS**.
   Remider: you must transform formulas with restricted domain quantifiers into proper formulas of $\mathcal{L}$.

   (i) $\exists_{C(x)}(A(x) \cup B) \not\equiv (\exists_{C(x)} A(x) \cup B)$.

   (ii) $\forall_{C(x)}(A(x) \cap B) \not\equiv (\forall_{C(x)} A(x) \cap B)$.

   (iii) $\exists_{C(x)}(A(x) \Rightarrow B) \not\equiv (\forall_{C(x)} A(x) \Rightarrow B)$.

   (iv) $\exists_{C(x)}(B \Rightarrow A(x)) \not\equiv (B \Rightarrow \exists x A(x))$.

4. Prove that following formulas are not provable in **QRS**.

   (i) $(\exists x\ \neg A(x) \Rightarrow \forall x\ A(x))$

   (ii) $(\forall x \exists y\ A(x, y) \Rightarrow \exists x \forall y\ A(x, y))$.

   (iii) $(\exists x \exists y\ A(x, y) \Rightarrow \exists y\ A(y, y))$.

   (iv) $(\forall x \exists y\ A(x, y) \Rightarrow \exists y\ A(y, y))$.

   (v) $(\forall x\ (A(x) \Rightarrow\ B(x)) \Rightarrow (\forall x\ A(x) \Rightarrow \exists x\ B(x)))$.

5. Prove that following formulas are not provable in **QRS**.

   (i) $A_1 :\ \forall x \neg \exists y(P(x, g(y, y)) \cup P(x, g(g(y, y), d)))$.

   (ii) $A_2 :\ (\neg \forall y P(f(x, y), c) \Rightarrow (P(x, c) \cup P(y, c)))$

   (iii) $A_3 :\ \forall x(P(x) \Rightarrow \exists y Q(x, y))$.

   (iv)$A_4 :\ \forall x \neg \exists y(P(x) \cap \neg Q(x, y))$.

6. Find counter-models determined by the decomposition trees $\mathbf{T}_{A_i}$ for the following formulas A_i, $i = 1, 2, 3, 4$.

   (i) $A_1 :\ \forall x \neg \exists y(Q(x, g(y)) \cup R(x, f(x, y), c)))$.

   (ii) $A_2 :\ (\neg \forall y R(f(x, y), c) \Rightarrow (Q(x, c) \cup Q(y, c)))$

   (iii) $A_3 :\ \forall x(P(x) \Rightarrow \exists y Q(x, y))$.

   (iv)$A_4 :\ \forall x \neg \exists y(P(x) \cap \neg Q(f(x, y)))$.

7. Find prenex normal form **PNF** of the following formulas.
   Reminder: We assume that the formula A in **PNF** is always **closed**. If it is not closed we form its **closure** (definition 10.16) instead.

   (i) $(\forall x(P(x) \Rightarrow \neg \forall y P(y)) \Rightarrow (\exists x\ R(x, y) \Rightarrow \exists y\ (R(x, y) \cap P(y))))$.

   (ii) $((\forall x Q(x) \Rightarrow (\exists x R(x) \cup \neg \forall x Q(x))) \Rightarrow (\neg \exists x Q(x) \cap R(x)))$.

(iii) $(\forall x\, R(f(x,y),c) \Rightarrow (\exists x R(f(x,y),c)) \cap \neg R(f(x,y),c)) \Rightarrow (\neg\forall x\, R(f(x,y),c) \Rightarrow \exists x\ R(f(x,y),c)))$.

(iv) $((\exists_{R(y)} P(y) \Rightarrow Q(x)) \Rightarrow (P(y) \Rightarrow \exists x Q(x)))$

8. Find a **Skolem form** of the following formulas (the formula $B(y_1, y_2, y_3, y_4, y_4)$ is quantifiers free).

   (i) $\forall y_1 \forall y_2 \forall y_3 \exists y_4\ B(y_1, y_2, y_3, y_4, y_4)$.

   (ii) $\exists y_1 \exists y_2 \forall y_3 \exists y_4\ B(y_1, y_2, y_3, y_4, y_4)$.

   (iii) $\exists y_1 \forall y_2 \exists y_3 \exists y_4\ B(y_1, y_2, y_3, y_4, y_4)$.

   (iv) $\forall y_1 \forall y_2 \exists y_3 \exists y_4\ B(y_1, y_2, y_3, y_4, y_4)$.

9. Find the clausal form of the following formulas.

   (i) $(\forall x(P(x) \Rightarrow \neg\forall y P(y)) \Rightarrow (\exists x\ R(x,y) \Rightarrow \exists y\ (R(x,y) \cap P(y))))$.

   (ii) $((\forall x Q(x) \Rightarrow (\exists x R(x) \cup \neg\forall x Q(x))) \Rightarrow (\neg\exists x Q(x) \cap R(x)))$.

   (iii) $(\forall x\, R(f(x,y),c) \Rightarrow (\exists x R(f(x,y),c)) \cap \neg R(f(x,y),c)) \Rightarrow (\neg\forall x\, R(f(x,y),c) \Rightarrow \exists x\ R(f(x,y),c)))$.

   (iv) $((\exists_{R(y)} P(y) \Rightarrow Q(x)) \Rightarrow (P(y) \Rightarrow \exists x Q(x)))$

10. Find the set of clauses logically equivalent to clausal form of the following formulas.

    (i) $(((a \Rightarrow b) \cap \neg c) \cup (a \Rightarrow c))$.

    (ii) $((a \cup b) \Rightarrow \neg a) \cup (\neg a \Rightarrow \neg c))$.

    (iii) $(\neg(\neg a => (a \cap \neg b)) => (\neg a \cap (\neg a \cup \neg b)))$.

    (iv) $(\neg(a \cap b) \Rightarrow (\neg a \cup \neg b))$.

    (v) $(((\neg a \Rightarrow (b \cap c)) \cap \neg(c \cup a)) \cup (a \Rightarrow c))$

Chapter 11

Formal Theories and Gödel Theorems

Formal theories play crucial role in mathematics and were historically defined for classical predicate (first order logic) and consequently for other first and higher order logics, classical and non-classical.

The idea of formalism in mathematics, which resulted in the concept of formal theories, or formalized theories, as they are also called. Their concept was developed in connection with the *Hilbert Program.* One of the main objects of the program was to construct a formal theory that would cover the whole mathematics and to prove its *consistency* by employing the simplest of logical means. This part of the program was called the *Consistency Program*, where a formal theory is said to be *consistent* if no formal proof can be carried in that theory for a formula A and at the same time for its negation $\neg A$.

In 1930, while still in his twenties Kurt Gödel made a historic announcement: Hilbert Consistency Program could not be carried out. He justified his claim by proving his Inconsistency Theorem, called also Second Incompleteness Theorem. Roughly speaking the theorem states that a proof of the consistency of every formal theory that contains arithmetic of natural numbers can be carried out only in mathematical theory which is more comprehensive than the one whose consistency is to be proved. In particular, a proof of the consistency of formal (elementary, first order) arithmetic can be carried out only in mathematical theory which contains the whole arithmetic and also other theorems that do not belong to arithmetic. It applies to a formal theory that would cover the whole mathematics because it would obviously contain the arithmetic on natural numbers. Hence the *Hilbert Consistency Program* fails.

Gödel's result concerning the proofs of the consistency of formal mathematical theories has had a decisive impact on research in properties of formal theories.

Instead of looking for direct proofs of inconsistency of mathematical theories, mathematicians concentrated largely on relative proofs that demonstrate that a theory under consideration is consistent if a certain other theory, for example a formal theory of natural numbers, is consistent. All those relative proofs are rooted in a deep conviction that even though it cannot be proved that the theory of natural numbers is free of inconsistencies, it is consistent. This conviction is confirmed by centuries of development of mathematics and experiences of mathematicians.

A formal theory is called *complete* if for every sentence (formula without free variables) of the language of that theory there is a formal proof of it or of its negation. A formal theory which does not have this property is called *incomplete*. Hence a formal theory is *incomplete* if there is a sentence A of the language of that theory, such that neither A nor $\neg A$ are provable in it. Such sentences are called *undecidable* in the theory in question or *independent* of the theory.

It might seem that one should be able to formalize a theory such as the formal theory of natural numbers in a way to make it complete, i.e. free of undecidable (independent) sentences. But it is not the case in view of Gödel's Incompleteness Theorem. It states that every consistent formal theory which contains the arithmetic of natural numbers is incomplete. The Inconsistency Theorem follows from it. This is why the Incompleteness and Inconsistency Theorems are now called Gödel First Incompleteness Theorem (theorems 11.3, 11.6) and Gödel Second Incompleteness (theorems 11.4, 11.7), respectively.

The third part of the *Hilbert Program* posed and was concerned with the problem of *decidability* of formal mathematical theories. A formal theory is called **decidable** if there is a method of determining, in a finite number of steps, whether any given formula in that theory is its theorem or not. If a theory is decidable and if the decision algorithm is known, then the study of problems expressible in the language of the theory reduces to a purely mechanical procedure. In undecidable theories there is no mechanical procedure. Most of mathematical theories are undecidable. Gödel proved in 1931 that the arithmetic of of natural numbers is undecidable.

We discuss the *Hilbert Program* and Gödel's Theorems in more details in sections 11.3.1 and 11.3.2, respectively.

11.1 Formal Theories: Definition and Examples

We define here a notion of a formal theory based on a predicate (first order) language. Formal theories are also routinely called *first order theories, elementary theories, formal axiomatic theories*, or just *theories*, when it is clear from the context that they are formal theories. We will often use the term *theory* for simplicity.

Remark 11.1

We consider here only classical formal theories based on a complete classical Hilbert style proof system. We also assume that its language contains the full set $\{\neg, \cap, \cup, \Rightarrow\}$ *of propositional connectives.*

Given a **classical** Hilbert style proof system

$$H = (\mathcal{L},\ \mathcal{F},\ LA,\ \mathcal{R}) \tag{11.1}$$

with a predicate (first order) language

$$\mathcal{L} = \mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow\}}(\mathbf{P}, \mathbf{F}, \mathbf{C}),$$

where the sets **P, F, C** are infinitely enumerable.

A formal theory *based on* H is a proof system obtained from H by adding a new special set SA of axioms to it, called the set of *specific axioms.* The specific axioms are characteristic descriptions of the universe of the formal theory.This is why we call them specific axioms and denote by SA. The specific axioms are to be true only in a certain structure as opposed to logical axioms LA that are true in all structures.

Definition 11.1 (Specific Axioms)

Let SA *be a certain set of formulas of* $\mathcal{L}$ *of* $H = (\mathcal{L},\ \mathcal{F},\ LA,\ \mathcal{R})$*, such that*

$$SA \subseteq \mathcal{F} \quad \textit{and} \quad LA \cap SA = \emptyset. \tag{11.2}$$

We call the set SA *a set of* **specific axioms** *of* H.

Definition 11.2 (Language $\mathcal{L}_{SA}$)

Given a proof system $H = (\mathcal{L},\ \mathcal{F},\ LA,\ \mathcal{R})$ *and a non-empty set* SA *(11.2) of specific axioms. We define a language*

$$\mathcal{L}_{SA} \subseteq \mathcal{L} \tag{11.3}$$

by restricting the sets **P, F, C** *of predicate, functional, and constant symbols of* $\mathcal{L}$ *to predicate, functional, and constant symbols appearing in the set* SA *of specific axioms. Both languages* $\mathcal{L}_{SA}$ *and* $\mathcal{L}$ *share the same set of propositional connectives.*

Obviously, if $SA = \emptyset$, then $\mathcal{L}_{SA} = \mathcal{L}$.

Now we are ready to define a formal (first order) classical theory as follows.

Definition 11.3 (Formal Theory)

A proof system

$$T = (\mathcal{L},\ \mathcal{F},\ LA,\ SA,\ \mathcal{R}), \tag{11.4}$$

is called a **formal theory** *with the set SA of* **specific axioms**.

The language $\mathcal{L}_{SA}$ defined by (11.3) is called the **language of the theory** *T. The theory T (11.4) is based on a complete classical proof system*

$$H = (\mathcal{L},\ \mathcal{F},\ LA,\ \mathcal{R}).$$

Definition 11.4

Given a theory $T = (\mathcal{L},\ \mathcal{F},\ LA,\ SA,\ \mathcal{R})$. We denote by $\mathcal{F}_{SA}$ the set of formulas of the language $\mathcal{L}_{SA}$ of T. We denote by $\mathbf{T}$ *the set all provable formulas in the theory T, i.e.*

$$\mathbf{T} = \{B \in \mathcal{F}_{SA} :\quad SA \vdash\ B.\} \tag{11.5}$$

We also write $\vdash_T B$ to denote that $B \in \mathbf{T}$.

Definition 11.5 (Theory with Equality)

A theory T is called a **theory with equality** *if and only if its language $\mathcal{L}_{SA}$ has as one of its predicates, a two argument predicate P which we denote by $=$, and all Equality Axioms (11.6) are* **provable** *in T.*

Equality Axioms (11.6)

For any any free variable or constant of $\mathcal{L}_{SA}$, $R \in \mathbf{P}$, and $t \in \mathbf{T}$, where R is an arbitrary n-ary relation symbol of $\mathcal{L}_{SA}$ and $t \in \mathbf{T}$ is an arbitrary n-ary term of $\mathcal{L}_{SA}$ the following properties hold.

E1 $u = u$,

E2 $(u = w \Rightarrow w = u)$,

E3 $((u_1 = u_2 \cap u_2 = u_3) \Rightarrow u_1 = u_3)$,

E4 $((u_1 = w_1 \cap ... \cap u_n = w_n) \Rightarrow (R(u_1, ..., u_n) \Rightarrow R(w_1, ..., w_n)))$,

E5 $((u_1 = w_1 \cap ... \cap u_n = w_n) \Rightarrow (t(u_1, ..., u_n) \Rightarrow t(w_1, ..., w_n)))$.

Directly from above definitions we have the following.

Fact 11.1 *The Hilbert style proof system* **H** *defined in chapter 9 is a theory with equality with the set of specific axioms $SA = \emptyset$.*

Some Examples of Formal Theories

Formal theories are abstract models of real mathematical theories we develop using laws of logic. Hence the theories we present here are based on a **complete**

proof system H for classical predicate logic with a language

$$\mathcal{L} = (\mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow\}}(\mathbf{P}, \mathbf{F}, \mathbf{C}).$$

The first order formal theories are also called *Elementary Theories.*

$T1$. Theory **of equality**

Language

$$\mathcal{L}_{T1} = \mathcal{L}_{\{\neg,\Rightarrow,\cup,\cap\}}(\mathbf{P} = \{P\},\ \mathbf{F} = \emptyset,\ \mathbf{C} = \emptyset),$$

where # P = 2, i.e. P is a two argument predicate. The intended interpretation of P is equality, so we use the equality symbol $=$ instead of P. We write $x = y$ instead $= (x, y)$. We write the language of $T1$ as follows.

$$\mathcal{L}_{T1} = \mathcal{L}_{\{\neg,\Rightarrow,\cup,\cap\}}(\{=\},\ \emptyset,\ \emptyset).$$

Specific Axioms

e1 $\quad x = x,$

e2 $\quad (x = y \Rightarrow y = x),$

e3 $\quad (x = y \Rightarrow (y = z \Rightarrow x = z)),$

for any $x, y, z \in VAR$,

Exercise 11.1

Show that the theory $T1$ of equality is a theory with equality of definition 11.5.

Solution
The first to axioms $e1, e2$ are particular cases of $E1, E2$. We have only to show that the axiom $E3$ is provable in $T1$, i.e. that the formula

$$((x = y \cap y = z) \Rightarrow x = z) \in \mathbf{T1}, \tag{11.7}$$

where, by (11.5) $\mathbf{T1} = \{A \in \mathcal{F}_{\{e1,e2,e3\}} :\ \{e1, e2, e3\} \vdash A\}$.

Observe that by definition, $T1$ is based on a complete Hilbert style proof system. A formula

$$(((A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \cap B) \Rightarrow C))$$

is a predicate tautology, hence is **provable** in $T1$ for any $A, B, C \in \mathcal{F}_{\{e1,e2,e3\}}$. In particular its instance for $A : x = y,\ B : y = z,\ C : x = z$ is also provable in $T1$ what means that

$$(((x = y \Rightarrow (y = z \Rightarrow x = z)) \Rightarrow ((x = y \cap y = z) \Rightarrow x = z)) \in \mathbf{T1}. \tag{11.8}$$

Applying Modus Ponens (MP) to axiom e3 and (11.8), we get that

$$((x = y \cap y = z) \Rightarrow x = z) \in \mathbf{T1}.$$

It proves that (11.7) holds and **ends** the proof.

Observation 11.1 *We have chosen to write the specific axioms as open formulas. Sometimes it is more convenient to write them as closed formulas (sentences). In this case new axioms will be closures of axioms that were open formulas.*

Taking closures of axioms of $T1$ we obtain the following new formalization.

$T2$. Theory **of equality** (2)

We adopt a closure of the axioms $e1, e2, e3$, i.e. the following new set of axioms.

Specific Axioms

(e1) $\forall x(x = x)$,

(e2) $\forall x \forall y(x = y \Rightarrow y = x)$,

(e3) $\forall x \forall y \forall z(x = y \Rightarrow (y = z \Rightarrow x = z))$.

$T3$. Theory of **Partial Order**

Partial order relation is also called **order** relation.

Language

$$\mathcal{L}_{T1} = \mathcal{L}_{\{\neg, \Rightarrow, \cup, \cap\}}(\mathbf{P} = \{P, Q\},\ \mathbf{F} = \emptyset,\ \mathbf{C} = \emptyset),$$

where # P = 2, i.e. P is a two argument predicate. The intended interpretation of P is equality, so we use the equality symbol $=$ instead of P. We write $x = y$ instead $= (x, y)$.
Q is a two argument predicate. The intended interpretation of Q is partial order, called also order relation, so we use the order symbol $\leq$ instead of Q. We write $x \leq y$ instead $\leq (x, y)$.
The language is

$$\mathcal{L}_{T3} = \mathcal{L}_{\{\neg, \Rightarrow, \cup, \cap\}}(\{=,\ \leq\},\ \emptyset,\ \emptyset).$$

Specific Axioms

There are two groups of specific axioms: equality and order axioms. We adopt equality axioms (11.6) to the language $\mathcal{L}_{T3}$ as follows.

Equality Axioms

For any $x, y, z, x_1, x_2, y_1, y_2, \in VAR$,

e1 $x = x$,

e2 $(x = y \Rightarrow y = x)$,

e3 $((x = y \cap y = z) \Rightarrow x = z)$,

e4 $((x_1 = y_1 \cap x_2 = y_2) \Rightarrow (x_1 \leq x_2 \Rightarrow y_1 \leq y_2))$.

Partial Order Axioms

o1 $x \leq x$, *(reflexivity)*

o2 $((x \leq y \cap y \leq x) \Rightarrow x = y)$, *(antisymmetry)*

o3 $((x \leq y \cap y \leq z) \Rightarrow x \leq z)$, *(trasitivity)*

where $x, y, z \in VAR$.

The model of $T3$ is called a *partially ordered structure.*

$T4$. Theory of **Partial Order** (2)

Here is another formalization for partial order.

Language

$$\mathcal{L}_{T4} = \mathcal{L}_{\{\neg, \Rightarrow, \cup, \cap\}}(\mathbf{P} = \{P\},\ \mathbf{F} = \emptyset,\ \mathbf{C} = \emptyset),$$

where # P = 2, i.e. P is a two argument predicate. The intended interpretation of $P(x, y)$ is $x < y$, so we use the "less" symbol $<$ instead of P. We write $x < y$ instead $< (x, y)$. We also write $x \not< y$ for $\neg(x < y)$, i.e. $\neg < (x, y)$.

The language of $T4$ is

$$\mathcal{L}_{T4} = \mathcal{L}_{\{\neg, \Rightarrow, \cup, \cap\}}(\{<\},\ \emptyset,\ \emptyset).$$

Specific Axioms

For any $x, y, z \in VAR$,

p1 $x \not< x$, *(irreflexivity)*

p2 $((x \leq y \cap y \leq z) \Rightarrow x \leq z)$. *(trasitivity)*

$T5$. **Theory of Linear Order**

Linear order relation is also called **total order** relation.

Language

$$\mathcal{L}_{T5} = \mathcal{L}_{\{\neg, \Rightarrow, \cup, \cap\}}(\{=,\ \leq\},\ \emptyset,\ \emptyset).$$

Specific Axioms

We adopt all axioms of theory $T3$ of partial order and add the following additional axiom.

o4 $(x \leq y) \cup (y \leq x)$.

This axiom says that in linearly ordered sets each two elements are comparable.

$T6$. Theory of **Dense Order**

Language

$$\mathcal{L}_{T6} = \mathcal{L}_{\{\neg, \Rightarrow, \cup, \cap\}}(\{=,\ \leq\},\ \emptyset,\ \emptyset).$$

Specific Axioms

We adopt all axioms of theory $T5$ of linear order and add the following additional axiom. We write $x \neq y$ for $\neg(x = y)$, i.e. for the formula $\neg = (x, y)$.

o5 $((x \leq y \cap x \neq y) \Rightarrow \exists z((x \leq z \cap x \neq z) \cap (z \leq y \cap z \neq y)))$.

This axiom says that in linearly ordered sets between any two different elements there is a third element between them, respective to the order.

$T7$. **Lattice Theory**

Language

$$\mathcal{L}_{T7} = \mathcal{L}_{\{\neg, \Rightarrow, \cup, \cap\}}(\mathbf{P} = \{P, Q\},\ \mathbf{F} = \{f, g\},\ \mathbf{C} = \emptyset),$$

where # P = 2, i.e. P is a two argument predicate. The intended interpretation of P is equality, so we use the equality symbol $=$ instead of P. We write $x = y$ instead $= (x, y)$.
Q is a two argument predicate. The intended interpretation of Q is partial order, called also order relation, so we use the order symbol $\leq$ instead of Q. We write $x \leq y$ instead $\leq (x, y)$.
# f = # g 2, i.e. f, g are a two argument functional symbols. The intended interpretation of f, g is the lattice intersection $\wedge$ and union $\vee$, respectively.
We write $(x \wedge y)$ for $\wedge(x, y)$ and $(x \vee y)$ for $\vee(x, y)$.

Observe that $(x \cap y)$, $(x \cup y)$ are **atomic formulas** of $\mathcal{L}_{T7}$ and $(x \wedge y)$ and $(x \vee y)$ are **terms** of $\mathcal{L}_{T7}$.

We write the language as

$$\mathcal{L}_{T7} = \mathcal{L}_{\{\neg, \Rightarrow, \cup, \cap\}}(\{=, \leq\},\ \{\wedge, \vee\},\ \emptyset).$$

Specific Axioms

There are three groups of specific axioms: equality axioms, order axioms, and lattice axioms. We adopt equality axioms (11.6) to the language $\mathcal{L}_{T7}$ as follows.

Equality Axioms

For any $x, y, z, x_1, x_2, y_1, y_2, \in VAR$,

e1 $x = x$,

e2 $(x = y \Rightarrow y = x)$,

e3 $((x = y \cap y = z) \Rightarrow x = z)$,

e4 $((x_1 = y_1 \cap x_2 = y_2) \Rightarrow (x_1 \leq x_2 \ \Rightarrow \ y_1 \leq y_2))$,

e5 $((x_1 = y_1 \cap x_2 = y_2) \Rightarrow (x_1 \wedge x_2 \ \Rightarrow \ y_1 \wedge y_2))$,

e6 $((x_1 = y_1 \cap x_2 = y_2) \Rightarrow (x_1 \vee x_2 \ \Rightarrow \ y_1 \vee y_2))$.

Remark 11.2

We write $\wedge$ for the lattice functional symbol of intersection in order to better distinguish it from the conjunction symbol $\cap$ in the formula.
The same applies to the next axiom e7 that involves lattice functional symbol $\vee$ for the union and disjunction symbol $\cup$ in the formula.

Partial Order Axioms

For any $x, y, z \in VAR$,

o1 $x \leq x$, *(reflexivity)*

o2 $((x \leq y \cap y \leq x) \Rightarrow x = y)$, *(antisymmetry)*

o3 $((x \leq y \cap y \leq z) \Rightarrow x \leq z)$. *(trasitivity)*

Lattice Axioms

For any $x, y, z \in VAR$,

b1 $(x \wedge y) = (y \wedge x)$, $(x \wedge y) = (x \wedge y)$,

b2 $(x \wedge (y \wedge z)) = ((x \wedge y) \wedge z)$, $(x \vee (y \vee z)) = ((x \vee y) \vee z)$,

b3 $(((x \wedge y) \vee y) = y)$, $((x \wedge (x \vee y)) = x)$.

$T8$. Theory of **Distributive Lattices**

Language

$$\mathcal{L}_{T8} = \mathcal{L}_{\{\neg, \Rightarrow, \cup, \cap\}}(\{=, \leq\},\ \{\wedge, \vee\},\ \emptyset).$$

Specific Axioms

We adopt all axioms of theory $T7$ of lattice theory and add the following additional axiom.

b4 $(x \wedge (y \vee z)) = ((x \wedge y) \vee (x \wedge z))$.

$T9$. Theory of **Boolean Algebras**

Language

$$\mathcal{L}_{T9} = \mathcal{L}_{\{\neg, \Rightarrow, \cup, \cap\}}(\{=, \leq\},\ \{\wedge, \vee, -\},\ \emptyset),$$

where $-$ is one argument function symbol representing algebra complement.

Specific Axioms

We adopt all axioms of theory $T8$ of distributive lattices theory and add the following additional axioms that characterize the algebra complement $-$.

b5 $(((x \wedge -x) \vee y) = y), \qquad (((x \vee -x) \wedge y) = y).$

$T10$. Theory of **Groups**

Language

$$\mathcal{L}_{T10} = \mathcal{L}_{\{\neg, \Rightarrow, \cup, \cap\}}(\mathbf{P} = \{P\},\ \mathbf{F} = \{f, g\},\ \mathbf{C} = \{c\}),$$

where # P = 2, i.e. P is a two argument predicate. The intended interpretation of P is equality, so we use the equality symbol $=$ instead of P. We write $x = y$ instead $= (x, y)$.
f is a two argument functional symbol. The intended interpretation of f is group operation $\circ$. We write $(x \circ y)$ for the formula $\circ(x, y)$.
g is a one argument functional symbol. $g(x)$ represent a group inverse element to a given x and we denote it by x^{-1}. We hence use a symbol $^{-1}$ for g.
c is a constant symbol representing unit element in the group and we use a symbol e to denote it

$$\mathcal{L}_{T10} = \mathcal{L}_{\{\neg, \Rightarrow, \cup, \cap\}}(\{=\},\ \{\circ,\ ^{-1}\},\ \{e\}).$$

Specific Axioms

There are two groups of specific axioms: equality axioms and group axioms. We adopt equality axioms (11.6) to the language $\mathcal{L}_{T10}$ as follows.

Equality Axioms

For any $x, y, z, x_1, x_2, y_1, y_2, \in VAR$,

e1 $x = x$,

e2 $(x = y \Rightarrow y = x)$,

e3 $((x = y \cap y = z) \Rightarrow x = z)$,

e4 $(x = y \Rightarrow x^{-1} = y^{-1})$,

e5 $((x_1 = y_1 \cap x_2 = y_2) \Rightarrow (x_1 \circ x_2 \Rightarrow y_1 \circ y_2))$.

Group Axioms

g1 $(x \circ (y \circ z)) = ((x \circ y) \circ z)$,

g2 $(x \circ e) = x$,

g3 $(x \circ x^{-1}) = e.$

$T11$. Theory of **Abelian Groups**

Language is the same as $\mathcal{L}_{T11}$, i.e.

$$\mathcal{L}_{T11} = \mathcal{L}_{\{\neg, \Rightarrow, \cup, \cap\}}(\{=\},\ \{\circ,\ ^{-1}\},\ \{e\})'$$

Specific Axioms

We adopt all axioms of theory $T11$ of groups and add the following additional axiom.

g4 $(x \circ y) = (y \circ x).$

$T12$. **Theory of Groups** (2)

Here is another formalization for group theory.

Language

$$\mathcal{L}_{T12} = \mathcal{L}_{\{\neg, \Rightarrow, \cup, \cap\}}(\mathbf{P} = \{P\},\ \mathbf{F} = \{f\},\ \mathbf{C} = \{c\}),$$

where # P = 2, i.e. P is a two argument predicate. The intended interpretation of P is equality, so we use the equality symbol $=$ instead of P. We write $x = y$ instead $= (x, y)$.
f is a two argument functional symbols. The intended interpretation of f is group operation $\circ$. We write $(x \circ y)$ for the formula $\circ(x, y)$.
c is a constant symbol representing unit element in the group and we use a unit symbol e to denote it.

$$\mathcal{L}_{T12} = \mathcal{L}_{\{\neg, \Rightarrow, \cup, \cap\}}(\{=\},\ \{\circ\},\ \{e\}).$$

Specific Axioms

For any $x, y, z, x_1, x_2, y_1, y_2, \in VAR$,

a1 $(x \circ (y \circ z)) = ((x \circ y) \circ z),$

a2 $(x \circ e) = x,$

a3 $\forall x \exists y ((x \circ y) = e),$

a4 $x = x,$

a5 $(x = y \Rightarrow y = x),$

a6 $(x = y \Rightarrow (y = z \Rightarrow x = z)),$

a7 $(x = y \Rightarrow (x \circ z = y \cap z \circ x = z \circ y)).$

$T13$. Theory of **Abelian Groups** (2)

We adopt the language and all axioms of theory $T12$ of groups and add the following additional axiom.

a8 $(x \circ y) = (y \circ x)$.

Observe that what we formally prove in the formal axiomatic theories presented here represents only fragments of corresponding axiomatic theories developed in mathematics. For example Group Theory, Boolean Algebras Theory are fields in mathematics and many theorems developed there, like the Representation Theorem for Boolean Algebras, and many, many others in other domains can not be expressed in the languages of respective formal theories. This is a reason why we also call them elementary theories. For example, we say *elementary group theory* to distinguish it from the Group Theory as a lager field of mathematics.

11.2 PA: Formal Theory of Natural Numbers

Next to geometry, the theory of natural numbers is the most intuitive and intuitively known of all branches of mathematics. This is why the first attempts to formalize mathematics begin with with arithmetic of natural numbers. The first attempts of axiomatic formalization of arithmetic of natural numbers was given by Dedekind in 1879 and by Peano in 1889. The Peano formalization became known as **Peano Postulates** (axioms) and can be written as follows.

p1 0 is a natural number.

p2 If $n$is a natural number, there is another number which we denote by n'.

We call n' a *successor* of n. The intuitive meaning of n' is $n+1$.

p3 $0 \neq n'$, for any natural number n.

p4 If $n' = m'$, then $n = m$, for any natural numbers n, m.

p5 If W is is a property that may or may not hold for natural numbers, and if (i) 0 has the property W and (ii) whenever a natural number n has the property W, then n' has the property W,
then all natural numbers have the property W.

p5 is called *Principle of Induction.*

These axioms, together with a certain amount of set theory, are sufficient to develop not only theory of natural numbers, but also theory of rational and even real numbers. But they can't act as a fully formal theory as they include intuitive notions like "property" and "has a property".

A formal theory of natural numbers based on Peano Postulates is referred in literature as *Peano Arithmetic*, or simply PA. We present here formalization by Mendelson (1973) that is included and worked out in smallest details in his book *Intoduction to Mathematical Logic* (1987). We refer the reader to this excellent book for details and further reading.

We assume, as we did in the previous section **??** that $T14$ and other theories considered here are based on a **complete** Hilbert style proof system

$$H = (\mathcal{L},\ \mathcal{F},\ LA,\ \mathcal{R}) \tag{11.9}$$

for classical predicate logic with a language

$$\mathcal{L} = (\mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow\}}(\mathbf{P},\mathbf{F},\mathbf{C}),$$

We additionally assume now that the system H has as one of ts inference rules a a generalization rule

$$(G)\ \ \frac{A(x)}{\forall x A(x)}. \tag{11.10}$$

We do so to facilitate use the Mendelson's book as a supplementary reading to the material included here and for additional reading for material not covered here.

Remark 11.3

The Deduction Theorem as proved in chapter 9 holds for the proof system system H defined by (11.9).

$$T14. \quad \textbf{Peano Arithmetic}\ PA \tag{11.11}$$

Language

$$\mathcal{L}_{PA} = \mathcal{L}(\mathbf{P} = \{P\},\ \mathbf{F} = \{f, g, h\},\ \mathbf{C} = \{c\}),$$

where # P = 2, i.e. P is a two argument predicate. The intended interpretation of P is equality, so we use the equality symbol $=$ instead of P. We write $x = y$ instead $= (x, y)$. We write $x \neq y$ for $\neg(x = y)$.
f is a one argument functional symbol. $f(x)$ represent the successor of a given x and we denote it by x'. We hence use a symbol $'$ for f.
g, h is are two argument functional symbols. The intended interpretation of f is addition and the intended interpretation of g is multiplication. We write $x + y$ for $f(x, y)$ and $x \cdot y$ for $g(x, y)$.

c is a constant symbol representing zero and we use a symbol 0 to denote c.

$$\mathcal{L}_{PA} = \mathcal{L}_{\{\neg,\Rightarrow,\cup,\cap\}}(\{=\},\ \{\ ',\ +,\ \cdot\},\ \{0\}).$$

Specific Axioms

P1 $(x = y \Rightarrow (x = z \Rightarrow y = z))$,

P2 $(x = y \Rightarrow x' = y')$,

P3 $0 \neq x'$,

P4 $(x' = y' \Rightarrow x = y)$,

P5 $x + 0 = x$,

P6 $x + y' = (x + y)'$

P7 $x \cdot 0 = 0$,

P8 $x \cdot y' = (x \cdot y) + x$,

P9 $(A(0) \Rightarrow (\forall x(A(x) \Rightarrow A(x') \Rightarrow \forall x A(x))))$,

for all formulas $A(x)$ of $\mathcal{L}_{PA}$ and all $x, y, z \in VAR$.

The axiom P9 is called **Principle of Mathematical Induction**. It does not fully corresponds to Peano Postulate p5 which refers intuitively to all possible properties on natural numbers (uncountably many). The P7 axiom applies only to properties defined by infinitely countably formulas of $A(x)$ of $\mathcal{L}_{PA}$.
Axioms P3, P4 correspond to Peano Postulates p3, p4. The Postulates p1, p2 are fulfilled by presence of 0 and successor function. Axioms P1, P2 deal with some needed properties of equality that were probably assumed as intuitively obvious by Peano and Dedekind. Axioms P5 - P8 are the recursion equations for addition and multiplication. They are not stated in the Peano Postulates as Dedekind and Peano allowed the use of intuitive set theory within which the existence of addition and multiplication and their properties P5-P8 can be proved (Mendelson, 1973).

Observe that while axioms P1 - P6 of theory of Peano Arithmetic PA are particular formulas of $\mathcal{L}_{PA}$, the axiom P9 as an axiom schema providing an infinite number of axioms. This means that the set of axioms P1 - P9 does not provide a *finite axiomatization* for Peano Arithmetic. But any formalization of Peano Postulates must include formalization of the Principle of Induction p5 and hence must contain some form of induction axiom P9. It was proved formally in 1952 by Czeslaw Ryll-Nardzewski and Rabin in 1961.

Theorem 11.1 (Ryll-Nardzewski)

Peano Arithmetic is **is not** *finitely axiomatizable; that is there is no theory K having inly a finite number of proper axioms, whose theprems are the same as those of PA.*

By definition 11.4, given a theory T, we denote by $\mathbf{T}$ the set all provable formulas in T. In particular, $\mathbf{PA}$ denotes the set of all formulas provable in Peano Arithmetic PA.

Theory PA is one of many formalizations of Peano Arithmetic. They all represent what we call Peano Arithmetic if they have the same set of theorems. We adopt hence the following definition.

Definition 11.6 *Any theory T such that $\mathbf{T} = \mathbf{PA}$ for PA defined by (11.11) is called a* Peano arithmetic*.*

Taking closure of axioms P1 - P8 of $T14$ we obtain new theory $T15$. The axiom P9 is a sentence (closed formula) already.

T15. Theory CPA

$$\mathcal{L}_{T15} = \mathcal{L}_{T14} = \mathcal{L}_{\{\neg,\Rightarrow,\cup,\cap\}}(\{=\},\ \{',\ +,\ \cdot\},\ \{0\}).$$

We denote the specific axioms of $T15$ by CPA to express that its specific axioms are closures of specific axioms of PA.

Specific Axioms

C1 $\forall x\forall y\forall z(x = y \Rightarrow (x = z \Rightarrow y = z))$,

C2 $\forall x\forall y(x = y \Rightarrow x' = y')$,

C3 $\forall x(0 \neq x')$,

C4 $\forall x\forall y(x' = y' \Rightarrow x = y)$,

C5 $\forall x(x + 0 = x)$,

C6 $\forall x\forall y(x + y' = (x + y)')$

C7 $\forall x(x \cdot 0 = 0)$,

C8 $\forall x\forall y(x \cdot y' = (x \cdot y) + x)$,

C9 $(A(0) \Rightarrow (\forall x(A(x) \Rightarrow A(x')) \Rightarrow \forall xA(x)))$,

for all formulas $A(x)$ of $\mathcal{L}_{PA}$ and all $x, y, z \in VAR$.

Here is a very simple exercise.

Fact 11.2

Theory CPA is a Peano Arithmetic.

Proof
By definition 11.6 we have to show that $\mathbf{PA} = \mathbf{CPA}$. Observe that $\mathcal{L}_{CPA} = \mathcal{L}_{PA}$, so we have to show that for any formula B of $\mathcal{L}_{PA}$,

$$\vdash_{PA} B \quad \text{if and only if} \quad \vdash_{CPA} B. \tag{11.12}$$

Both theories are based on the same Hilbert proof system H, so to prove (11.12) means to prove that

(1) all axioms $C1 - C8$ of CPA are provable in PA and vice versa,

(2) all axioms $P1 - P8$ of $\mathcal{L}_{PA}$ are provable in CPA.

Here are detailed proofs for axioms P1, and C1. The proofs for other axioms follow the same pattern.

(1) We prove that the axiom C1 $\forall x\forall y\forall z(x = y \Rightarrow (y = z \Rightarrow x = z))$ is provable in PA as follows.

Observe that axioms of CPA are closures of respective axioms of PA. Consider axiom P1: $(x = y \Rightarrow (y = z \Rightarrow x = z))$. As the proof system H has a generalization rule 11.10

$$(G)\ \ \frac{A(x)}{\forall x A(x)}$$

as its rule of inference, we obtain a proof $B1, B2, B3, B4$ of C1 as follows.

B1: $(x = y \Rightarrow (x = z \Rightarrow y = z))$, (axiom P1)

B2: $\forall z(x = y \Rightarrow (x = z \Rightarrow y = z))$, (GA)

B3: $\forall y\forall z(x = y \Rightarrow (x = z \Rightarrow y = z))$, (GA)

B4: $\forall x\forall y\forall z(x = y \Rightarrow (x = z \Rightarrow y = z))$. (axiom C1)

This **ends the proof** of (1) for axioms P1, and C1.

(2) We prove that the axiom P1 $(x = y \Rightarrow (y = z \Rightarrow x = z))$ of $\mathcal{L}_{PA}$ is provable in CPA as follows.

By H completeness a predicate tautology

$$(\forall x A(x) \Rightarrow A(t)), \tag{11.13}$$

where term t is free for x in $A(x)$ is provable in H for any formula $A(x)$ of $\mathcal{L}$ and hence for any formula $A(x)$ of its particular sublanguage $\mathcal{L}_{PA}$. So its particular case for $A(x) = (x = y \Rightarrow (x = z \Rightarrow y = z))$ and $t = x$ is provable in CPA, i.e. we have that the formula

$$(\forall x\forall y\forall z(x = y \Rightarrow (x = z \Rightarrow y = z)) \Rightarrow \forall y\forall z(x = y \Rightarrow (x = z \Rightarrow y = z)))$$

is provable in CPA.

We construct a proof $B1, B2, B3, B4, B5, B6, B7$ of P1 in CPA in as follows.

B1 $\forall x\forall y\forall z(x = y \Rightarrow (x = z \Rightarrow y = z))$, (axiom C1)

B2 $(\forall x\forall y\forall z(x = y \Rightarrow (x = z \Rightarrow y = z)) \Rightarrow \forall y\forall z(x = y \Rightarrow (x = z \Rightarrow y = z)))$, by (11.15)

B3 $\forall y\forall z(x = y \Rightarrow (x = z \Rightarrow y = z))$, MP on B1, B2

B4 $(\forall y\forall z(x = y \Rightarrow (x = z \Rightarrow y = z)) \Rightarrow \forall z(x = y \Rightarrow (x = z \Rightarrow y = z)))$, by (11.15)

B5 $\forall z(x = y \Rightarrow (x = z \Rightarrow y = z))$, MP on B3, B4

B6 $(\forall z(x = y \Rightarrow (x = z \Rightarrow y = z)) \Rightarrow (x = y \Rightarrow (x = z \Rightarrow y = z)))$, by (11.15)

B7 $(x = y \Rightarrow (x = z \Rightarrow y = z))$ MP on B5, B6

This **ends the proof** of (2) for axioms P1, and C1.
The proofs for other axioms is similar and are left as homework assignment.

Here are some more basic facts about PA.

Fact 11.3

The following formulas are provable in PA for any terms t, s, r of $\mathcal{L}_{PA}$.

P1' $(t = r \Rightarrow (t = s \Rightarrow r = s))$,

P2' $(t = r \Rightarrow t' = r')$,

P3' $0 \neq t'$,

P4' $(t' = r' \Rightarrow t = r)$,

P5' $t + 0 = t$,

P6' $t + r' = (t + r)'$

P7' $t \cdot 0 = 0$,

P8' $t \cdot r' = (t \cdot r) + t$.

We named the properties as P1'- P8' to stress the fact that they are generalizations of axioms P1 - P8 to the set of all terms of $\mathcal{L}_{PA}$.
Proof
We write the proof for P1' as an example. Proofs of all other formulas follow the same pattern.
Consider axiom P1: $(x = y \Rightarrow (y = z \Rightarrow x = z))$. By Fact 11.2 its closure $\forall x\forall y\forall z(x = y \Rightarrow (x = z \Rightarrow y = z))$ is provable in Th_{PA}, i.e.

$$\vdash_{PA} \forall x\forall y\forall z(x = y \Rightarrow (x = z \Rightarrow y = z)) \qquad (11.14)$$

By H completeness a predicate tautology

$$(\forall x A(x) \Rightarrow A(t)), \qquad (11.15)$$

where term t is free for x in $A(x)$ is provable in H for any formula $A(x)$ of $\mathcal{L}$ and hence for any formula $A(x)$ of its particular sublanguage $\mathcal{L}_{PA}$. So its particular case for $A(x) = \forall y\forall z(x = y \Rightarrow (x = z \Rightarrow y = z))$ the formula (11.15) is provable in Th_{PA}. Observe that any term t is free for x in this particular $A(x)$. We get that for any term t,

$$\vdash_{PA}(\forall x\forall y\forall z(x=y \Rightarrow (x=z \Rightarrow y=z)) \Rightarrow \forall y\forall z(t=y \Rightarrow (t=z \Rightarrow y=z))), \tag{11.16}$$

Applying MP to (11.14) and (11.16) we get that for any t

$$\vdash_{PA} \forall y\forall z(t=y \Rightarrow (t=z \Rightarrow y=z)). \tag{11.17}$$

Observe that any term r is free for for y in $\forall z(t=y \Rightarrow (t=z \Rightarrow y=z))$. so we have that for all terms r

$$\vdash_{PA}(\forall y\forall z(t=y \Rightarrow (t=z \Rightarrow y=z)) \Rightarrow \forall z(t=r \Rightarrow (t=z \Rightarrow r=z))), \tag{11.18}$$

as a particular case of 11.15). Applying MP to (11.17) and (11.18) we get that for any terms t, r

$$\vdash_{PA} \forall z(t=r \Rightarrow (t=z \Rightarrow r=z)). \tag{11.19}$$

Observe that any term s is free for for z in $(t=r \Rightarrow (t=z \Rightarrow r=z))$. so we have that

$$\vdash_{PA}(\forall z(t=y \Rightarrow (t=z \Rightarrow y=z)) \Rightarrow (t=r \Rightarrow (t=s \Rightarrow r=s))), \tag{11.20}$$

for all terms r, t, s as a particular case of 11.15). Applying MP to (11.19) and (11.20) we get that for any t, r

$$\vdash_{PA}(t=r \Rightarrow (t=s \Rightarrow r=s)).$$

This **ends the proof** of P'.
The proofs of properties P2' - P8' follow the same pattern and are left as a homework assignment.

Fact 11.4

The following formulas are provable in PA for any terms t, s, r of $\mathcal{L}_{PA}$.

a1 $t=t$,

a2 $(t=r \Rightarrow r=t)$,

a3 $(t=r \Rightarrow (r=s \Rightarrow t=s))$,

a4 $(r=t \Rightarrow (t=s \Rightarrow r=s))$,

a5 $(t=r \Rightarrow (t+s=r+s))$,

a6 $t=0+t$.

Proof
We use in the proof Fact 11.2, Fact 11.3, axioms of PA (11.11, and completeness of the system H. We denote it in the comments. The details of the steps

are similar to the proof of Fact 11.3 and is left to the reader as as homework assignment.

a1 We construct a proof of $t = t$ in CPA in as follows.

B1 $t + 0 = t$, P5' in Fact 11.3

B2 $(t + 0 = t \Rightarrow (t + 0 = t \Rightarrow t = t))$, P1' in Fact 11.3 for $t = t + 0, r = t, s = t$

B3 $(t + 0 = t \Rightarrow t = t)$, MP on B1, B2

B4 $t = t$. MP on B1, B3

a2 We construct a proof of $(t = r \Rightarrow r = t)$ as follows.

B1 $(t = r \Rightarrow (t = t \Rightarrow r = t))$, P1' in Fact 11.3 for $r = t, s = t$

B2 $(t = t \Rightarrow (t = r \Rightarrow r = t))$, B1, tautology

B3 $t = r \Rightarrow r = t$. MP on B2, a1

a3 We construct a proof of $(t = r \Rightarrow (r = s \Rightarrow t = s))$ as follows.

B1 $(r = t \Rightarrow (r = s \Rightarrow t = s))$, P1' in Fact 11.3

B2 $t = r \Rightarrow r = t$, a2

B3 $(t = r \Rightarrow r = t)$. MP on B1, B2

a4 We construct a proof of $(r = t \Rightarrow (t = s \Rightarrow r = s))$ as follows.

B1 $(r = t \Rightarrow (t = s \Rightarrow r = s))$, a3 for $t = r, r = t$

B2 $(t = s \Rightarrow (r = t \Rightarrow r = s))$, B1, tautology

B3 $s = t \Rightarrow t = s$, a2

B4 $(s = t \Rightarrow (r = t \Rightarrow r = s))$, B1, B2, tautology

B5 $(r = t \Rightarrow (t = s \Rightarrow r = s))$. B4, tautology

a5 We prove $(t = r \Rightarrow (t + s = r + s))$ by the Principle of Mathematical Induction P9 $(A(0) \Rightarrow (\forall x(A(x) \Rightarrow A(x') \Rightarrow \forall x A(x))))$.

The proof uses the Deduction Theorem which holds for the proof system H (Remark 11.3) and so can be use in PA.

We first apply the Induction Rule to $A(z) : (x = y \Rightarrow x + z = y + z)$ to prove

$$\vdash_{PA} \forall z(x = y \Rightarrow x + z = y + z).$$

(i) We prove that $\vdash_{PA} A(0)$, i.e. $\vdash_{PA} (x = y \Rightarrow x + 0 = y + 0)$. Here the steps in the proof.

B1 $x + 0 = x$, P5'

B2 $y + 0 = y$, P5'

B3 $x = y$, Hyp

B4 $(x + 0 = x \Rightarrow (x = y \Rightarrow x + 0 = y))$, a3 for $t = x + 0, r = x, s = y$

B5 $(x = y \Rightarrow x + 0 = y)$, MP on B1, B4

B6 $x + 0 = y$, MP on B3, B5

B7 $(x{+}0 = y \Rightarrow (y{+}0 = y \Rightarrow x{+}0 = y{+}0)$, a4 for $r = x{+}0, t = y, s = y = 0$

B8 $(y + 0 = y \Rightarrow x + 0 = y + 0)$, MP on B6, B7

B9 $x + 0 = y + 0)$, MP on B2, B8

B10 $(x = y \Rightarrow x + 0 = y + 0)$. B1- B9, Deduction Theorem

Thus, $\vdash_{PA}$ $A(0)$.

(ii) We prove that $\vdash_{PA} \forall z(A(z) \Rightarrow A(z'))$, i.e.

$\forall z((x = y \Rightarrow x + z = y + z) \Rightarrow (x = y \Rightarrow x + z' = y + z'))$. Here the steps in the proof.

C1 $(x = y \Rightarrow x + z = y + z)$, Hyp

C2 $x = y$, Hyp

C3 $x + z' = (x + z)'$, P6'

C4 $y + z' = (y + z)'$, P6'

C5 $x + z = y + z)$, MP on B1, B2

C6 $(x + z = y + z \Rightarrow (x + z)' = (y + z)')$ P2' for $t = x + z, r = y + z$,

C7 $(x + z)' = (y + z)'$, MP on B5, B6

C8 $x + z' = y + z'$, a3 substitution and MP on B3, B7

C9 $((x = y \Rightarrow x + z = y + z) \Rightarrow x + z' = y + z')$ B1- B8, Deduction Theorem

This proves $\vdash A(z) \Rightarrow A(z')$.

C10 $(((x = y \Rightarrow x + 0 = y + 0) \Rightarrow ((x = y \Rightarrow x + z = y + z) \Rightarrow x + z' = y + z')) \Rightarrow \forall z(x = y \Rightarrow x + z = y + z))$, P9 for $A(z) : (x = y \Rightarrow x + z = y + z)$

C11 $((x = y \Rightarrow x + z = y + z) \Rightarrow x + z' = y + z')) \Rightarrow \forall z(x = y \Rightarrow x + z = y + z)$, MP on C10 and B10

C12 $\forall z(x = y \Rightarrow x + z = y + z)$, MP on C11 and C9

C13 $\forall y \forall z(x = y \Rightarrow x + z = y + z)$, (GA)

C14 $\forall x \forall y \forall z(x = y \Rightarrow x + z = y + z)$, (GA)

Now we repeat here the proof of P1' of Fact 11.3. We apply it step by step to

C14. We eliminate the quantifiers $\forall x \forall y \forall z$ and replace variables x, y, z by terms t, r, s using the tautology (11.15) $(\forall x A(x) \Rightarrow A(t))$ and Modus Ponens. Finally, we obtain the proof of a5, i.e.

$$\vdash_{PA} (t = r \Rightarrow (t + s = r + s)).$$

We go on proving other basic properties of addition and multiplications including for example the following.

Fact 11.5

The following formulas are provable in PA for any terms t, s, r of $\mathcal{L}_{PA}$.

(i) $t \cdot (r + s) = (t \cdot r) + (t \cdot s)$, *distributivity*

(ii) $(r + s) \cdot t = (r \cdot t) + (s \cdot t)$, *distributivity*

(iii) $(r \cdot t) \cdot s = r \cdot (t \cdot s)$, *associativity of* $\cdot$

(iv) $(t + s = r + s \Rightarrow t = r)$, *canlcellation law for* $+$

Proof
(i) Prove $\vdash_{PA} t \cdot (x + z) = (x \cdot y) + (x \cdot z)$ by induction on z.
(ii) Prove from (i) and property $t \cdot r = r \cdot t$.
(iii) Prove $\vdash_{PA} (x \cdot y) \cdot z = x \cdot (y \cdot z)$ by induction on z.
(iv) Prove $(x + z = y + z \Rightarrow x = y)$ by induction on z

Definition 11.7 (Numerals)

The terms $0,\ 0',\ 0'',\ 0''',\dots$ *are called* numerals *and denoted by* $\overline{0},\ \overline{1},\ \overline{2},\ \overline{3},\dots$. *More precisely,*

(1) $\overline{0}$ *is* 0,

(2) for any natural number n, $\overline{n+1}$ *is* $(\overline{n})'$.

In general, if n is a natural number, $\overline{n}$ stands for the corresponding numeral 0" ...$'$, i.e. by 0 followed by n strokes.

The numerals can be defined recursively as follows.
(1) 0 is a numeral,
(2) if u is a numeral, then u' is also a numeral.

Here are some more of many properties, intuitively obvious, that provable in Th_{PA}. We give some proofs and an example, and leave the others as an exercise.

Reminder
We use numerals $\overline{n},\ \overline{m}$ as un **abbreviation** of the terms r, s they represent.

Fact 11.6 *The following formulas are provable in PA for any terms t, s of* $\mathcal{L}_{PA}$.

1. $t + \overline{1} = t'$,
2. $t \cdot \overline{1} = t$,
3. $t \cdot \overline{2} = t + t$,
4. $(t + s = 0 \Rightarrow (t = 0 \cap s = 0))$,
5. $(t \neq 0 \Rightarrow (s \cdot t = 0 \Rightarrow s = 0))$,

Proof
1. Major steps in the proof of $t + \overline{1} = t'$ in PA are as follows.
The comments at each step explain how to reconstruct the formal proof from the properties already proven.

B1 $t + 0' = (t + 0)'$, P6'

B2 $t + 0 = t$, P5'

B3 $(t + 0)' = t'$, B2, P2', MP

B4 $t + 0' = t'$, B1, B3, Fact 11.4 a3, MP

B5 $t + \overline{1} = t'$. B4, abbreviation

2. Major steps in the proof of $t \cdot \overline{1} = t$ in PA are as follows.

B1 $t \cdot 0' = t \cdot 0 + t$, P8'

B2 $t \cdot 0 = 0$, P7'

B3 $(t \cdot 0) + t = 0 + t$, B1, Fact 11.4 a4, MP

B4 $t \cdot 0' = 0 + t$, B1, B3, Fact 11.4 a3, MP

B5 $0 + t = t$, Fact 11.4 a3, a6, MP

B6 $t \cdot 0' = t$, B4, B5, Fact 11.4 a3, MP

B7 $t + \overline{1} = t'$, B6, abbreviation

3. Major steps in the proof of $t \cdot \overline{2} = t + t$ in PA are as follows.

B1 $t + \overline{1'} = (t \cdot \overline{1'}) + t$, P8'

B2 $t + \overline{1} = t'$, part 2.

B3 $(t \cdot \overline{1}) + t = t + t$, B2, Fact 11.4 a5, MP

B4 $t \cdot \overline{1'} = t + t$, B1, B3, Fact 11.4 a3, MP

B5 $t \cdot \overline{2} = t + t$, B4, abbreviation

4. We prove $(t + s = 0 \Rightarrow (t = 0 \cap s = 0))$ by the following steps.

(s1) We apply the Principle of Mathematical Induction to $A(y) : (x + y = 0 \Rightarrow$

$(x = 0 \cap y = 0))$ and prove

$$\forall y(x + y = 0 \Rightarrow (x = 0 \cap y = 0)). \tag{11.21}$$

(s2) We apply the generalization rule (G) to (5.24) and get

$$\forall x \forall y(x + y = 0 \Rightarrow (x = 0 \cap y = 0)). \tag{11.22}$$

(s3) We repeat here the proof of P1' of Fact 11.3. We apply it step by step to (11.22). We eliminate the quantifiers $\forall x \forall y$ and replace variables x, y by terms t, s using the tautology (11.15) $(\forall x A(x) \Rightarrow A(t))$ and Modus Ponens. Finally, we obtain the proof of 4., i.e.

$$\vdash_{PA} (t + s = 0 \Rightarrow (t = 0 \cap s = 0)).$$

We are going to prove now, as an example, the following.

Fact 11.7

Let n, m be any natural numbers.

(1) *If $m \neq n$, then $\overline{m} \neq \overline{n}$.*

(2) $\overline{m + n} = \overline{m} + \overline{n}$ *and* $\overline{m \cdot mn} = \overline{m} \cdot \overline{n}$ *are provable in PA .*

(3) *Any model for PA is infinite.*

Proof
Assume $m \neq n$, then $m < n$ or $n < m$. Assume $m < n$. Her are major steps in the formal proof of $\overline{n} \neq \overline{m}$.

The proof uses the Deduction Theorem which holds for the proof system H (Remark 11.3) and so can be use in PA.

B1 $\overline{m} = \overline{n}$, Hyp

B2 $0''''' = 0''''$ B2 is abbreviation of B2 for m applications of $'$ on left side of equation and n applications of $'$ on the right

B3 $0 = t'$, for $t = \overline{n - m - 1}$

We apply P4' m times a=in a row. Then $0 = 0'''$ with $n - m$ applications of $'$ on right side. Let $t = \overline{n - m - 1}$. Since $n > m$, $n - m - 1 \geq 0$. Thus, $0 = t'$.

B4 $0 \neq t'$, P3'

B5 $0 = t' \cap 0 \neq t'$, B3, B4, tautology $(A \Rightarrow (B \Rightarrow (A \cap B)))$, MP

B6 $(\overline{m} = \overline{n} \Rightarrow 0 = t' \cap 0 \neq t')$, B1, B5, Deduction Theorem

B7 $m \neq n$ B6, tautology $((A \Rightarrow (C \cap \neg C)) \Rightarrow \neg A)$, MP

The proof of the case $n < m$ is similar and left to the reader.

(2) We use mathematical induction for natural numbers in the metalanguage with respect to natural number n. Base case: $\overline{m+0}$ is $\overline{m}$.
By P3' $\overline{m} = \overline{m} + \overline{0}$, hence $\overline{m+0} = \overline{m} + \overline{0}$ and the base step holds.

Inductive step. Assume that $\overline{m+n} = \overline{m} + \overline{n}$ is provable. By P2' and P6' we get $\overline{(m+n)'} = \overline{m} + (\overline{n})'$. But $\overline{m+(n+1)}$ is $(\overline{m+n})'$ and $\overline{n+1}$ is $(\overline{n})'$. Hence, $\overline{m+(n+1)} = \overline{m} + \overline{n+1}$ and by mathematical induction $\overline{m+n} = \overline{m} + \overline{n}$ is provable in Th_{PA}, for all n, m. The proof that $\overline{m \cdot mn} = \overline{m} \cdot \overline{n}$ is provable in PA for all n, m is similar.

(3) By (2), in a model for PA the objects corresponding to numerals must be distinct. But the set of numerals is infinitely countable, so universe of any model for PS must contain infinitely countable subset and hence is infinite.

An order relation can be introduced by in PA as follows.

Definition 11.8 (Order)

For any terms t, s of $\mathcal{L}_{PA}$, we write

$t < s$ *for a formula* $\exists w(w \neq 0 \cap w + t = s)$,
where we choose w to be the first variable not in t or s,

$t \leq s$ *for a formula* $t < s \cup t = s$,

$t > s$ *for a formula* $s < t$,

$t \geq s$ *for a formula* $s \leq t$,

$t \not< s$ *for a formula* $\neg(t < s)$, *and so on...*

Then we prove properties of order relation, for example the following.

Fact 11.8

For any terms t, r, s of $\mathcal{L}_{PA}$, the following formulas are provable in PA.

o1 $t \leq t$,

o2 $(t \leq s \Rightarrow (s \leq r \Rightarrow t \leq r))$,

o3 $((t \leq s \cap s \leq t) \Rightarrow t = s)$,

o4 $(t \leq s \Rightarrow (t + r \leq s + r))$,

o5 $(r > 0 \Rightarrow (t > 0 \Rightarrow r \cdot t > 0))$.

There are several stronger forms of the the Principle of Mathematical Induction P9 $(A(0) \Rightarrow (\forall x(A(x) \Rightarrow A(x') \Rightarrow \forall x A(x))))$ that are provable in PA. Here is one of them.

Fact 11.9 (Complete Induction)

The following formula, called Complete Induction Principle is provable in PA.

$$(\forall x \forall z(z < x \Rightarrow A(z)) \Rightarrow A(x)) \Rightarrow \forall x A(x)).$$

In plain English, Complete Induction Principle says:
consider a property **P** such that , for any x, if **P** holds for for all natural numbers less then x, then **P** holds for x also. Then **P** holds for all natural numbers.

We proved and cited only some of the basic properties corresponding to properties of arithmetic of natural numbers. There are many more of them, developed in many Classical Logic textbooks. We refer the reader especially to Mendelson (1997) that we found the most rigorous and complete. The proofs included here are more precise and complete versions of the few of the Mendelson's proofs.

We selected and proved some direct consequences Peano Arithmetic axioms not only because they are needed as the starting point for a strict development of the formal theory of arithmetic of natural numbers but also because they are good examples of how one develops any formal theory.

From this point on one can generally translate onto the language $\mathcal{L}_{PA}$ and prove in the PA the results from any text on elementary number theory. Some standard results of number theory are proved with the aid of theory of complex variables and it is often not known whether elementary proofs (or proofs in PA can be given for such theorems. The statements of some other results of number theory cannot even be formulated in PA.

Hence a natural question about the *strength* and *expressive powers* of PA is a very important one. We will address it shortly in next section with connection of the formulation and proofs of Gödel Theorems. Gödel, in order to prove them developed the huge scientific apparatus which grew into new field of Mathematics of *Recursion Theory*, and into *Theory of Computation* with input from Church and Turing.

We know by Ryll Nardzewski Theorem 11.1 that PA is not finitely axiomatizable. We want to bring reader's attention a finitely axiomatizable proper sub-theory of PA, RR, that has the same expressive power with respect to the Gödel Theorems. Here it is, as formalized and discussed in detail in Mendelson's book.

$T16$. **Robinson System** RR (11.23)

Language

The language of RR is the same as the language of PA, i.e.

$$\mathcal{L}_{RR} = \mathcal{L}_{\{\neg, \Rightarrow, \cup, \cap\}}(\{=\},\ \{',\ +,\ \cdot\},\ \{0\}).$$

Specific Axioms

r1 $x = x$,

r2 $(x = y \Rightarrow y = x)$,

r3 $(x = y \Rightarrow (y = z \Rightarrow x = z))$,

r4 $(x = y \Rightarrow x' = y')$,

r5 $(x = y \Rightarrow (x + z = y + z \Rightarrow z + x = z + y))$,

r6 $(x = y \Rightarrow (x \cdot z = y \cdot z \Rightarrow z \cdot x = z \cdot y))$,

r7 $(x' = y' \Rightarrow x = y)$,

r8 $0 \neq x'$,

r9 $(x \neq 0 \Rightarrow \exists y\ x = y')$,

r10 $x + 0 = x$,

r11 $x + y' = (x + y)'$,

r12 $x \cdot 0 = 0$,

r13 $x \cdot y' = x \cdot y + x$,

r14 $(y = x \cdot z + p \cap ((p < x \cap y < x \cdot q + r) \cap r < x) \Rightarrow p = r)$.

for any $x, y, z, p, q, r \in VAR$,

Axioms r1 - r13 are due to Robinson (1950), hence the name. Axiom r14 is due to Mendelson (1973). It expresses the uniqueness of remainder. The relation $<$ is as defined by definition 11.8.

Gödel showed that there are closed formulas of the language $\mathcal{L}_{PA}$ of PA that are neither provable nor disprovable in PA, if PA is consistent. Hence there is a formula that is true under standard interpretation but is not provable in PA. We also see that the *incompleteness* of PA cannot be attributed to omission of some essential axiom but has deeper underlying causes that apply to other theories as well. Robinson proved in 1950, that the Gödel Theorems hold his system RR. In particular RR has the same incompleteness property as PA.

11.3 Consistency, Completeness, Gödel Theorems

Formal theories, because of their precise structure, became themselves an object of of mathematical research. The mathematical theory concerned with the study of formalized mathematical theories is called, after Hilbert, *metamathematics*. The most important open problems of metamathematics were introduced by

Hilbert as a part of the *Hilbert Program.* They were concerned with notions of consistency, completeness, and decidability. The answers to Hilbert problems were given by Gödel in 1930 in a form of his two theorem. They are some of the most important and influential results in twentieth century mathematics. We will discuss here these notions and Gödel's results.

There are two definitions of consistency: semantical and syntactical.

The **semantical** one is based on the notion of a model and says, in plain English: *a theory is consistent if the set of its specific axioms has a model.*

The **syntactical** one uses the notion of provability and says: *a theory is consistent if one can't prove a contradiction in it.*

We have used, in the proof two of the completeness theorem for propositional logic (chapter 5) the **syntactical** definition of consistency. In chapter 9, section about the reduction predicate logic to propositional logic we used the **semantical** definition. Both were defined for propositional semantics. We extend now these definitions to the predicate language, predicate semantics, and formal theories. In order to distinguish these two definitions we call the semantic one *model-consistent*, and syntactic one just *consistent.*

Definition 11.9 (Model for a Theory)

Given a first order theory (definition 11.3)

$$T = (\mathcal{L},\ \mathcal{F},\ LA,\ SA,\ \mathcal{R}).$$

Any structure $\mathcal{M} = [M, I]$ that is a model for the set SA of the specific axioms of T, i.e. such that $\mathcal{M} \models SA$, is called a **model** *for the theory T.*

Definition 11.10 (Model - Consistent Theory)

A first order theory $T = (\mathcal{L},\ \mathcal{F},\ LA,\ SA,\ \mathcal{R})$ is **model - consistent** *if and only if it has a* **model**.

Consider the Peano Arithmetics PA and a structure $\mathcal{M} = [M, I]$ for its language

$$\mathcal{L}_{PA} = \mathcal{L}_{\{\neg,\Rightarrow,\cup,\cap\}}(\{=\},\ \{',\ +,\ \cdot\},\ \{0\}),$$

such that the universe M is the set N of natural numbers (nonnegative integers) and the interpretation I is defined as follows

(1) the constant symbol 0 is interpreted as a natural number 0,

(2) the one argument function symbol $'$ (successor) is interpreted as successor operation (addition of 1) on natural numbers; $succ(n) = n + 1$,

(3) the two argument function symbols $+, \cdot$ are interpreted as ordinary addition and multiplication in N,

(4) the predicate symbol "=" is interpreted as equality relation in N.

Definition 11.11 (Standard Model)

We denote $\mathcal{M} = [N, I]$ for I defined by (1) - (4) as

$$\mathcal{M} = [N,\ =,\ succ,\ +,\ \cdot\ ] \tag{11.24}$$

and call it a **standard model** *for PA. The interpretation I defined by (1) - (4) is called a* **standard interpretation**.

Any model for PA in which the predicate symbol "=" is interpreted as equality relation in N that is not isomorphic to the standard model is called a **nonstandard model** *for PA.*

Observe that if we recognize that the set N of natural numbers with the standard interpretation, i.e. the structure (11.24) to be a model for PA, then, of course, PA is consistent (model-consistent). However, semantic methods, involving a fair amount of set-theoretic reasoning, are regarded by many (and were regarded as such by Gödel) as too precarious to serve as basis of consistency proofs. Moreover, we have not proved formally that the axioms of PA are true under standard interpretation; we only have taken it as intuitively obvious. Hence for this and other reasons it is common practice to take the model-consistency of PA as un explicit, unproved assumption and to adopt, after Gödel the following syntactic definition of consistency.

Definition 11.12 (Consistent Theory)

Given a theory $T = (\mathcal{L},\ \mathcal{F}\ \ LA,\ SA,\ \mathcal{R})$.
Let $\mathbf{T}$ be the set (11.5) of all provable formulas in T.

The theory T is **consistent** *if and only if* **there is no** *formula A of the language $\mathcal{L}_{SA}$ such that*

$$\vdash_T\ A \quad and \quad \vdash_T\ \neg A. \tag{11.25}$$

We also write the condition (11.25) as

$$A \in \mathbf{T} \quad and \quad \neg A \in \mathbf{T}.$$

Directly from definition 11.12 we get the definition of inconsistency.
We list it separately for its importance to the proof of the Gödel Theorem 11.4.

Definition 11.13 (Inconsistent Theory)

The theory $T = (\mathcal{L},\ \mathcal{F},\ LA,\ SA,\ \mathcal{R})$ is **inconsistent** *if and only if* **there is** *a formula A of the language $\mathcal{L}_{SA}$ such that*

$$\vdash_T\ A \quad and \quad \vdash_T\ \neg A.$$

Observe that the definitions 11.12, 11.13 have purely syntactic meaning. They express the common intuition what proper provability should mean. They say that a provability (formal theory) is a good one (consistent) only when one can't prove a formula and its negation; and is inconsistent when it is possible to prove a contradiction in it.

Here is one of basic characterization of consistent theories.

Theorem 11.2 (Consistent)

A theory $T = (\mathcal{L},\ \mathcal{F},\ LA,\ SA,\ \mathcal{R})$ *based on the proof system* $H = (\mathcal{L},\ \mathcal{F},\ LA,\ \mathcal{R})$ *defined by (11.1) is* **consistent** *if and only if* **there is** *a formula* A *of the language* $\mathcal{L}_{SA}$ *such that*

$$A \notin \mathbf{T}.$$

Proof
Let denote by CC the consistency condition in the definition 11.12 and by CT consistency condition in the theorem 11.2.

1. We prove implication " if CC, then CT".
Assume not CT. This means that $A \in \mathbf{T}$ for all formulas A,

$$A \in \mathbf{T} \quad \text{and} \quad \neg A \in \mathbf{T}. \tag{11.26}$$

In particular there is B such that and $B \in \mathbf{T}$ and $\neg B \in \mathbf{T}$ and not CC holds.

2. We prove implication " if CT, then CD".
Assume not CD. This means that there is A of $\mathcal{L}_{SA}$, such that $A \in \mathbf{T}$. By definition 11.12 all tautologies are derivable in $\mathbf{T}$. Hence

$$(((A \cap B) \Rightarrow C) \Rightarrow ((A \Rightarrow (B \Rightarrow C)))),\ \ ((A \cap \neg A) \Rightarrow C) \in \mathbf{T}, \tag{11.27}$$

for all $A, B, C \in \mathcal{F}$. In particular, when $B = \neg A$ we get that

$$(((A \cap \neg A) \Rightarrow C) \Rightarrow ((A \Rightarrow (\neg A \Rightarrow C)))) \in \mathbf{T}. \tag{11.28}$$

Applying MP (11.27) and (11.28) we get

$$((A \Rightarrow (\neg A \Rightarrow C))) \in \mathbf{T}. \tag{11.29}$$

Applying MP twice to (11.29) and (11.26) we get that $C \in \mathbf{T}$, for all C. We proved not CT. This **ends the proof** of 2. and of the theorem.

Theorem 11.2 often serves a following definition of consistency.

Definition 11.14

A theory T *is* **consistent** *if and only if* $\mathbf{T} \neq \mathcal{F}_{SA}$*, i.e. there is* A *of* $\mathcal{L}_{SA}$*, such that* $A \notin \mathbf{T}$.

The next important characterization of a formal theory is the one of its completeness understood as the ability of proving or disapproving any of its statements, provided it is correctly formulated in its language.

Definition 11.15 (Complete Theory)

A theory $T = (\mathcal{L},\ \mathcal{F},\ LA,\ SA,\ \mathcal{R})$ *is* **complete** *if and only if for any closed formula (sentence)* A *of the language* $\mathcal{L}_{SA}$*,*

$$\vdash_T\ A \quad or \quad \vdash_T\ \neg A. \tag{11.30}$$

We also write the condition (11.30) as

$$A \in \mathbf{T} \quad or \quad \neg A \in \mathbf{T}. \tag{11.31}$$

Directly from definition 11.15 we get the definition of incompleteness.
We list it separately for its importance to the proof of the Gödel Incompleteness Theorem 11.3, Theorem 11.6.

Definition 11.16 (Incomplete Theory)

A theory T *is* **incomplete** *if and only if there is a closed formula (sentence)* A *of the language* $\mathcal{L}_{SA}$*, such that*

$$\not\vdash_T\ A \quad and \quad \not\vdash_T\ \neg A. \tag{11.32}$$

We also write the condition (11.32) as

$$A \notin \mathbf{T} \quad and \quad \neg A \notin \mathbf{T}.$$

Any sentence A *with the property (11.32) is called an* **independent***, or* **undecidable** *sentence of the theory* T*.*

By definition 11.16, in order to prove that a given theory T is incomplete we have to construct a sentence A of $\mathcal{L}_{SA}$ and prove that either A nor $\neg A$ has a proof in it.

We are now almost ready to discuss Gödel Theorems. One of the most (if not the most) comprehensive development with detailed and strict proofs of all needed to carry proofs of Gödel Theorems can be found the Mendelson (1984) book. The Gödel Theorems chapter in Mendelson book is over 50 pages long, technically sound and beautiful. We are confident that our readers, at this stage of our book, are ready and able to follow Mendelson's or other authors' work.

We present here a short, high level approach adopting style of Smorynski's chapter in the *Handbook of Mathematical Logic*, Studies in Logic and Foundations of Mathematics, Volume 20 (1977). The chapter also is over 40 pages long (it seems to be a norm when one wants to really prove Gödel's results). It is written in a very condensed and general way and concentrates on presentation of modern results. It assumes that readers are already familiar with the traditional approach so beautifully presented in Mendelson's book, but I encourage readers to reach for it, as it is, in its own style a very interesting work.

We also want to bring to readers attention that the introduction to Smorynski's chapter contains an excellent discussion of Hilbert Program and its relationship to Gödel's results. It gives an explanation why and how devastating Gödel Theorems were to the optimism reflected in Hilbert's Consistency and Conservation Programs.

11.3.1 Hilbert's Conservation and Consistency Programs

Hilbert proposed his Conservation Program and Consistency Programs as response to L.E.J. Brouwer and Herman Weyl (1920) propagation of their theory that existence (as early as 1908) of Zermello's paradoxes free axiomatization of set theory makes the need for investigations (and proof) into consistency of mathematics superfluous. Hilbert decided to intervene. He wrote:

" .... they (Brouwer and Weil) would chop and mangle the science. If we would follow such a reform as the one they suggest, we would run the risk of losing a great part of our most valuable treasures!"

Hilbert stated his **Conservation Program** as follows: To justify the use of abstract techniques he would show - by as simple and concrete a means as possible - that the use of abstract techniques was *conservative* - i.e. that any concrete assertion one could derive by means of such abstract techniques would be derivable without them.

We follow Smorynski's clarification of some of Hilbertian jargon whose exact meaning was never defined by Hilbert. We hence talk, in the domain of mathematics, about *finitistically meaningful* statements and *finitistic* means of proof.

By the *finitistically meaningful* statements we mean for example identities of the form

$$\forall x(f(x) = g(x)),$$

where f, g are reasonably simple functions, for example primitive recursive. We will call them *real* statements. *Finitistic* proofs correspond to computations or combinatorial manipulations.

More complicated statements are called *ideal* ones and, as such, have no meaning, but can be manipulated abstractly and the use of ideal statements and abstract reasoning about them would not allow one to derive any new *real* statements, i.e. none which were not already derivable To refute Weyl and Brouwer, Hilbert required that this latter *conservation property* itself be finitistically provable.

Hilbert's **Consistency Program** asks to devise a finitistic means of proving the consistency of various formal systems *encoding* abstract reasoning with *ideal* statements.

The Consistency Program is a natural outgrowth and successor to the Conservation Program. There are two reasons for this.
R1. *Consistency* is the assertion that some string of symbols is not provable. Since derivations are simple combinatorial manipulations, this is a *finitistically meaningful* and ought to have a *finitistic proof.*

R2. Proving a consistency of a formal system encoding the abstract concepts already establishes the conservation result!

Reason R1 is straightforward. We will discuss R2 as it is particularly important.

Let's denote by R a formal systems encoding real statements with their finitistic proofs and by I the *ideal* system with its abstract reasoning.

Let A be a *real* statement $\forall x(f(x) = g(x))$.

Assume $\vdash_I A$. Then there is a derivation d of A in I. But, derivations are concrete objects and, for some real formula $P(x, y)$ *encoding* derivations in I,

$$\vdash_R P(d, \ulcorner A \urcorner),$$

where $\ulcorner A \urcorner$ is some *code* for A.

Now, if A were false, one would have $f(a) \neq g(a)$ for some a and hence

$$\vdash_R P(c, \ulcorner \neg A \urcorner)$$

for some c. In fact, one would have a stronger assertion

$$\vdash_R (f(x) \neq g(x) \Rightarrow P(c_x, \ulcorner \neg A \urcorner)).$$

But, if R proves *consistency* of I, we have

$$\vdash_R \neg(P(d, \ulcorner A \urcorner) \cap P(c, \ulcorner \neg A \urcorner)),$$

whence $\vdash_R f(x) = g(x)$, with free variable x, i.e. $\vdash_R \forall x(f(x) = g(x))$.

To make the above argument rigorous, one has to define and explain the basics of *encoding*, the assumptions on the formula $P(x, y)$ and to deliver the whole argument in a formal rigorous way, i.e. to develop rigorously the whole apparatus developed originally by Gödel and needed for the proofs of his theorems. We bring it here because it clearly invited Hilbert to establish his Consistency Program. Since Consistency Program was as broad as the general Conservation Program and, since it was more tractable, Hilbert fixed on it asserting:

"if the arbitrary given axioms do not contradict each other through their consequences, then they are true, then the objects defined through the axioms exist. That, for me, is the criterion of truth and existence".

The Consistency Program had as its goal the proof, by finitistic means of the consistence of strong systems. The solution would completely justify the use of abstract concepts and would repudiate Brouwer and Weyl.
Gödel proved that it couldn't work.

11.3.2 Gödel Incompleteness Theorems

In 1920, while in his twenties, Kurt Gödel announced that Hilbert's Consistency Program could not be carried out. He had proved two theorems which gave a blow to the Hilbert's Program but on the other hand changed the face of mathematics establishing mathematical logic as strong and rapidly developing discipline.

Loosely stated these theorems are:

Theorem 11.3 (First Incompleteness Theorem)

Let T be a formal theory containing arithmetic. Then there is a sentence A in the language of T which asserts its own unprovability and is such that:

(i) If T is consistent, then $\nvdash_T\ A$.

(ii) If T is ω- consistent, then $\nvdash_T\ \neg A$.

Theorem 11.4 (Second Incompleteness Theorem)

Let T be a consistent formal theory containing arithmetic. Then

$$\nvdash_T\ Con_T,$$

where Con_T is the sentence in the language of T asserting the consistency of T.

Observe that the Second Incompleteness Theorem destroys the Consistency Program. It states that R can't prove its own consistency, so obviously it can't prove consistency of I.

Smorynski's argument that the First Incompleteness Theorem destroys the Conservation Program is as follows. The the sentence A is real and is easily seen to be true. It asserts its own unprovability and is indeed unprovable. Thus the Conservation Program cannot be carried out and, hence, the same must hold for the Consistency Program.

M. Detlefsen in the Appendix of his book "Hilbert Program: An Essay on Mathematical Instrumentalism", Springer, 2013, argues that Smorynski's argument is ambiguous, as he doesn't tell us whether it is unprovability in R or unprovability in I. We recommend to the reader interested a philosophical discussion of Hilbert Program to read this Appendix, if not the whole book.

We will now formulate the Incompleteness Theorems in a more precise formal way and describe the main ideas behind their proofs.

Arithmetization and Encoding (11.33)

Observe that that in order to formalize the Incompleteness Theorems one has first to "translate" the sentences A and Con_T into the language of T. For the First Incompleteness Theorems 11.3 one needs to " translate " a self-referring sentence *"I am not provable in a theory T"*; for the Second Theorem 11.4 the self-referring sentence is *"I am consistent"*.

The assumption in both theorems is that T contains arithmetic means usually it contains the Peano Arithmetic PA (11.11), or even its sub-theory RR (11.23), called Robinson System. In this case the final product of such "translation" must be a sentence A or sentence Con_T of the language $\mathcal{L}_{PA}$ of PA, usually written as

$$\mathcal{L}_{PA} = \mathcal{L}(\{=\},\ \{',\ +,\ \cdot\},\ \{0\}).$$

This "translation" process into the language of some formal system containing arithmetic is called *arithmetization* and *encoding*, or *encoding* for short. We define a notion of *arithmetization* as follows.

An arithmetization of a theory T is a one-to-one function g from the set of symbols of the language of T, expressions (formulas) of T, and finite sequences of expressions of T (proofs) into the set of positive integers. The function g must satisfy the following conditions.
(1) g is effectively computable;
(2) there is an effective procedure that determines whether any given positive integer n is in the range of g and, if n is in the range of g, the procedure finds the object x such that $g(x) = m$.

Arithmetization, i.e. a method of associating numbers with symbols, expressions, and sequences of expressions was originally devised by Gödel in 1931 in order to *arithmetize* Peano Arithmetic PA and *encode* the arithmetization process PA in order to formulate and to prove his Incompleteness Theorems 11.3, 11.4.

Functions and relations whose arguments and values are natural numbers are called the number-theoretic functions and relations.

In order to arithmetize and encode in a formal system we have to

1. associate numbers with symbols symbols of the language of the system, associate numbers with expressions, and sequences of expressions of the language of the system (arithmetization, encoding of basic syntax, and encoding of syntax)

2. replace assertions *about* the system by number-theoretic statements, and express these number-theoretic statements *within* the formal system itself (arithmetization,, encoding).

We want the number - theoretic function to be *representable* PA and the predicates to be *expressible* in PA, i.e. their characteristic functions to be *representable* in PA.

The study of *representability* of functions in PA leads to the class of number-theoretic functions that turn out to be of great importance in mathematical logic, namely the x *primitive recursive* and *recursive functions.* Their definition and study in a form of a *Recursion Theory* is an important field of mathematics and of computer science which developed out of the Gödel proof of the Incompleteness Theorems.

We prove that the class of recursive functions is identical with the class of functions representable in PA, i.e. we prove: *every recursive function is representable in PA* and *every function representable in PA is recursive.*

The representability of primitive recursive and recursive functions in S in general and in PA in particular plays crucial role in the encoding process and consequently in the proof of Gödel Theorems.

The details of arithmetization and encoding are as complicated and tedious as fascinating but are out of scope of our book. We recommend Mendelson's book "Introduction to Mathematical Logic", 4th ed., Chapman & Hall (1997) as the one with the most comprehensive and detailed presentation.

Theories T and S (11.34)

We assume at this moment that T is some fixed, but for a moment unspecified **consistent** formal theory. We also assume that **encoding** is done in some fixed theory S and that T contains S, i.e. the language of T is an extension of the language of S and

$$\mathbf{S} \subseteq \mathbf{T},$$

i.e. for any formula A,

$$\text{if } \vdash_S A, \text{ then } \vdash_T A. \tag{11.35}$$

We also assume that T and S contain as constants only numerals (definition 11.7)

$$\overline{0},\ \overline{1},\ \overline{2},\ \overline{3}, \dots,$$

and T contains infinitely countably many functional and predicate symbols.

Usually S is taken to be a formal theory of arithmetic, but sometimes S can be a weak set theory. But in any case S always *contains numerals.*

We also assume that theories T and S as defined by (11.34) are such that the following Principles of Encoding (11.36) hold.

Principles of Encoding for T and S (11.36)

The mechanics, conditions and details of *encoding* for T and S for S being Peano Arithmetic PA or its sub-theory Robinson Arithmetic RR (11.23) are beautifully presented in the smallest detail in Mendelson.

The Smorynski's approach we discuss here covers a larger class of formal theories and uses a more general and modern approach. We can't include all details but we are convinced that at this stage the reader will be able to follow Smorynski's chapter in the Encyclopedia. The chapter is very well and clearly written and is now classical. We wholeheartedly recommend it as a future reading.

We also follow Smorynski approach explaining *what* is to be encoded, *where* it is to be encoded, and which are the most important encoding and provability conditions needed for the proofs of the Incompleteness Theorems.

We first encode the *syntax* of T in S.
Since encoding takes place in S, it has a sufficient supply of *constants* (countably infinite set of numerals

$$\overline{0},\ \overline{1},\ \overline{2},\ \overline{3},\dots,$$

and *closed terms* to be used as codes.

We assign to each formula A of the language of T a *closed term*,

$$\ulcorner A \urcorner$$

called the *code* of A. If $A(x)$ is a formula with a free variable x, then the code $\ulcorner A(x)\urcorner$ is a closed term encoding the formula $A(x)$, with x viewed as a *syntactic object* and not as a parameter.

We do it recursively, first we assign *codes* (unique closed terms from S) to its basic syntactic objects, i.e. elements of the alphabet of the language of T. Terms and formulas are finite sequences of these symbols and derivations (formal proofs) are also finite sequences of formulas. It means that S have to be able to *encode* and *manipulate* finite sequences. We use for such encoding a class primitive recursive functions and relations. We assume S admits a *representation* of these functions and relations and finish encoding syntax.

S will also have to have certain function symbols and we have to be able to encode them.

1. S must have we functional symbols, *neg*, *impl*, etc., corresponding to the logical connectives and quantifiers, such that, such that, for all formulas A, B of the language of T,

$$\vdash_S\ neg(\ulcorner A\urcorner),\quad \vdash_S\ impl(\ulcorner A \Rightarrow B\urcorner),\ etc.$$

An operation of substitution of a variable x in a formula A(x) by a term t is of a special importance in logic, so it must be represented in S, i.e.

2. S must have in a functional symbol *sub* that represents the substitution operator, such that for any formula $A(x)$ and term t with codes $\ulcorner A(x)\urcorner$, $\ulcorner t\urcorner$,

respectively,

$$\vdash_S \ sub(\ulcorner A(x) \urcorner, \ \ulcorner t \urcorner) = \ulcorner A(t) \urcorner. \tag{11.37}$$

Iteratation of *sub* allows one to define $sub_3, \ sub_4, \ sub_5, \dots$, such that

$$\vdash_S \ sub_n(\ulcorner A(x_1, \dots, x_n) \urcorner, \ \ulcorner t_1 \urcorner, \dots, \ulcorner t_n \urcorner) = \ulcorner A(t_1, \dots, t_n) \urcorner.$$

Finally, we have to encode derivations in S , i.e.

3. S has to have in a binary relation $Prov_T(x, \ y)$, such that for closed terms t_1, t_2,

$\vdash_S \ Prov_T(t_1, \ t_2)$ if and only if t_1 is a code of a derivation in T of the formula with a code t_2.

We read $Prov_T(x, \ y)$ as "x proves y in T " or " x is a proof of y in T".
It follows that for some closed term t,

$$\vdash_T \ A \ \text{ if and only if } \ \vdash_S \ Prov_T(t, \ \ulcorner A \urcorner).$$

We define

$$Pr_T(y) \ \Leftrightarrow \ \exists x Prov_T(x, \ y) \tag{11.38}$$

and obtain a predicate asserting provability.

However, it is not always true

$$\vdash \ T \ A \ \text{ if and only if } \ \vdash_S \ Pr_T(\ulcorner A \urcorner),$$

unless S is fairly *sound* (to be defined separately).

The encoding can be carried out, however, in such a way that the following conditions essential to the proofs of the Incompleteness Theorems hold for any sentence A of T.

Derivability Conditions (Hilbert-Bernays, 1939) (11.39)

D1 $\vdash_T \ A$ implies $\vdash_S \ Pr_T(\ulcorner A \urcorner)$.

D2 $\vdash_S \ ((Pr_T(\ulcorner A \urcorner) \Rightarrow Pr_T(\ulcorner Pr_T(\ulcorner A \urcorner) \urcorner)))$.

D3 $\vdash_S \ ((Pr_T(\ulcorner A \urcorner) \cap Pr_T(\ulcorner (A \Rightarrow B) \urcorner)) \Rightarrow Pr_T(\ulcorner B \urcorner))$.

11.4 Proof of the Incompleteness Theorems

The following theorem 11.5 is essential to the proof of the Incompleteness Theorems. It is called historically *Diagonalization Lemma* or *Fixed Point Theorem* and both names are used interchangeably. The fist name as is historically older, important for convenience of references and the second name is routinely used in computer science community.

Mendelson (1977) believes that the central idea was first explicitly mentions by Carnap who pointed out in 1934 that the result was implicit in the work of Gödel (1931). Gödel was not aware of Carnap work until 1937.

The theorem 11.5 is called Diagonalization Lemma because the argument used in its proof has some resemblance to the the *diagonal arguments* used by Cantor in 1891. He first used it proving that there are infinite sets that can not be put in one-to-one correspondence with the set on natural numbers. He then used its generalization in the proof of his famous Cantor Theorem: for every set X, its set of all subsets has a larger cardinality than X itself (see chapter 1).

In mathematics, a *fixed-point theorem* is a name of a theorem saying that a function f under some conditions, will have a at least one fixed point, i.e. a point x such that $f(x) = x$.

The theorem 11.5 says that for any formula A in the language of theory T with one free variable there is a sentence B such that the formula $(B \Leftrightarrow A(\ulcorner B \urcorner))$ is provable in T.

Intuitively, B is a *self-referential* sentence saying that B has property A. The sentence B can be viewed as a *fixed point* of the operation assigning to each formula A the sentence $A(\ulcorner B \urcorner)$. Hence the name *Fixed Point Theorem.*

Theorem 11.5 proves the existence of self-referential sentences in certain formal theories of natural numbers. These sentences then, in turn, are to be used to prove Gödel's Incompleteness Theorems. Here it is.

Theorem 11.5 (Diagonalization Lemma)

Let T, S be theories defined by (11.34).
Let $A(x)$ be a formula in the language of T with x as the only free variable. Then there is a sentence B such that

$$\vdash_S \ (B \Leftrightarrow A(\ulcorner B \urcorner)).$$

NOTE: If A, B are not in the language of S, then by $\vdash_S \ (B \Leftrightarrow A(\ulcorner B \urcorner))$ we mean that the equivalence is proved in the theoryS' in the language of T whose only non-logical axioms are those of S.

Proof
Given $A(x)$, let $(C(x) \Leftrightarrow A(sub(x, x))$ be a diagonalization of $A(x)$.

Let $m = \ulcorner C(x) \urcorner$ and $B = C(m)$.

Then we claim

$$\vdash_S \ (B \Leftrightarrow A(\ulcorner B \urcorner)).$$

For, in S, we see that

$$
\begin{aligned}
B \; &\Leftrightarrow C(m) \Leftrightarrow A(sub(m,m))\\
&\Leftrightarrow A(sub(\ulcorner C(x)\urcorner,\ m) \qquad (\text{since } m = \ulcorner C(x)\urcorner)\\
&\Leftrightarrow A(\ulcorner C(m)\urcorner) \Leftrightarrow A(\ulcorner B\urcorner) \qquad \text{by (11.37)) and } B = C(m).
\end{aligned}
$$

This proves (we leave details to the reader as a homework exercise)

$$\vdash_S \ (B \Leftrightarrow A(\ulcorner B\urcorner)).$$

Theorem 11.6 (First Incompleteness Theorem)

Let T, S be theories defined by (11.34).
Then there is a sentence G in the language of T such that:

(i) $\not\vdash_T \ G$.

(ii) *under an additional assumption,* $\not\vdash_T \ \neg A$.

Proof
Applying Diagonalization Lemma 11.5 for a formula $A(x)$ being $\neg Pr_T(x)$, where $Pr_T(x)$ is defined by (11.38) we get that there is a sentence G such that

$$\vdash_S \ (G \Leftrightarrow \ \neg Pr_T(\ulcorner G\urcorner)).$$

By the assumed property (11.35) in the definition (11.34) of T, S we have that also

$$\vdash_T \ (G \Leftrightarrow \ \neg Pr_T(\ulcorner G\urcorner)). \tag{11.40}$$

(i) We conduct the proof by contradiction. Assume $\vdash_T \ G$.
Observe that $\vdash_T \ G$ implies $\vdash_T \ Pr_T(\ulcorner G\urcorner)$ by **D1** and (11.35). This and the above (11.40) contradicts the *consistency* of T.

(ii) The *additional assumption* is assuming that the converse implication to **D1** holds, i.e that $\vdash_T \ Pr_T(\ulcorner G\urcorner)$ implies $\vdash_T \ G$.

We conduct the proof by contradiction. Assume $\vdash_T \ \neg G$.
Hence $\vdash_T \ \neg\neg Pr_T(\ulcorner B\urcorner))$ so we have that $\vdash_T \ Pr_T(\ulcorner B\urcorner))$. By the *additional assumption* it implies that $\vdash_T \ G$ what contradicts contradicting the consistency of T.
This **ends** the proof.

Observe that the sentence G is equivalent in T to an assertion that G is unprovable in T. In other words it says *" I am not provable in T"* and hence theorem 11.6 is a strict mathematical formalization of the intuitively stated theorem 11.3. We call G the Gödel's sentence.

Theorem 11.7 (Second Incompleteness Theorem)

Let T, S be theories defined by (11.34).
Let Con_T be a sentence $\neg Pr_T(\ulcorner C \urcorner))$, where is C is any contradictory statement. Then

$$\nvdash_T \ Con_T.$$

Proof
Let G the Gödel's sentence of the First Incompleteness Theorem 11.6.
We prove that

$$\vdash_T \ (Con_T \Leftrightarrow G) \tag{11.41}$$

and use it to prove that $\nvdash_T \ Con_T$. We conduct the proof by contradiction. Assume $\vdash_T \quad Con_T$. By (11.41) $\vdash_T \ (Con_T \Leftrightarrow G)$, so $\vdash_T \ G$ what contradicts the First Incompleteness Theorem 11.6.

To complete the proof we have to to prove now (11.41). We know by Logic 11.1 that

$$\vdash_T \ (Con_T \Leftrightarrow G) \ \text{ if and only if } \ \vdash_T \ (Con_T \Rightarrow G) \ \text{ and } \ \vdash_T \ (G \Rightarrow Con_T).$$

1. We prove the implication $\vdash_T \ (G \Rightarrow Con_T)$.
By definition of Con_T we have to prove now

$$\vdash_T \ (G \Rightarrow \neg Pr_T(\ulcorner C \urcorner)). \tag{11.42}$$

The formula C is a contradiction, so $(C \Rightarrow G)$ is a predicate tautology. Hence $\vdash_T \ (C \Rightarrow G)$ and by **D1**

$$\vdash_S \ Pr_T(\ulcorner (C \Rightarrow G) \urcorner).$$

We write **D3** for $A = Pr_T(\ulcorner C \urcorner)$ and $B = \vdash_S \ Pr_T(\ulcorner (C \Rightarrow G) \urcorner)$ and obtain that

$$\vdash_S \ ((Pr_T(\ulcorner C \urcorner) \cap Pr_T(\ulcorner (C \Rightarrow G) \urcorner)) \Rightarrow Pr_T(\ulcorner G \urcorner)). \tag{11.43}$$

We have by Logic 11.2

$$\vdash_S \ (Pr_T(\ulcorner C \urcorner) \Rightarrow (Pr_T(\ulcorner C \urcorner) \cap Pr_T(\ulcorner (C \Rightarrow G) \urcorner))). \tag{11.44}$$

We get from (11.44), (11.43), and Logic 11.3

$$\vdash_S \ (Pr_T(\ulcorner C \urcorner) \Rightarrow Pr_T(\ulcorner G \urcorner)). \tag{11.45}$$

We apply Logic 11.4 (contraposition) to the above (11.45) and get

$$\vdash_S \ (\neg Pr_T(\ulcorner G \urcorner) \Rightarrow \neg Pr_T(\ulcorner C \urcorner)). \tag{11.46}$$

Observe that we by the property (11.40) in the proof of the First Incompleteness Theorem 11.3 we have

$$\vdash_S \ (G \Rightarrow \ \neg Pr_T(\ulcorner G \urcorner)). \tag{11.47}$$

We put (11.46) and (11.47) together and get

$$\vdash_S \ (G \Rightarrow \ \neg Pr_T(\ulcorner G \urcorner)) \quad \text{and} \quad \vdash_S \ (\neg Pr_T(\ulcorner G \urcorner) \Rightarrow \neg Pr_T(\ulcorner C \urcorner)).$$

Applying Logic 11.4 to the above we get $\vdash_S \ (G \Rightarrow \neg Pr_T(\ulcorner C \urcorner))$. But by C is by definition Con_T and hence we have proved the $\vdash_S \ (G \Rightarrow Con_T)$ and hence also

$$\vdash_T \ (G \Rightarrow Con_T).$$

2 . We prove now $\vdash_T \ (Con_T \Rightarrow G)$, i.e. the implication

$$\vdash_T \ (\neg Pr_T(\ulcorner C \urcorner) \Rightarrow G). \tag{11.48}$$

Here is a concise proof. We leave it to the reader as an exercise to write a detailed version.

By **D2**,

$$\vdash_S \ ((Pr_T(\ulcorner G \urcorner) \Rightarrow Pr_T(\ulcorner Pr_T(\ulcorner G \urcorner) \urcorner))).$$

This implies

$$\vdash_S \ (Pr_T(\ulcorner G \urcorner) \Rightarrow Pr_T(\ulcorner \neg G \urcorner)),$$

by **D1**, **D3**, since $\vdash_S \ (G \Rightarrow \ \neg Pr_T(\ulcorner G \urcorner))$.
This yields

$$\vdash_S \ ((Pr_T(\ulcorner G \urcorner) \Rightarrow Pr_T(\ulcorner (G \cap \neg G) \urcorner)),$$

by **D1**, **D3**, and logic properties, which imples

$$\vdash_S \ ((Pr_T(\ulcorner G \urcorner) \Rightarrow Pr_T(\ulcorner C \urcorner)),$$

by **D1**, **D3**, and logic properties. By Logic 11.4 (contraposition)

$$\vdash_S \ (\neg Pr_T(\ulcorner G \urcorner) \Rightarrow \neg Pr_T(\ulcorner C \urcorner)),$$

which is $\vdash_S \ (Con_T \Rightarrow G)$ and hence also

$$\vdash_T \ (Con_T \Rightarrow G).$$

This **ends** the proof.

We prove now, as an exercise and reminder, the steps in the proof of part **1.** that follow the predicate logic properties, hence the name Logic. The discovery of needed properties and their proofs for the part **2.** is left as a homework exercise.

Remark 11.4

By definition 11.3 the theories T, S are based on a complete proof system for predicate logic and by the monotonicity of classical consequence everything provable there is provable in T, S. In particular all predicate tautologies are provable in T and in S.

Logic 11.1

Given a complete proof system H, for any formulas A, B of the language of H,

$$\vdash (A \Leftrightarrow B) \ \textit{ if and only if } \ \vdash (A \Rightarrow B) \ \textit{ and } \vdash (B \Rightarrow A).$$

Proof
1. We prove implication *if* $\vdash (A \Leftrightarrow B)$, *then* $\vdash (A \Rightarrow B)$ *and* $\vdash (B \Rightarrow A)$.
Directly from provability of a tautology $((A \Leftrightarrow B) \Rightarrow ((A \Rightarrow B) \cap (B \Rightarrow A)))$, assumption $\vdash (A \Leftrightarrow B)$, and MP we get $\vdash ((A \Rightarrow B) \cap (B \Rightarrow A))$. Consequently, from $\vdash ((A \Rightarrow B) \cap (B \Rightarrow A))$, provability of tautologies $((A \cap B) \Rightarrow A)$, $((A \cap B) \Rightarrow B)$ and MP applied twice we get $\vdash (A \Rightarrow B), \vdash (B \Rightarrow A)$.

2. We prove implication *if* $\vdash (A \Rightarrow B)$ *and* $\vdash (B \Rightarrow A)$, *then* $\vdash (A \Leftrightarrow B)$.
Directly from provability of tautology $((A \Rightarrow B) \Rightarrow ((B \Rightarrow A) \Rightarrow (A \Leftrightarrow B)))$, assumption $\vdash (A \Rightarrow B), \vdash (B \Rightarrow A)$, MP applied twice we get $\vdash (A \Leftrightarrow B)$.

Logic 11.2 *Given a complete proof system H, for any formulas A, B of the language of H,*

$$\vdash (A \Rightarrow (A \cup B)) \ \textit{ and } \ \vdash (A \Rightarrow (B \cup A)).$$

Proof Directly from predicate tautologies $(A \Rightarrow (A \cup B)), (A \Rightarrow (B \cup A))$ and completeness.

Logic 11.3

Given a complete proof system H, for any formulas A, B of the language of H,

$$\textit{if } \ \vdash (A \Rightarrow B) \ \textit{ and } \vdash (B \Rightarrow C), \ \textit{ then } \ \vdash (A \Rightarrow C).$$

Proof From completeness and predicate respective tautology we get

$$\vdash ((A \Rightarrow B) \Rightarrow ((B \Rightarrow C) \Rightarrow (A \Rightarrow C))). \tag{11.49}$$

Assume $(A \Rightarrow B)$. Applying MP to (11.49) twice we get the proof of $(A \Rightarrow C)$.

Logic 11.4

Given a complete proof system H, for any formulas A, B of the language of H,

$$\vdash (A \Rightarrow B) \ \textit{ if and only if } \ \vdash (\neg B \Rightarrow \neg A).$$

Proof Directly from predicate tautology $((A \Rightarrow B) \Leftrightarrow (\neg B \Rightarrow \neg A))$, completeness and MP.

Observation 11.2

We proved, a part of proof of the Second Incompleteness Theorem 11.7 the equivalence (11.41) which says that the self-referential Gödel sentence G which asserts its own unprovability is equivalent to the sentence asserting consistency. Hence, the sentence G is unique up to provable equivalence (11.41) and we can say that G is the sentence that asserts its own unprovability.

ω-consistency

We used, in the part (ii) of the First Incompleteness Theorem 11.6, an additional assumption that $\vdash_T \ Pr_T(\ulcorner G \urcorner)$ implies $\vdash_T \ G$, instead of a habitual assumption of ω-consistency.

The concept of ω-consistency was introduced by Gödel for purpose of stating assumption needed for the proof of his First Incompleteness Theorem 11.3. The modern researchers proved that the assuption of the ω-consistency can be replaced, as we did, by other more general better suited for new proofs conditions.

Informally, we say that T is ω- **consistent** if the following two conditions are *not* satisfied for any formula A:

(i) $\vdash_T \exists x A(x)$;

(ii) $\vdash_T \neg A(\overline{n})$ for every natural number n.

Formally, ω-consistency can be represented (in varying degrees of generality) by (modification of) the following formula

$$(Pr_T\ (\ulcorner \exists x A(x) \urcorner) \Rightarrow \exists x\ \neg Pr_T(\ulcorner \neg A(x) \urcorner)). \tag{11.50}$$

11.4.1 The Formalized Completeness Theorem

Proving completeness of a proof system with respect to a given semantics is the first and most important goal while developing a logic and was the central focus of our study. So we now conclude our book with presentation the formalized completeness theorem 11.8. We discuss its proof and show how to use it to give new type of proofs, called model-theoretic proofs, of the incompleteness theorems for Peano Arithmetic PA, i.e. for the case when $S = PA$.

Formalizing the proof of completeness theorem for classical predicate logic from chapter 9 within PA we get the following.

Theorem 11.8 (Hilbert-Bernays Completeness Theorem)

Let U be a theory with a primitive recursive set of axioms.
There is a set Tr_M of formulas such that in $PA + Con_U$ one can prove that this

set Tr_M defines a model M of U:

$$\vdash_{PA+Con_U} \forall x(Pr_U(x) \Rightarrow Tr_M(x)) \tag{11.51}$$

Moreover the set Tr_M is of type Δ_2.

The Hilbert-Bernays Completeness Theorem 11.8 asserts that modulo Con_U, one can prove in PA the existence of a model of U whose truth definition is of type Δ_2. Its proof is just an arithmetization of the Henkin proof presented in chapter 9. Following the Henkin proof one adds to the language of U an infinite primitive recursive set of new constants

$$c_0,\ c_1,\ c_2\ \ldots,$$

and adds the axiom (Henkin Axiom)

$$(\exists x A(x) \Rightarrow A(c_{A[x]})) \tag{11.52}$$

for each formula $A(x)$. One then enumerates sentences

$$A_0,\ A_1,\ A_2,\ \ldots \tag{11.53}$$

in this augmented language and defines a *complete theory* by staring with $U$and adding at each step n a sentence A_n, or $\neg A_n$ according to whether A_n is *consistent* with what has been chosen before or not.

The construction is then described within PA. Assuming Con_U one can also prove that the construction never terminates. The resulting set of sentences forms a *complete theory* which by axioms (11.52) forms a *model* of U. Inspection shows that the truth definition Tr_M of type Δ_2.

The Hilbert-Bernays Completeness Theorem 11.8 makes possible to conduct new type of proofs of the incompleteness theorems, model- theoretic proofs. Gödel chose as the self-referring sentence a **syntactic statement** " I do not have a proof". He did not want (and saw difficulties with) to use the sentence involving the notion of truth, i.e. the sentence "I am not true". The new proofs use exactly this and this is why they are called **model-theoretic** proofs.

Dana Scott was the first to observe that one can give a **model- theoretic** proof of the First Incompleteness Theorem. Here is the theorem and his short proof.

Theorem 11.9 (First Incompleteness Theorem)

Let PA be a Peano Arithmetic.
There is a sentence G of PA, such that

(i) $\nvdash_{PA}\ G$;

(ii) $\nvdash_{PA}\ \neg G$.

Proof
Assume PA is complete. Then, since PA is true, $\vdash_{PA} Con_{PA}$ and we can apply the completeness theorem 11.8 to obtain a formula Tr_M which gives a truth definition for the model of PA. Observe that once PA is complete we have that Pr_{PA} is Tr_M. We choose G by

$$\vdash_{PA} (G \Leftrightarrow \neg Tr_M(\ulcorner G \urcorner)). \tag{11.54}$$

We claim $\nvdash_{PA} G$, $\nvdash_{PA} \neg G$. For if $\vdash_{PA} G$, then $\vdash_{PA} Tr_M(\ulcorner G \urcorner))$. By (11.54) $\vdash_{PA} \neg G$. Contradiction. Similarly, $\vdash_{PA} \neg G$ implies $\vdash_{PA} G$.

Observe that the sentence G as defined by (11.54) asserts *"I am not true"*.

Scott 's proof differs from the proof of the First Incompleteness Theorem 11.6 not only by the choice of the **model- theoretic** method, but also by be a choice of the **model- theoretic** sentence G.

Let's compare these two independent sentences G:
the classic syntactic one of theorem 11.6 representing statement *" I do not have a proof"* and
the model- theoretic one of theorem 11.9 representing statement *"I am not true"*.

Property 11.1

The sentence G_S of the First Incompleteness Theorem 11.6 asserting its own provability *is*
(i) *unique up to provable equivalence (Observation 11.2);*
(ii) *the sentence is* Π_1 *and hence true.*

The sentence G of the First Incompleteness Theorem 11.6 asserting its own falsity *in the model constructed is*
(iii) *not unique - for the following implication holds*

$$\textit{if } (G \Leftrightarrow \neg Tr_M(\ulcorner G \urcorner)), \textit{ then } (\neg G \Leftrightarrow \neg Tr_M(\ulcorner \neg G \urcorner)).$$

(iv) *the sentence is* Δ_2 *(theorem 11.8, and, by* (iii) *there is no obvious way od deciding its truth or falsity.*

Georg Kreisler was the first to present a **model- theoretic** proof of the following.

Theorem 11.10 (Second Incompleteness Theorem)

Let PA be a Peano Arithmetic. $\nvdash_{PA} Con_{PA}$.

The proof is uses, as did the proof of Hilbert-Bernays Completeness Theorem 11.8 the arithmetization of Henkin proof of completeness theorem presented in chapter 9. The proof is carried by contradiction. We assume $\vdash_{PA} Con_{PA}$.

Then we show, for any presentation of the Henkin proof construction (as given by encoding, the enumeration of sentences (11.53) . . . etc.) there is a number m such that, for any model $\mathcal{N}$ of PA, the sequence of models determined by the given presentations must *stop* after fewer then m steps with a model in which Con_{PA} is false.

11.5 Homework Problems

1. Follow the proof of Fact 11.2 for the case of axioms P1 and C1 to prove the case of axioms P2 and C2.

2. Prove the case of axioms P2, C2 and axioms P23, C3 of the Fact 11.2.

3. Prove Fact 11.2 in case of axioms P5, C5 and axioms P8, C8 of the Fact 11.2.

4. Complete the proof of Fact 11.2 or all cases.

5. We proved that the property $P1'$ of Fact 11.3 is a generalization of axiom $P1$ of PA (11.11, i.e. it is provable in PA.

   (i) Write detailed proofs of properties $P2' - P5'$ in PA.

   (i) Write detailed proofs of properties $P6' - P8'$ in PA.

6. Follow the definition 11.8 and prove the following formulas pre provable in PA for ant terms t, r, s.

   (i) $t \not< t$.

   (ii) $(t < s \Rightarrow (s < r \Rightarrow t < r))$.

   (iii) $(0 < \bar{1})$, $(\bar{1} < \bar{2})$, $(\bar{2} < \bar{3})$, $(\bar{3} < \bar{4})$, $\ldots$.

   (iv) $0 \leq t$.

   (v) $t \leq t$.

   (vi) $(t \leq r \cup r \leq t)$.

   (vii) $(t \leq r \Rightarrow (r \leq t \Rightarrow t = r))$.

7. Follow the definition 11.8 and prove the following formulas pre provable in PA for ant terms t, r, s.

   (i) $(t \leq s \Rightarrow (s \leq r \Rightarrow t \leq r))$,

   (ii) $(t \leq s \Rightarrow (t + r \leq s + r))$,

   (ii) $(r > 0 \Rightarrow (t > 0 \Rightarrow r \cdot t > 0))$.

8. Let RR be the Robinson System (11.23). Show that RR is a proper subtheory of PA by finding a model of RR that is not a model for PA.

9. Let RR be the Robinson System (11.23). Let n, m be any natural numbers. Prove the following holds in RR.

   (i) If $m \neq n$, then $\overline{m} \neq \overline{n}$.

   (ii) $\overline{m+n} = \overline{m} + \overline{n}$ and $\overline{m \cdot mn} = \overline{m} \cdot \overline{n}$ are provable in RR .

   (ii) Any model for RR is infinite.

10. Here us the reasoning we used explaining Hilbert Consistency Program.

    " Let A be a *real* statement $\forall x(f(x) = g(x))$. Assume $\vdash_I A$. Then there is a derivation d of A in I. But, derivations are concrete objects and, for some real formula $P(x, y)$ *encoding* derivations in I, $\vdash_R P(d, \ulcorner A \urcorner)$, where $\ulcorner A \urcorner$ is some *code* for A.
    Now, if A were false, one would have $f(a) \neq g(a)$ for some a and hence $\vdash_R P(c, \ulcorner \neg A \urcorner)$ for some c. In fact, one would have a stronger assertion $\vdash_R (f(x) \neq g(x) \Rightarrow P(c_x, \ulcorner \neg A \urcorner))$. But, if R proves *consistency* of I, we have

    $$\vdash_R \neg(P(d, \ulcorner A \urcorner) \cap P(c, \ulcorner \neg A \urcorner)),$$

    whence $\vdash_R f(x) = g(x)$, with free variable x, i.e. $\vdash_R \forall x(f(x) = g(x))$."

    (i) Write down a detailed proof of correctness of the last part of reasoning:

    "But, if R proves *consistency* of I, we have

    $$\vdash_R \neg(P(d, \ulcorner A \urcorner) \cap P(c, \ulcorner \neg A \urcorner)),$$

    whence $\vdash_R f(x) = g(x)$, with free variable x, i.e. $\vdash_R \forall x(f(x) = g(x))$."

    (ii) List, prove and use proper Logic Properties similar to properties Logic 11.1 - Logic 11.4 in the proof of Theorem 11.7.