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 of both propositional and predicate languages and discuss their semantics.

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 "The 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 do not 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.

The similar examples can be found in natural language rather then in math-

ematical language. For example we tend to accept a statement: "The earth circulates around 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

$$SL = (\mathcal{A},\ \mathcal{F}). \tag{1}$$

a *symbolic language.*

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

Definition 1

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

$$\mathcal{L} = (\mathcal{A}, \mathcal{F}), \tag{2}$$

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 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 in 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

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 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 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* $2 \ngtr 1$ *or* $2 + 2 = 4$.

Symbols for Connectives

The symbols for connectives used in our book are not the only one 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
$-\mathrm{A}$	$(A \cup B)$	$(A \cap B)$	$(A \Rightarrow B)$	$(A \Leftrightarrow B)$
$N\mathrm{A}$	$D\mathrm{AB}$	$C\mathrm{AB}$	$I\mathrm{AB}$	$E\mathrm{AB}$
$\overline{A}$	$(A \vee B)$	(A & B)	$(A \rightarrow B)$	$(A \leftrightarrow B)$
$\sim A$	$(A \vee B)$	$(A \cdot B)$	$(A \supset B)$	$(A \equiv B)$
A'	$(A + B)$	$(A \cdot B)$	$(A \rightarrow B)$	$(A \equiv B)$

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.

1.1 Homework Problems

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 4 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 $\mathcal{L}$.

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

(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

(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

(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,

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

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

Thus the following propositions are **true**:

$$If \quad 6 \quad DIVIDES \quad 2, \quad then \quad 3 \quad DIVIDES \quad 2. \tag{7}$$

$$If \quad 6 \quad DIVIDES \quad 3, \quad then \quad 3 \quad DIVIDES \quad 3. \tag{8}$$

$$If \quad 6 \quad DIVIDES \quad 6, \quad then \quad 3 \quad DIVIDES \quad 6. \tag{9}$$

It follows from (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 (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 (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

(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

(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

(12)

The table (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.

$$\begin{aligned}
\neg T = F, \quad \neg F = T; \\
(T \cap T) = T, \quad (T \cap F) = F, \quad (F \cap T) = F, \quad (F \cap F) = F; \\
(T \cup T) = T, \quad (T \cup F) = T, \quad (F \cup T) = T, \quad (F \cup F) = F; \\
(T \Rightarrow T) = T, \quad (T \Rightarrow F) = F, \quad (F \Rightarrow T) = T, \quad (F \Rightarrow F) = T; \\
(T \Leftrightarrow T) = T, \quad (T \Leftrightarrow F) = F, \quad (F \Leftrightarrow T) = F, \quad (T \Leftrightarrow T) = T.
\end{aligned} \tag{13}$$

We use the above set of equations (13) to evaluate logical values of formulas.

Example 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 (13) as follows.*

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

Exercise 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

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 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 (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

(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 (4), (5), (48), (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

(15)

Functional Dependency

It can be proved that all propositional connectives, as defined by tables (14) and (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 (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 *All two-valued propositional connectives and in particular our connectives* $\neg, \cup, \cap, \Rightarrow$, *and* $\Leftrightarrow$ *are* **a functionally dependent**.

In particular, we prove the following.

Fact 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 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

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

(16)

We now write the table (16) in the "formula independed" form of the following equations.

$$(T \uparrow T) = F,\ (T \uparrow F) = T,\ (F \uparrow T) = T,\ (F \uparrow F) = T \tag{17}$$

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

$$\neg A = (A \uparrow A) \tag{18}$$

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**

(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 (18) is correct.

Example 3

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

Observe now that the Sheffer's connective table (16) looks as a negation of the conjunction table (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

$$(A \cap B) = \neg(A \uparrow B). \tag{20}$$

We have just proved the formula (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 (24) becomes $(A \cap B) = (A \uparrow B) \uparrow (A \uparrow B)$.

We call the equality

$$(A \cap B) = (A \uparrow B) \uparrow (A \uparrow B) \tag{21}$$

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

(22)

We now write the table (22) in an "formula independed" form of the following equations.

$$(T \downarrow T) = F,\ (T \downarrow F) = F,\ (F \downarrow T) = F,\ (F \downarrow F) = T \tag{23}$$

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

$$\neg A = (A \downarrow A) \tag{24}$$

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**

(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 (24) is correct.

Exercise 6

Prove that the equality

$$(A \cup B) = ((A \downarrow B) \downarrow (A \downarrow B)) \tag{26}$$

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

Solution
To prove the correctness of the equation (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**	$((F \downarrow F) \downarrow (F \downarrow F)) = (T \downarrow T) = \mathbf{F}$

(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 (26) is correct.

2.1 Homework Problems

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.

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 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**

(28)

The logical value of the formula $(A \Rightarrow A)$ is always T, what means that it is a **propositional tautology**. The table (28) is called a **truth table** for the formula $(A \Rightarrow A)$.

Example 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**

(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 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 (28). We leave the proofs of others as an easy exercise.

Identity for Implication

$$\models (A \Rightarrow A) \tag{30}$$

Identity for Equivalence

$$\models (A \Leftrightarrow A) \tag{31}$$

Excluded Middle

$$\models (\neg A \cup A) \tag{32}$$

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 B.C) and a rule of inference, based on it is called *Modus Ponens*, so we use the same name here.

Modus Ponens

$$\models ((A \cap (A \Rightarrow B)) \Rightarrow B) \tag{33}$$

Detachment

$$\models ((A \cap (A \Leftrightarrow B)) \Rightarrow B) \tag{34}$$
$$\models ((B \cap (A \Leftrightarrow B)) \Rightarrow A)$$

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 textitcontrapositive implication.

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

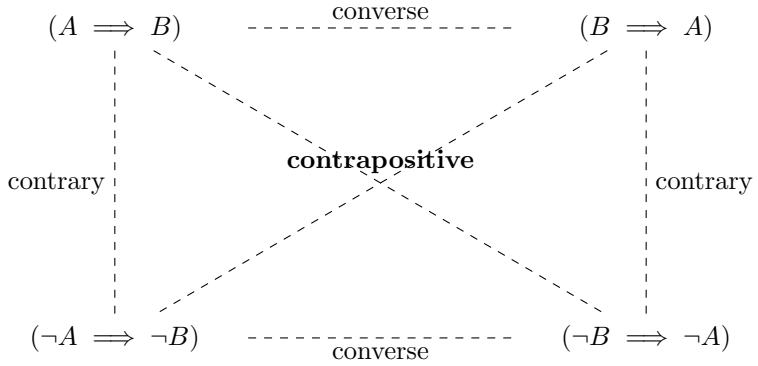
Each of the following pairs of implications: *a simple* and *an opposite*, and *a converse* and *a contrary* are equivalent, i.e. the following formulas are tautologies:

Laws of contraposition (1)

$$\models ((A \Rightarrow B) \Leftrightarrow (\neg B \Rightarrow \neg A)), \tag{35}$$

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

The laws of contraposition (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*.



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:

$$\models ((A \Leftrightarrow B)) \Leftrightarrow ((A \Rightarrow B) \cap (B \Rightarrow A))). \tag{36}$$

The above tautology (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

$$\begin{aligned} &\models (((A \Rightarrow B) \cap (B \Rightarrow C)) \Rightarrow (A \Rightarrow C)) \\ &\models ((A \Rightarrow B) \Rightarrow ((B \Rightarrow C) \Rightarrow (A \Rightarrow C))) \\ &\models ((B \Rightarrow C) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C))). \end{aligned} \tag{37}$$

Modus tollendo ponens

$$\begin{aligned} &\models (((A \cup B) \cap \neg A) \Rightarrow B) \\ &\models (((A \cup B) \cap \neg B) \Rightarrow A) \end{aligned} \tag{38}$$

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

$$\models (\neg A \Rightarrow (A \Rightarrow B)) \tag{39}$$

Clavius

$$\models ((\neg A \Rightarrow A) \Rightarrow A) \tag{40}$$

Frege

$$\models (((A \Rightarrow (B \Rightarrow C)) \cap (A \Rightarrow B)) \Rightarrow (A \Rightarrow C)) \tag{41}$$

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

Double Negation

$$\models (\neg\neg A \Leftrightarrow A) \tag{42}$$

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

$$\models ((\neg A \Rightarrow (B \cap \neg B)) \Rightarrow A) \tag{43}$$

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 (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 (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)

$$\models ((\neg A \Rightarrow B) \Leftrightarrow (\neg B \Rightarrow A)), \tag{44}$$

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

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

$$\models ((A \cap (B \cup C)) \Leftrightarrow ((A \cap B) \cup (A \cap C))), \tag{45}$$

$$\models ((A \cup (B \cap C)) \Leftrightarrow ((A \cup B) \cap (A \cup C))). \tag{46}$$

De Morgan

$$\models (\neg(A \cup B) \Leftrightarrow (\neg A \cap \neg B)), \quad \models (\neg(A \cap B) \Leftrightarrow (\neg A \cup \neg B)). \tag{47}$$

Implication

$$\models ((A \Rightarrow B) \Leftrightarrow (\neg A \cup B)). \tag{48}$$

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

$$\models (\neg\neg A \Leftrightarrow A). \tag{49}$$

Exportation and Importation

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

De Morgan laws (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.

3.1 Homework Problems

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 $b \neq 0$.

4 Predicate Language: Description and Applications to Artificial Intelligence

We define a *predicate language* $\mathcal{L}$ following the pattern established by the symbolic and propositional languages definitions (1), (2). 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 4

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

$$\mathcal{L} = (\mathcal{A}, \mathbf{T}, \mathcal{F}), \tag{50}$$

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 build 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 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.* $t_1, t_2, ..., t_n \in \mathbf{T}$.

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 6

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

1. *All* **atomic formulas** *(definition 5) 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 $\forall x A,\ \exists x A$ are formulas for any variable $x \in VAR$.*

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

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

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

Let's look now closer at the following formulas.

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

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 $\exists x R(x, y)$, 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 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 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 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.

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 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

$$\forall_{A(x)}\ B(x) \equiv \forall x(A(x) \Rightarrow B(x)). \tag{51}$$

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

$$\exists_{A(x)}\ B(x) \equiv \exists x(A(x) \cap B(x)) \tag{52}$$

The definition 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 (51) and (52) **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 a 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 (51) and (52) 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 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 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 $N(x), Z(x)$ 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 the exercise 8. We presented the long solution in order to explain all steps needed to be performed when one writes a the short solution.

Example 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 7 and transformation rules (51), (52) 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 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 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 se language to our 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 (51) and (52) to write $A \in \mathcal{F}$ of the predicate language $\mathcal{L}$.

Example 8

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*

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

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 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 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, a 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 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 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 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 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*.

4.1 Homework Problems

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 drunk only if they are drink too much.

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

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 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 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 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 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. $A_{1I} : x \leq y$ 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. $A_{2I} : \forall_{y \in N}\ x \leq y$ and $A_2 : \forall y R(x, y)$ is *satisfied* in $\mathbf{M} = (N, I)$ only by the natural number 0.
3. $A_{3I} : \exists_{x \in N} \forall_{y \in N}\ x \leq y$ 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 $A_{3I} : \exists_{x \in Z} \forall_{y \in Z}\ x \leq y$ 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 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 8 of a following definition of a notion *" A is not a predicate tautology".*

Definition 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 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:

$$\models_p \ (\forall x \ A(x) \Rightarrow \exists x \ A(x)). \tag{53}$$

Observe that (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 (53) is not a predicate tautology.

$$\not\models_p \ (\exists x \ A(x) \Rightarrow \forall x \ A(x)) \tag{54}$$

To prove (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 (53) fails when we use the quantifiers with restricted domain.

Example 15

Show that

$$\not\models_p \ (\forall_{B(x)} \ A(x) \Rightarrow \exists_{B(x)} \ A(x)). \tag{55}$$

Observe that (55) means that corresponding proper formula F of $\mathcal{L}$ obtained by the restricted quantifiers *transformations rules* (51), (52) is not a predicate tautology, i.e.

$$\not\models_p \ (\forall x(B(x) \Rightarrow A(x)) \Rightarrow \exists x(B(x) \cap A(x))). \tag{56}$$

We construct a *counter-model* $\mathbf{M}$ for (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 (56) becomes a mathematical statement

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

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 (53) is:

$$\models_p \ (\forall_{B(x)} \ A(x) \Rightarrow (\exists x \ B(x) \Rightarrow \exists_{B(x)} \ A(x))). \tag{57}$$

We remind that (57) means that corresponding proper formula of $\mathcal{L}$ obtained by the restricted quantifiers *transformations rules* (51), (52) is a predicate tautology, i.e.

$$\models_p (\forall x(B(x) \Rightarrow A(x)) \Rightarrow (\exists x \ B(x) \Rightarrow \exists x \ (B(x) \cap A(x)))) \tag{58}$$

.

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

$$\models_p \ (\forall x \ A(x) \Rightarrow A(y)), \tag{59}$$

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 (59) is:

$$\models_p \ (\forall_{B(x)} \ A(x) \Rightarrow (B(y) \Rightarrow A(y))), \tag{60}$$

where $y \in VAR$ satisfies the same condition as in (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* (51), (52) is a predicate tautology.

A more general version of (59) is:

$$\models_p \ (\forall x \ A(x) \Rightarrow A(t)), \tag{61}$$

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.*

$$\models_p \ (A(x) \Rightarrow \forall x \ A(x)). \tag{62}$$

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.

$$\models_p \ (\exists x \ (A(x) \cap B(x)) \ \Rightarrow \ (\exists x A(x) \cap \exists x B(x))) \tag{63}$$

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

$$\not\models_p \ ((\exists x A(x) \cap \exists x B(x)) \Rightarrow \exists x \ (A(x) \cap B(x))). \tag{64}$$

To prove (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.

$$\models_p \ ((\forall x A(x) \cup \forall x B(x)) \Rightarrow \forall x \ (A(x) \cup B(x))). \tag{65}$$

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

$$\not\models_p \ (\forall x \ (A(x) \cup B(x)) \Rightarrow (\forall x A(x) \cup \forall x B(x))). \tag{66}$$

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 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 1) and propositional tautology (definition 3).

Equational Laws for Quantifiers

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

$$\neg \forall x A(x) \equiv \exists x \neg A(x), \qquad \neg \exists x A(x) \equiv \forall x \neg A(x). \tag{67}$$

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

$$\forall x A(x) \equiv \neg \exists x \neg A(x), \quad \exists x A(x) \equiv \neg \forall x \neg A(x). \tag{68}$$

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.

$$\forall x A(x) \equiv \forall y A(y), \qquad \exists x A(x) \equiv \exists y A(y). \tag{69}$$

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

$$\forall x \forall y \ (A(x, y) \ \equiv \ \forall y \forall x \ (A(x, y) \tag{70}$$

$$\exists x \exists y \ (A(x, y) \ \equiv \ \exists y \exists x \ (A(x, y) \tag{71}$$

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.

$$\forall x(A(x) \cup B) \equiv (\forall x A(x) \cup B), \tag{72}$$

$$\exists x(A(x) \cup B) \equiv (\exists x A(x) \cup B), \tag{73}$$

$$\forall x(A(x) \cap B) \equiv (\forall x A(x) \cap B), \tag{74}$$

$$\exists x(A(x) \cap B) \equiv (\exists x A(x) \cap B), \tag{75}$$

$$\forall x(A(x) \Rightarrow B) \equiv (\exists x A(x) \Rightarrow B), \tag{76}$$

$$\exists x(A(x) \Rightarrow B) \equiv (\forall x A(x) \Rightarrow B), \tag{77}$$

$$\forall x(B \Rightarrow A(x)) \equiv (B \Rightarrow \forall x A(x)), \tag{78}$$

$$\exists x(B \Rightarrow A(x)) \equiv (B \Rightarrow \exists x A(x)). \tag{79}$$

Distributivity Laws

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

$$\forall x \ (A(x) \cap B(x)) \ \equiv \ (\forall x A(x) \cap \forall x B(x)) \tag{80}$$

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

$$\exists x \ (A(x) \cup B(x)) \ \equiv \ (\exists x A(x) \cup \exists x B(x)) \tag{81}$$

We also define the notion of logical equivalence $\equiv$ for the formulas of the —textitpropositional language (definition 1) and its semantics.

Definition 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 (48), (49) into the following predicate equivalences.

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

$$(A \Rightarrow B) \equiv (\neg A \cup B), \tag{82}$$

$$\neg\neg A \equiv A \tag{83}$$

We use the 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,

$$\neg\forall_{B(x)}\ A(x) \equiv \exists_{B(x)}\ \neg A(x), \qquad \neg\exists_{B(x)}\ A(x) \equiv \forall_{B(x)}\neg A(x). \tag{84}$$

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 (45), (46) into the following predicate equivalences.

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

$$(A \cap (B \cup C)) \equiv ((A \cap B) \cup (A \cap C)), \tag{85}$$

$$(A \cup (B \cap C)) \equiv ((A \cup B) \cap (A \cup C)) \tag{86}$$

We use the to prove the following Distributivity Laws for *restricted quantifiers*.

Restricted Distributivity Laws
We generalize the *Introduction and Elimination Laws* (72), (75), (76), (78) 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 $A(x), B(x), C(x)$.

$$\forall_{C(x)}(A(x) \cup B) \equiv (\forall_{C(x)}A(x) \cup B), \tag{87}$$

$$\exists_{C(x)}\ (A(x) \cap B) \equiv (\exists_{C(x)}\ A(x) \cap B), \tag{88}$$

$$\forall_{C(x)}(A(x) \Rightarrow B) \equiv (\exists_{C(x)}A(x) \Rightarrow B), \tag{89}$$

$$\forall_{C(x)}(B \Rightarrow A(x)) \equiv (B \Rightarrow \forall_{C(x)}A(x)). \tag{90}$$

The proofs are similar to the proof of the restricted de Morgan Laws.
The similar generalization of the other *Introduction and Elimination Laws* (73), (74), (77), (79) fails. We can easily follow Example15 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 (73) and leave the other cases to the reader as an exercise.

Example 16

The restricted quantifiers version of (73) is the following.

$$\exists_{C(x)}(A(x) \cup B) \equiv (\exists_{C(x)}A(x) \cup (\exists x\ C(x) \cap B)). \tag{91}$$

We derive (91) 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.

5.1 Homework Problems

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 $\mathbf{M}$ 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 $\mathbf{M}$ and corresponding closed formulas determine for each of them whether $\mathbf{M}$ 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) $\forall_{C(x)}(B \Rightarrow A(x)) \equiv (B \Rightarrow \forall_{C(x)} A(x))$