LOGICS FOR COMPUTER SCIENCE: Classical and Non-Classical Springer 2019

Anita Wasilewska

Chapter 4
General Proof Systems: Syntax and Semantics

CHAPTER 4 SLIDES

Chapter 4
General Proof Systems: Syntax and Semantics

Slides Set 1

PART 1 Introduction

PART 2 **Syntax:** Definition of Proof System, Formal Proofs

PART 3 Syntactic Decidability

PART 4 **Consequence Operation**, Non Monotonic Reasoning and Syntactic Consistency

Slides Set 2

PART 5 **Semantics:** Soundness and Completeness

PART 6 Exercises and Examples

Chapter 4
General Proof Systems: Syntax and Semantics

Slides Set 1

PART 1 **Introduction**

Chapter 4
General Proof Systems: Introduction

Proof systems are built to prove, it means to construct **formal proofs** of statements formulated in a given language

First component of any proof system is hence its formal **language** $\mathcal{L}$

Proof systems are **inference** machines with statements called **provable** statements being their final products

Chapter 4
General Proof Systems: Axioms

The **starting** points of the inference machine of a proof system S are called its **axioms**

We distinguish two kinds of axioms: **logical** axioms LA and **specific** axioms SA

Semantical link: we usually build a proof systems for a given language and its **semantics** i.e. for a **logic** defined semantically

General Proof Systems: Logical Axioms

We choose as a set of **logical** axioms LA some subset of **tautologies**, under a given semantics

We will **consider** here only proof systems with **finite** sets of logical or specific axioms, i.e we will examine only **finitely axiomatizable** proof systems

General Proof Systems: Logical Axioms

We can, and we often do, consider proof systems with languages **without** yet established **semantics**

In this case the **logical** axioms LA serve as description of **tautologies** under a future **semantics** yet to be built

Logical axioms LA of a proof system S are hence not only tautologies under an established semantics, but they can also guide us how to **define** a semantics when it is yet **unknown**

General Proof Systems: Specific Axioms

The **specific axioms** SA consist of statements that describe a specific **knowledge** of an universe we want to use the proof system S to **prove** facts about

Specific axioms SA **are not** universally true

Specific axioms SA are true only in the universe we are interested to **describe** and **investigate** by the use of the proof system S

General Proof Systems: Formal Theory

Given a **proof system** S with **logical** axioms LA

We choose as **specific axioms** SA of the proof system S any **finite set** of formulas that are not tautologies, and hence the **specific axioms** SA are always disjoint with the set LA of **logical** axioms LA of S

The **proof system** S with added set of **specific** axioms SA is called a **formal theory** based on S

General Proof Systems: Inference Machine

The **inference** machine of a proof system S is defined by a **finite** set of inference rules

The **inference rules** describe the way we are allowed to **transform** the information **within** the proof system S with the **logical** axioms LA as a **starting** point

We depict it informally on the next slide

General Proof Systems: Inference Machine

AXIOMS

↓ ↓ ↓

RULES applied to AXIOMS

↓ ↓ ↓

RULES applied to any expressions above

↓ ↓ ↓

Provable formulas

General Proof Systems: Semantical Link

Rules of inference of a system S have to **preserve** the truthfulness of what they are being used to prove

The notion of **truthfulness** is always defined by a given semantics **M**

Rules of inference that **preserve** the truthfulness are called **sound rules** under a given a semantics **M**

Rules of inference can be **sound** under one semantics and **not sound** under another

General Proof Systems: Soundness Theorem

Goal 1

When developing a proof system S the first goal is to **prove** the following **theorem** about it and its semantics **M**

Soundness Theorem

For any formula A of the language of the system S

If a formula A is **provable** from **logical** axioms LA of S only, then A is a **tautology** under the semantics **M**

General Proof Systems: Soundness Theorem

By definition, the notion of **soundness** is connected with a given semantics

A proof system S can be sound under **one semantics** and not sound under the **other**

For **example** a set of axioms and rules sound under the **classical semantics** might not be sound under **L** semantics, or K semantics, or others

General Proof Systems: Completeness Property

Denote by $\mathbf{T_M}$ the set of all tautologies defined by the semantics $\mathbf{M}$, i.e.

$$\mathbf{T_M} = \{A \in \mathcal{F} : \ \models_{\mathbf{M}} A\}$$

A natural **question** arises:
are all tautologies i.e formulas $A \in \mathbf{T_M}$ **provable** in the proof system S ??

The positive answer to this question is called **completeness property** of the system S

General Proof Systems: Completeness Theorem

Goal 2

Given for a **sound** proof system S under the semantics **M**, our second goal is to **prove** the following theorem about S

Completeness Theorem

For any formula A of the language of S

A **is provable** in S if and only if A is a **tautology** under the semantics **M**

We write the **Completeness Theorem** symbolically as

$$\vdash_S A \quad \text{if and only if} \quad \models_{\mathbf{M}} A$$

Proving Soundness and Completeness

The **Completeness Theorem** is composed of two parts

The soundness part, i.e. the **Soundness Theorem** and the completeness part that proves the **completeness property** of already sound proof system

Proving the **Soundness Theorem** for S under a semantics **M** is usually a straightforward and not a very difficult task

We **first** prove that all **logical axioms** LA are **tautologies** under the given semantics and then we **prove** that all **inference rules** of the system S **preserve** the notion of the **truth** under it

Proving Soundness and Completeness

Proving the completeness part of the **Completeness Theorem** is always the crucial, difficult and sometimes impossible task

We study two proofs of the **Completeness Theorem** for **classical propositional** proof system in Chapter 5

We present a constructive proofs of the **Completeness Theorem** for different Gentzen style **automated** theorem proving systems for **classical** semantics in Chapter 6

We discuss the Inuitionistic and Modal Logics in Chapter 7
The **Predicate** Logics are discussed Chapters 8, 9, 10, 11

Chapter 4
General Proof Systems: Syntax and Semantics

Slides Set 1

PART 2 **Syntax** : Definition of Proof System, Formal Proofs

Syntax : Definition of Proof System

When **defining** a proof system S we **specify**, as the first step,
its formal language $\mathcal{L}$

This is a **first component** of the proof system S

Given a set $\mathcal{F}$ of well formed **formulas** of the language $\mathcal{L}$,
we often **extend** this set, and hence the language $\mathcal{L}$ to
a set $\mathcal{E}$ of **expressions** build out of the language $\mathcal{L}$ and
some additional symbols, if needed

It is a **second component** of the proof system S

Syntax : Definition of Proof System

Proof systems act as an inference machine, with **provable** expressions being its **final** products

This inference machine is **defined** by setting, as a **starting point** a certain non-empty, proper subset LA of $\mathcal{E}$, called a set of **logical axioms** of the system S

The production of **provable** statements is to be done by the means of **inference rules**

The inference rules transform an expression, or finite string of expressions, called **premisses**, into another expression, called a **conclusion**

Syntax : Definition of Proof System

At this stage the inference rules don't carry any **meaning**
They only **define** how to transform strings of **symbols** of a language into another string of **symbols**

This is a reason why investigation of proof systems is called **syntax** or **syntactic** investigation as opposed to **semantical** methods

The **syntax- semantics** connection within proof systems is established by **Soundness** and **Completeness** theorems and is discussed in detail in the **Slides Set 2**

Syntax : Definition of Proof System

Definition

By a **proof system** we understand a quadruple

$$S = (\mathcal{L}, \mathcal{E}, LA, \mathcal{R})$$

where

$\mathcal{L} = \{\mathcal{A}, \mathcal{F}\}$ is a **language** of S with a set $\mathcal{F}$ of formulas

$\mathcal{E}$ is a set of **expressions** of S

In particular case $\mathcal{E} = \mathcal{F}$

$LA \subseteq \mathcal{E}$ is a **non- empty, finite set** of logical axioms of S

$\mathcal{R}$ is a **non- empty, finite set** of rules of inference of S

Proof System Components: Language

Language of S is any formal language

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

We assume as before that both sets $\mathcal{A}$ and $\mathcal{F}$ are enumerable, i.e. we deal here with enumerable languages

The language $\mathcal{L}$ can be **propositional** or **first order (predicate)** but we discuss propositional languages first

Proof System Components: Expressions

Expressions $\mathcal{E}$ of S

Given a set $\mathcal{F}$ of **formulas** of the language $\mathcal{L}$ of S

We often extend the set $\mathcal{F}$ to some set $\mathcal{E}$ of **expressions** build out of the symbols of $\mathcal{L}$ and some extra symbols, if needed

In this case all other components of S are also defined on basis of elements of the set of **expressions** $\mathcal{E}$

In particular, and **most common case** we have that $\mathcal{E} = \mathcal{F}$

Expressions Examples

Automated theorem proving **systems** usually use as their basic components special sets of **expressions** build out of formulas of

$\mathcal{L}$

In Chapters 6 , 10 we consider **finite sequences** of formulas as basic expressions of proof systems **RS** and **RQ**
We also present there proof systems that use yet other kind of expressions, called **Gentzen sequents** or their modifications

Some systems also use other expressions such as **clauses**, **sets of clauses**, or **sets of formulas**

Proof System Components: Logical Axioms

Logical axioms LA of S

We distinguish a non-empty subset LA of the set $\mathcal{E}$ of expressions of S as a set of **logical axioms**, i.e.

$$LA \subseteq \mathcal{E}$$

In particular, LA is a non-empty subset of **formulas**, i.e.

$$LA \subseteq \mathcal{F}$$

We **assume** that one can effectively decide, for any $E \in \mathcal{E}$ whether $E \in LA$ or $E \notin LA$

We also **assume** that the set LA is always **finite**, i.e. that we consider here **finitely** axiomatizable proof systems

Proof System Components: Rules of Inference

Rules of inference $\mathcal{R}$ of S

We assume that S contains only a **finite** number of **inference rules**

We **assume** that each rule has a finite number of **premisses** and one **conclusion**

We also **assume** that one can effectively decide, for any **inference rule**, whether given strings of expressions **form** its premisses and conclusion or they **do not**

Proof System Components: Rules of Inference

Definition

Each **rule of inference** $r \in \mathcal{R}$ is a **relation** defined in the set $\mathcal{E}^m$, where $m \geq 1$ with values in $\mathcal{E}$, i.e.

$$r \subseteq \mathcal{E}^m \times \mathcal{E}$$

Elements $P_1, P_2, \ldots P_m$ of a tuple $(P_1, P_2, \ldots P_m, C) \in r$ are called **premisses** of the rule r and C is called its **conclusion**

Proof System Components: Rules of Inference

We write the **inference rules** in a following convenient way

One premiss rule

$$(r) \quad \frac{P_1}{C}$$

Two premisses rule

$$(r) \quad \frac{P_1 \;;\; P_2}{C}$$

m premisses rule

$$(r) \quad \frac{P_1 \;;\; P_2 \;;\; .... \;;\; P_m}{C}$$

Syntax: Formal Proofs

A **final** product of a single or multiple use of the **inference rules** of S, with axioms taken as a **starting** point are called **provable** expressions of the proof system S

A single use of an inference rule is called a **direct consequence**

A multiple application of rules of inference with axioms taken as a starting point is called a **proof**

Syntax: Direct Consequence

Formal **definitions** are as follows

Direct consequence

For any rule of inference $r \in \mathcal{R}$ of the form

$$(r) \qquad \frac{P_1 \; ; \; P_2 \; ; \; .... \; ; \; P_m}{C}$$

C is called a **direct consequence** of $P_1, ...P_m$ by virtue of the rule $r \in \mathcal{R}$

Syntax: Formal Proof Definition

Formal Proof of an expression $E \in \mathcal{E}$ in a proof system

$$S = (\mathcal{L}, \mathcal{E}, LA, \mathcal{R})$$

is a sequence

$$A_1, A_2, , A_n \quad \text{for} \quad n \geq 1$$

of expressions from $\mathcal{E}$, such that

$$A_1 \in LA, \qquad A_n = E$$

and for each $1 < i \leq n$, either $A_i \in LA$ or A_i is a **direct consequence** of some of the **preceding** expressions by virtue of one of the rules of inference

$n \geq 1$ is the **length** of the proof $A_1, A_2, , A_n$

Syntax: Formal Proof Notation

We write

$$\vdash_S E$$

to denote that $E \in \mathcal{E}$ **has a proof** in S and we call E a **provable** expression of S

The set of all **provable** expressions of S is denoted by $\mathbf{P}_S$, i.e. we put

$$\mathbf{P}_S = \{E \in \mathcal{E} : \ \vdash_S E\}$$

When the proof system S is **fixed** we write $\vdash E$

Simple System S_1

Example

Consider a very simple proof system system S_1 with $\mathcal{E} = \mathcal{F}$

$$S_1 = (\mathcal{L}_{\{P,\ \Rightarrow\}},\ \mathcal{F},\ LA = \{(A \Rightarrow A)\},\ \mathcal{R} = \{(r)\ \ \frac{B}{PB}\})$$

where $A,\ B \in \mathcal{F}$ are any formulas and where P is some one argument connective

We might read PA for example as " it is possible that A"

Observe that even the system S_1 has only one axiom, it represents an infinite number of formulas

We call such axiom an **axiom schema**

Simple System S_2

Example

Consider now a system S_2

$$S_2 = (\mathcal{L}_{\{P,\Rightarrow\}}, \quad \mathcal{F}, \quad \{(a \Rightarrow a)\}, \quad (r)\ \frac{B}{PB}\ ),$$

where $a \in VAR$ is any variable (atomic formula) and $B \in \mathcal{F}$ is any formula

Observe that the system S_2 also has only one axiom similar to the axiom of S_1 and they have the same rule of inference but they are **different proof systems** as

for example a formula

$$((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c)))$$

is an axiom of system S_1 but **is not** an axiom of S_2

Example

We have that

$$\vdash_{S_1} ((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c)))$$

because $((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c))) \in \mathrm{LA}$

Some other provable formulas are

$$\vdash_{S_1} P(a \Rightarrow a), \quad \vdash_{S_1} PP(a \Rightarrow a), \quad \vdash_{S_2} PP(a \Rightarrow a)$$

Formal Proofs

Formal proof of $P(a \Rightarrow a)$ in S_1 and S_2 is:

$A_1 = (a \Rightarrow a)$, axiom

$A_2 = P(a \Rightarrow a)$ rule application for $B = (a \Rightarrow a)$

Formal proof of $PP(a \Rightarrow a)$ in S_1 and S_2 is:

$A_1 = (a \Rightarrow a)$, axiom

$A_2 = P(a \Rightarrow a)$, rule application for $B = (a \Rightarrow a)$

$A_3 = PP(a \Rightarrow a)$ rule application for $B = P(a \Rightarrow a)$

Exercise

Given a proof system:

$$S = (\mathcal{L}_{\{\neg,\Rightarrow\}},\ \mathcal{F},\ \{(A \Rightarrow A), (A \Rightarrow (\neg A \Rightarrow B))\},\ \mathcal{R} = \{(r)\})$$

where $(r)\ \dfrac{(A \Rightarrow B)}{(B \Rightarrow (A \Rightarrow B))}$

Write a **formal proof** in S with 2 applications of the rule (r)

Solution: There are many solutions. Here is one of them.

Required formal proof is a sequence A_1, A_2, A_3, where

$A_1 = (A \Rightarrow A)$
(Axiom)

$A_2 = (A \Rightarrow (A \Rightarrow A))$
Rule (r) application 1 for $A = A,\ B = A$

$A_3 = ((A \Rightarrow A) \Rightarrow (A \Rightarrow (A \Rightarrow A)))$
Rule (r) application 2 for $A = A, B = (A \Rightarrow A)$

Simple System S_3

Consider a very simple proof system system S_3 defined as follows

$$S_3 = (\ \mathcal{L}_{\{P,\Rightarrow\}},\ \mathcal{F},\ \{(A \Rightarrow A)\},\quad (r_1)\ \frac{B}{PB},\ (r_2)\ \frac{A\ ;\ B}{P(A \Rightarrow B)}\ )$$

Exercise

Write two **formal proofs** in S_3 both of the lengths 4, one of which must contain at least one application of the inference rule r_2

Chapter 4
General Proof Systems: Syntax and Semantics

Slides Set 1

PART 3 **Syntactic Decidability**,

Automated Proof Systems

General Proof Systems: Syntactic Decidability

For any a proof system $S = (\mathcal{L}, \mathcal{E}, LA, \mathcal{R})$, we **assumed** that its sets LA of its logical axioms and $\mathcal{R}$ of rules of inference have the following **properties**

(LP) For any $E \in \mathcal{E}$ one can **effectively decide** whether $E \in LA$ or $E \notin LA$

(RP) For any infrence rule $r \in \mathcal{R}$ one can effectively decide whether a given strings of expressions **form** its premisses and conclusion or they **do not**

Observe that even if the set of axioms and the inference rules of a **proof system** S have the properties **(LP)** and **(RP)** it **does not** mean that a statement " E is provable " in S can be similarly effectively decided for every proof system

Decidable Proof Systems

Definition

A proof system $S = (\mathcal{L}, \mathcal{E}, LA, \mathcal{R})$ for which there is an **effective** decision procedure for determining for any expression $E \in \mathcal{E}$, whether there is or there is no proof of E in S is called a **decidable** proof system, otherwise S is called **undecidable**

Observe that the above notion of **decidability** of S does not require to find a proof of an expression $E \in \mathcal{E}$ (if exists)

We hence introduce a following notion

Syntactically Decidable Proof Systems

Definition

A proof system $S = (\mathcal{L}, \mathcal{E}, LA, \mathcal{R})$ for which there is an effective mechanical procedure that finds (generates) a formal proof of any expression $E \in \mathcal{E}$, if it exists, is called **syntactically semi- decidable**

If additionally there is an effective method of deciding that if a proof of E is not found that it does not exist, the system S is called **syntactically decidable**

Otherwise S is **syntactically undecidable**

Hilbert Program

The need for **existence** of proof systems for classical logic and parts of mathematics that are **syntactically decidable** or **syntactically semi-decidable** was stated (in a different form) by German mathematician David Hilbert in early 1900 as a part of what is called Hilbert program

The **main goal** of **Hilbert's program** was to provide secure **foundations** for all mathematics

In particular the Hilbert program addressed the problem of **decidability**

It stated that there should be an algorithm for **deciding** the truth or to falsify of any mathematical statement

Moreover, it should use only **"finitistic"** reasoning methods

Syntactically Decidable Proof Systems

Kurt Gdel **proved** in 1931 that most of the **goals** of Hilbert's program were **impossible** to achieve, at least if interpreted in the most obvious way

Nevertheless, Gerhard Gentzen in his work published in 1934/1935 gave a **positive** answer to the possibility of existence of **syntactical decidability**

He invented proof systems for classical and intiutionistic logics, now called **Gentzen style formalizations**

We study the Gentzen style formalizations in chapter 6 and chapters 7, 10

Automated Proof Systems

Gentzen work formed a basis for development of **Automated Theorem Proving** field of mathematics and computer science

Definition

A proof system $S = (\mathcal{L}, \mathcal{E}, LA, \mathcal{R})$ that is **proven** to be syntactically decidable or syntactically semi-decidable is called an **automated proof system**

Automated proof systems are also called **automated** theorem proving systems, **Gentzen style formalizations** and and we use all of these terms interchangeably

Example

Example

Any complete Hilbert style proof system for classical propositional logic is an example of a **decidable** , but

not syntactically decidable proof system

We conclude its **decidability** from the **Completeness Theorem** proved in chapter 5 and the **decidability** of the notion of classical tautology proved in chapter 3

Gentzen style proof systems for classical and intuiionistic **propositional logics** presented in chapters 6,7

are **examples** of proof systems that are of both **decidable** and **syntactically decidable**

Example: Simple System S

Consider now a simple proof system S

$$S = (\mathcal{L}_{\{P, \Rightarrow\}}, \mathcal{F} \ \ LA = \{(a \Rightarrow a)\}, (r)\ \frac{B}{PB})$$

where $a \in VAR$ is any variable (atomic formula) and $B \in \mathcal{F}$ is any formula

Let's **search for a proof** (if exists) of the following formula A

$$A = PP((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c)))$$

Observe, that if A had the proof, the only **last step** in this proof would be the application of the rule

$$(r)\ \frac{B}{PB})$$

to the formula

$$P((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c)))$$

Example: Simple System S

Lets now consider the formula

$$P((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c)))$$

This formula, in turn, if it had the proof, the **only** last step in its proof would be the application of the

$$(r)\ \frac{B}{PB})$$

to the formula

$$((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c)))$$

The **search process stops** here

Proof Search in System S

Observe that the final formula obtained **is not** an axiom of S, i.e.

$$((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c))) \notin LA$$

This means that our **search** for a proof of A in S has **found** sequence of formulas that **does not** constitute a **proof**

This alone does not yet **prove** that the proof **does not** exist

Fortunately, the **search** was at each step unique, so in fact, we **did prove** that the proof of A in S **does not exist**, i.e. we **proved**

$$\nvdash_S \quad PP((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c)))$$

Proof Search Procedure

We easily **generalize** above example to a proof search procedure to **any** formula A of S as follows

Procedure SP

Step: Check the **main** connective of A

If **main** connective is P (it means that A was obtained by the rule (r))

Erase the **main** connective P

Repeat until no P as a **main** connective is left.

If the main connective is $\Rightarrow$ check if a formula is an axiom

If it **is** an axiom, **stop** and **yes** we have a proof

If it is **not** an axiom, **stop** and **no**, proof **does not** exist

Syntactical Decidability of S

The **Procedure SP** is a finite, effective, automatic procedure of **searching** for proofs of formulas in S

Moreover we proved that it **determines** for any formula $A \in \mathcal{F}$, whether there is or there is no proof of A in S

It means that we proved the following.

Fact

The proof system

$$S = (\mathcal{L}_{\{P,\Rightarrow\}}, \mathcal{F}\ \ LA = \{(a \Rightarrow a)\}, (r)\ \frac{B}{PB})$$

where $a \in VAR$ and $B \in \mathcal{F}$

is **syntactically decidable**

Chapter 4
General Proof Systems: Syntax and Semantics

Slides Set 1

PART 4 Consequence Operation, Non Monotonic Reasoning and Syntactic Consistency

Proof from Hypothesis

Given a proof system $S = (\mathcal{L}, \mathcal{E}, LA, \mathcal{R})$

While proving expressions in S we often use some extra information available, besides the axioms of the proof system

This extra information is called **hypotheses** in the proof

A proof from the set of **hypotheses** Γ of an expression E in S is a **formal proof** in S, where the expressions from Γ are treated as additional information added to the set LA of the logical axioms of S

We define it formally as follows

Proof from Hypothesis

Definition

Given a proof system $S = (\mathcal{L}, \mathcal{E}, LA, \mathcal{R})$

Let $\Gamma \subseteq \mathcal{E}$

A **proof** of an expression E from Γ is a sequence

$$E_1, E_2, \ldots E_n$$

of expressions, such that

$$E_1 \in LA \cup \Gamma, \qquad E_n = E$$

and for each $1 < i \leq n$, either $E_i \in LA \cup \Gamma$ or

E_i is a **direct** consequence of some of the **preceding** expressions in the sequence $E_1, E_2, \ldots E_n$ by virtue of one of the **rules** of inference from $\mathcal{R}$.

Proof from Hypothesis

We write

$$\Gamma \vdash_S E$$

to denote that E has a **proof** from Γ in S and

$$\Gamma \vdash E$$

when the system S is fixed

When the set of **hypothesis** Γ is a **finite set** and $\Gamma = \{B_1, B_2, ..., B_n\}$, then we write

$$B_1, B_2, ..., B_n \vdash_S E$$

instead of

$$\{B_1, B_2, ..., B_n\} \vdash_S E$$

Conequences

The case of $\Gamma = \emptyset$ means that in the proof of E only logical axioms LA were used we write

$$\vdash_S E$$

to denote that E has a proof from the empty set Γ

Definition

For any $\Gamma \subseteq \mathcal{E}$, and $A \in \mathcal{E}$,

If $\Gamma \vdash_S A$, then A is called a **consequence** of Γ in S

Definition

We denote by $\mathbf{Cn}_S(\Gamma)$ the **set of all consequences** of Γ in S, i.e. we put

$$\mathbf{Cn}_S(\Gamma) = \{A \in \mathcal{E} : \quad \Gamma \vdash_S A\}$$

.

Consequence Operation

When talking about **consequences** of Γ in S, we define in fact a **function** which to every set $\Gamma \subseteq \mathcal{E}$ assigns a set of all its **consequences**

We denote this function by $\mathbf{Cn}_S$ and adopt the following definition

Definition

Given a proof system $S = (\mathcal{L}, \mathcal{E}, LA, \mathcal{R})$

Any function

$$\mathbf{Cn}_S : 2^{\mathcal{E}} \longrightarrow 2^{\mathcal{E}}$$

such that for every $\Gamma \in 2^{\mathcal{E}}$,

$$\mathbf{Cn}_S(\Gamma) = \{E \in \mathcal{E} : \Gamma \vdash_S E\}$$

is called a **consequence** determined by S

Consequence Operation: Monotonicity

Take any **consequence operation**

$$\mathbf{Cn}_S : 2^{\mathcal{E}} \longrightarrow 2^{\mathcal{E}}$$

Monotonicity Property

For any sets Γ, Δ of expressions of S,

if $\Gamma \subseteq \Delta$ **then** $\mathbf{Cn}_S(\Gamma) \subseteq \mathbf{Cn}_S(\Delta)$

Exercise: write the proof;

it follows directly from the definition of $\mathbf{Cn}_S$ and definition of the formal proof

Consequence Operation: Transitivity

Take any **consequence operation**

$$\mathbf{Cn}_S : 2^{\mathcal{E}} \longrightarrow 2^{\mathcal{E}}$$

Transitivity Property

For any sets $\Gamma_1, \Gamma_2, \Gamma_3$ of expressions of S,

if $\Gamma_1 \subseteq \mathbf{Cn}_S(\Gamma_2)$ and $\Gamma_2 \subseteq \mathbf{Cn}_S(\Gamma_3)$, **then** $\Gamma_1 \subseteq \mathbf{Cn}_S(\Gamma_3)$

Exercise: write the proof;

it follows directly from the definition of $\mathbf{Cn}_S$ and definition of the formal proof

Consequence Operation: Finiteness

Take any **consequence operation**

$$\mathbf{Cn}_S : 2^{\mathcal{E}} \longrightarrow 2^{\mathcal{E}}$$

Finiteness Property

For any expression $A \in \mathcal{E}$ and any set $\Gamma \subseteq \mathcal{E}$,
$A \in \mathbf{Cn}_S(\Gamma)$ if and only if there is a **finite subset** Γ_0 of Γ such that $A \in \mathbf{Cn}_S(\Gamma_0)$

Exercise: write the proof;

it follows directly from the definition of $\mathbf{Cn}_S$ and definition of the formal proof

Tarski Consequence Operation

The notions of **provability** from a set Γ in S and **consequence** determined by S **coincide**

We **use** both terms **interchangeably**, but the definition does do more then just re-naming provability by consequence

We **prove** that the consequence $\mathbf{Cn}_S$ determined by S is a special case of a notion a classic **consequence** operation as defined by Alfred Tarski in 1930 as a general **model** of deductive reasoning

Tarski definition is a **formalization** of the intuitive concept of deduction as a **consequence**, and therefore it has all the **properties** which our intuition attribute to this notion

Tarski Consequence Operation

Definition Tarski, 1930

By a **consequence operation** in a formal language $\mathcal{L} = (\mathcal{A}, \mathcal{F})$ we understand any mapping

$$\mathbf{C} : 2^{\mathcal{F}} \longrightarrow 2^{\mathcal{F}}$$

satisfying the following conditions **(t1) - (t3)** expressing properties of reflexivity, monotonicity, and transitivity of the **consequence**

For any sets $F, F_0, F_1, F_2, F_3 \in 2^{\mathcal{F}}$,

(t1) $F \subseteq \mathbf{C}(F)$ **reflexivity**

(t2) if $F_1 \subseteq F_2$, then $\mathbf{C}(F_1) \subseteq \mathbf{C}(F_2)$, **monotonicity**

(t3) if $F_1 \subseteq \mathbf{C}(F_2)$ and $F_2 \subseteq \mathbf{C}(F_3)$, then $F_1 \subseteq \mathbf{C}(F_3)$, **transitivity**

Tarski Consequence Operation

We say that the consequence operation **C** has a **finite character** if additionally it satisfies the following condition **t4**

(t4) if a formula $B \in \mathbf{C}(F)$, then there exists a **finite** set $F_0 \subseteq F$, such that $B \in \mathbf{C}(F_0)$ **finiteness**.

The monotonicity condition **(t2)** and transitivity condition **(t3)** are often replaced by the following conditions **(t2')**, **(t3')**, respectively

(t2') if $B \in \mathbf{C}(F)$, then $B \in \mathbf{C}(F \cup F')$

(t3') $\mathbf{C}(F) = \mathbf{C}(\mathbf{C}(F))$

Consequence Operations Equivalency

Definition

Given a formal language $\mathcal{L} = (\mathcal{A}, \mathcal{F})$ and a Tarski **consequence** $\mathbf{C}$

A system $D = (\mathcal{L}, \mathbf{C})$ is called a Tarski **deductive system** for the language $\mathcal{L}$

Observe that Tarski's deductive system as a model of reasoning does not provide a **method** of actually defining a consequence operation; it **assumes** that it is given

We **prove** that the consequence operation $\mathbf{Cn}_S$ determined by S is a Tarski consequence operation $\mathbf{C}$

Consequence Operations Equivalency

Each **proof** system S provides a different **example** of a consequence operation

Each **proof** system S can be treated and a syntactic Tarski **deductive** system and the following holds

Theorem

Given a proof system $S = (\mathcal{L}, \mathcal{E}, LA, \mathcal{R})$

The consequence operation $\mathbf{Cn}_S$ is a Tarski consequence **C** in the language $\mathcal{L}$ of the system S and the system

$$D_S = (\mathcal{L}, \mathbf{Cn}_S)$$

is Tarski **deductive system**

We call it a **syntactic** deductive system **determined** by S

Chapter 4
General Proof Systems: Syntax and Semantics

Slides Set 1

PART 3 **Non Monotonic Reasoning** and **Syntactic Consistency**

Non Monotonic Reasoning

The Tarski consequence **C** models reasoning which is called after its condition **(t2)** or (**t2')** a monotonic reasoning

The monotonicity of reasoning was, since antiquity the the **basic** assumption while developing models for classical and well established non-classical logics

Recently many of new non- classical logics were developed and are being developed by computer scientists

Nevertheless they usually are built following the Tarski **definition** of **consequence** and are called as the others the **monotonic** logics

Non Monotonic Reasoning

A new type of important **Non-monotonic** logics have been proposed at the beginning of the 80s

Historically the most important proposals are:
Non-monotonic logic by McDermott and Doyle, **Default logic**, by Reiter, **Circumscription**, by McCarthy, and **Autoepistemic** logic, by Moore

The term **non-monotonic** logic covers a family of formal frameworks devised to capture and represent **defeasible** inference

Defeasible inference is an inference in which it is **possible** to draw **conclusions** tentatively, reserving the right to retract them in the light of further information

We included most standard **examples** in Chapter 1, **Slides Set 2**

Syntactic Consistency: Formal Theories

Formal theories play crucial role in mathematics and were historically defined for classical **predicate (first order)** logic and consequently for other non-classical logics

They are routinely called **first order theories**

We discuss them in detail in Chapter 10 dealing formally with classical predicate logic

First order theories are hence based on a proof systems S with a predicate (first order) language $\mathcal{L}$

We sometimes consider **formal theories** based on proof systems with a propositional language $\mathcal{L}$ and we call them **propositional theories**

Syntactic Consistency: Formal Theories

Given a proof system $S = (\mathcal{L}, \mathcal{E}, LA, \mathcal{R})$

We build (define) a **formal theory** based on S as follows.

1. We **select** a certain **finite** subset SA of expressions of S, **disjoint** with the logical axioms LA of S

The set SA is called a set of **specific** axioms of the **formal theory** based on S

2. We use set SA of **specific** axioms to define a language $\mathcal{L}_{SA}$, called a **language** of the formal theory

Here we have two cases

Syntactic Consistency: Formal Theories

c1 S is a first order proof system, i.e. $\mathcal{L}$ of S is a **predicate** language

We **define** the language $\mathcal{L}_{SA}$ by **restricting** the sets of constant, functional, and predicate symbols of $\mathcal{L}$ to constant, functional, predicate symbols **appearing** in the set SA of **specific axioms**

Both languages $\mathcal{L}_{SA}$ and $\mathcal{L}$ **share** the same set of propositional connectives

c2 S is a **propositional** proof system, i.e. $\mathcal{L}$ of S is a **propositional** language $\mathcal{L}_{SA}$ is defined by **restricting** $\mathcal{L}$ to connectives appearing in the set SA

Syntactic Consistency: Formal Theories

Definition

Given a proof system $S = (\mathcal{L}, \mathcal{E}, LA, \mathcal{R})$ and **finite** subset SA of expressions of S, **disjoint** with the logical axioms LA

The system

$$T = (\mathcal{L}, \mathcal{E}, LA, SA, \mathcal{R})$$

is called a **formal theory** based on S

The set SA is the set of **specific axioms** of T

The language $\mathcal{L}_{SA}$ defined by **c1** or **c2** is called the language of the **theory** T

Syntactic Consistency

Definition

A theory

$$T = (\mathcal{L},\ \mathcal{E},\ LA,\ SA,\ \mathcal{R})$$

is **consistent** if and only if there exists an expression $E \in \mathcal{E}_{SA}$ such that $E \notin \mathbf{T}(SA)$, i.e. such that

$$SA \nvdash_S E$$

otherwise the theory T is **inconsistent**.

Observe that the definition has purely syntactic meaning

Syntactic Consistency: Formal Theories

The **consistency** definition reflexes our intuition what proper notion of **provability** should mean

Namely, it says that a formal **theory** T based on a proof system S is **consistent** only when it **does not prove** all expressions (formulas in particular cases) of $\mathcal{L}_{SA}$

The **theory** T such that it **proves everything** stated in $\mathcal{L}_{SA}$ obviously should be, and is defined as **inconsistent**

Syntactic Consistency: Formal Theories

In particular, we have the following **syntactic definition** of consistency and inconsistency for any proof system S

Definition

A proof system

$$S = (\mathcal{L}, \mathcal{E}, LA, \mathcal{R})$$

is **consistent** if and only if there exists $E \in \mathcal{E}$ such that $E \notin \mathbf{P}_S$, i.e. such that

$$\nvdash_S E$$

otherwise S is **inconsistent**

Chapter 4
General Proof Systems: Syntax and Semantics

Slides Set 2

PART 5 **Semantics:** Soundness and Completeness

PART 6 **Exercises** and **Examples**

Chapter 4
General Proof Systems: Syntax and Semantics

Slides Set 2

PART 4 **Semantics:** Soundness and Completeness

General Proof Systems: Semantics

We define formally a **semantics** for a given proof system

$$S = (\mathcal{L}, \mathcal{E}, LA, \mathcal{R})$$

by specifying the semantic links of all its **components** as follows

Semantic Link1: Language $\mathcal{L}$

The language $\mathcal{L}$ of S can be **propositional** or **predicate**

Let denote by **M** a semantic for the language L

We call **M**, for short, a **semantics** for the proof system S

Proof Systems: Semantics

The **semantics** **M** can be classical or non-classical

M can be propositional or predicate depending of the language $\mathcal{L}$ of S

M can be extensional or not extensional

We use **M** as a general **symbol** for a **semantics**

Proof Systems: Semantics

Semantic Link 2: Set $\mathcal{E}$ of Expressions

We always have to **extend** a given semantics **M** for the language $\mathcal{L}$ of the system S to the set $\mathcal{E}$ of all **expression** of S

Sometimes, like in case of **Resolution** based proof systems we have also to **prove** a semantic equivalency of new created expressions $\mathcal{E}$ (sets of clauses) with appropriate formulas of $\mathcal{L}$

Example

In the automated theorem proving system **RS** presented in Chapter 6 the basic expressions $\mathcal{E}$ are finite sequences of formulas of the language $\mathcal{L}_{\{\neg,\cap,\cup,\Rightarrow\}}$

We **extend** the classical semantics for $\mathcal{L}$ to the set $\mathcal{F}^*$ of all finite sequences of formulas as follows:

For any $v : VAR \longrightarrow \{F, T\}$ and any $\Delta \in \mathcal{F}^*$, $\Delta = A_1, A_2, ..A_n$, we put

$$v^*(\Delta) = v^*(A_1, A_2, ..A_n)$$

$$= v^*(A_1) \cup v^*(A_2) \cup .... \cup v^*(A_n)$$

i.e. in a shorthand notation

$$\Delta \equiv (A_1 \cup A_2 \cup ... \cup A_n)$$

Semantic Link 3: Logical Axioms LA

Given a semantics **M** for $\mathcal{L}$ and its **extension** to the set $\mathcal{E}$ of all expressions

We extend the notion of **tautology** to the expressions and write

$$\models_{\mathbf{M}} E$$

to denote that the **expression** $E \in \mathcal{E}$ is a **tautology** under semantics **M** and we put

$$\mathbf{T_M} = \{E \in \mathcal{E} : \ \models_{\mathbf{M}} E\}$$

Logical axioms LA are always a subset of expressions that are **tautologies** of under the semantics **M**, i.e.

$$LA \subseteq \mathbf{T_M}$$

Semantic Link 4: Rules of Inference $\mathcal{R}$

We want the **rules of inference** $r \in \mathcal{R}$ to preserve truthfulness i.e. to be **sound** under the semantics **M**

Definition

Given an inference rule $r \in \mathcal{R}$

$$(r) \quad \frac{P_1 \ ; \ P_2 \ ; \ .... \ ; \ P_m}{C}$$

We say that the inference rule $r \in \mathcal{R}$ is **sound** under a semantics **M** if and only if all **M models** of the set $\{P_1, P_2, .P_m\}$ of its **premisses** are also **M models** of its **conclusion** C

In the case of propositional language and the extensional semantics **M** the **M models** are defined in terms of the truth assignment $v : VAR \longrightarrow LV$, where LV is the set of logical values for the semantics **M**, the **Sound Rule** definition becomes as follows

Definition

An inference rule $r \in \mathcal{R}$, such that

$$(r) \quad \frac{P_1 \; ; \; P_2 \; ; \; .... \; ; \; P_m}{C}$$

is sound under a semantics **M** if and only if the condition below holds or any $v : VAR \longrightarrow LV$

If $v \models_{\mathbf{M}} \{P_1, P_2, .P_m\}$, then $v \models_{\mathbf{M}} C$

Proof Systems: Semantics

Observe that we can rewrite the condition

If $v \models_{\mathbf{M}} \{P_1, P_2, .P_m\}$, then $v \models_{\mathbf{M}} C$

as follows

If $v^*(P_1) = v^*(P_2) = ... = v^*(P_m) = T$, then $v^*(C) = T$

Remark

A rule of inference can be **sound** under different semantics

But also rule of inference can be **sound** under one semantics and **not sound** under the other

Proof Systems: Semantics

Example

Given a propositional language $\mathcal{L}_{\{\neg, \cup, \Rightarrow\}}$

Consider two rules of inference:

$$(r1)\ \frac{(A \Rightarrow B)}{(B \Rightarrow (A \Rightarrow B))} \quad \text{and} \quad (r2)\ \frac{\neg\neg A}{A}$$

The rule (r1) is **sound** under classical, **H** and **L** semantics

The (r2) is **sound** under classical and **L** semantics

The (r2) is **not sound** under **H** semantics

We introduce now new important notions of **strongly sound** rule under a semantics **M**

Definition

Given a language $\mathcal{L}$, an inference rule $r \in \mathcal{R}$ of the form

$$(r) \quad \frac{P_1 \ ; \ P_2 \ ; \ .... \ ; \ P_m}{C}$$

is **strongly sound** under a semantics **M** if and only if the following condition holds for all **M** model structures $\mathcal{M}$,

$$\mathcal{M} \models_{\mathbf{M}} \{P_1, P_2, .P_m\} \quad \text{if and only if} \quad \mathcal{M} \models_{\mathbf{M}} C$$

In case of a **propositional** language $\mathcal{L}$ and extensional semantics **M** the **M** model structure $\mathcal{M}$ is the truth assignment v and the **strong soundness** condition is as follows

For for any $v : VAR \longrightarrow LV$,

$$v \models_{\mathbf{M}} \{P_1, P_2, .P_m\} \quad \text{if and only if} \quad v \models_{\mathbf{M}} C$$

Proof Systems: Semantics

Example

Given a propositional language $\mathcal{L}_{\{\neg,\cup,\Rightarrow\}}$

Consider two rules of inference:

$$(r1)\ \frac{A\ ;\ B}{(A \cup \neg B)} \qquad \text{and} \qquad (r2)\ \frac{A}{\neg\neg A}$$

Both rules (r1) and (r2) are **sound** under classical and **H** semantics

The rule (r2) is **strongly** under classical semantics

The rule (r2) is **not strongly sound** under **H** semantics

The rule (r1) is **not strongly sound** under either semantics

Proof Systems: Semantics

Now we **define** a notion of a **sound** and **strongly sound** proof system. Strongly sound proof systems play a role in **constructive** proofs of **completeness theorem**. This is why we introduce them here

Definition

Given a proof system $S = (\mathcal{L}, \mathcal{E}, LA, \mathcal{R})$

We say that the proof system S is **sound** under a semantics **M** if and only if the following conditions hold

C1 $LA \subseteq \mathbf{T_M}$

C2. Each rule of inference $r \in \mathcal{R}$ is **sound** under **M**

The proof system S is **strongly sound** under a semantics **M** if the condition **C2** is replaced by the following condition

C2' Each rule of inference $r \in \mathcal{R}$ is **strongly sound** under **M**

Example

Consider aproof system

$$S = (\mathcal{L}_{\{\neg, \Rightarrow\}},\ \mathcal{F},\ \{(\neg\neg A \Rightarrow A),\ (A \Rightarrow (\neg A \Rightarrow B))\},\ \mathcal{R} = \{(r)\})$$

where

$$(r)\ \frac{(A \Rightarrow B)}{(B \Rightarrow (A \Rightarrow B))}$$

The proof system S is **sound**, but **not strongly sound** under classical and **L** semantics

S is **not sound** under **H** semantics

Proof

We proof here only the condition **C1**. The complete proof, as proofs of many other examples, is included in the book chapter

Proof Systems: Semantics

C1 $LA \subseteq \mathbf{T_M}$

Both axioms are basic classical tautologies

Hence to prove that first axiom is **L** tautology we we have to verify only the case (shorthand notation) $A = \perp$

We evaluate

$$\neg\neg \perp \Rightarrow \perp = \neg \perp \Rightarrow \perp = \perp \Rightarrow \perp = T$$

This proves $\models_{\mathbf{L}} (\neg\neg A \Rightarrow A)$

Proof Systems: Semantics

Consider the second axiom

$$(A \Rightarrow (\neg A \Rightarrow B))$$

Observe that $(A \Rightarrow (\neg A \Rightarrow B)) = \perp$ if and only if $A = T$ and

$(\neg A \Rightarrow B) = \perp$ if and only if $(\neg T \Rightarrow B) = \perp$ if and only if $(F \Rightarrow B) = \perp$, what is **impossible** under **L** semantics

This proves

$$\models_{\mathbf{L}} (A \Rightarrow (\neg A \Rightarrow B))$$

and the condition **C1** holds for the classical and **L** semantics

Proof Systems: Semantics

We prove now that

$$\not\models_{\mathbf{H}} (\neg\neg A \Rightarrow A)$$

as follows

Consider any truth assignment such that $A = \bot$

We evaluate

$$\neg\neg \bot \Rightarrow \bot = \neg \bot \Rightarrow \bot = F \Rightarrow \bot = \bot$$

This proves that S is **not sound** under **H** semantics.

Proof Systems: Soundness Theorem

When we define (develop) a proof system S and its semantics **M** our **first goal** is to make sure that it the proof system S is a "**sound one**", i.e. that it has a property stating that all we **prove** in S is always true with respect to the given semantics **M**

This **goal** is established by formulating and proving a theorem, called **Soundness Theorem** that defines a relationship between **provability** in a proof system S and the **tautologies** defined by the system S semantics **M**

Proof Systems: Soundness Theorem

Let $\mathbf{P}_S = \{E \in \mathcal{E} : \vdash_S E\}$ be the set of all provable expressions of S, and let $\mathbf{T_M}$ be a set of all expressions of S that are **M** tautologies i.e. $\mathbf{T_M} = \{E \in \mathcal{E} : \models_{\mathbf{M}} E\}$

Soundness Theorem

Given a proof system S and its semantics **M**,

$$\mathbf{P}_S \subseteq \mathbf{T_M}$$

i.e. for any $E \in \mathcal{E}$, the following implication holds

$$\text{if } \vdash_S E \text{ then } \models_{\mathbf{M}} E$$

Observe that the **Soundness Theorem** holds for S if and only if the proof system S is **sound**, hence the **name** of the theorem.

Proof Systems: Soundness Theorem

Obviously, if S is **not sound** there is an expression E such that $\vdash_S E$ and E is not **M** tautology. Hence $\mathbf{P}_S \not\subseteq \mathbf{T}_\mathbf{M}$ and the **Soundness Theorem** fails

Assume now that S is **sound** and $\vdash_S E$

We prove that $E \in \mathbf{T}_\mathbf{M}$, by Mathematical Induction over the length of a proof of E and we have proved the following

Soundness Fact

Given a proof system $S = (\mathcal{L}, \mathcal{E}, LA, \mathcal{R})$

In order to prove/disprove the **Soundness Theorem** for S under semantics **M** it is sufficient to to verify the two conditions:

1. $LA \subseteq \mathbf{T}_\mathbf{M}$ and

2. Each rule of inference $r \in \mathcal{R}$ of S is **sound** under **M**

Proof Systems: Completeness Theorem

The next step in developing a proof system (logic) is to formally state and **answer** another necessary **question**

Given a proof system S, about which we already **know** that **all it proves** is a tautology with respect to its given semantics

Can S **prove** all statements we know to be tautologies with respect to its semantics?

The answer is formulated in form of a theorem, called **Completeness Theorem** that has to be proved/disproved about the proof system S

Proof Systems: Completeness Theorem

Completeness Theorem

Given a proof system S and its semantics $\mathbf{M}$,

$$\mathbf{P}_S = \mathbf{T}_\mathbf{M}$$

i.e. for any $E \in \mathcal{E}$, the following holds

$$\vdash_S E \quad \text{if and only if} \quad \models_\mathbf{M} E$$

The **Completeness Theorem** consists of two parts

Part 1 **Soundness Theorem:** $\mathbf{P}_S \subseteq \mathbf{T}_\mathbf{M}$

Part 2 **Completeness Part:** $\mathbf{T}_\mathbf{M} \subseteq \mathbf{P}_S$

Proof Systems: Completeness Theorem

Proving/ disproving the **Soundness Theorem** for S under a semantics **M** is usually a straightforward and not a very difficult task

Proving/ disproving the of the **Completeness Part** is always **crucial** and **very difficult** task

There are many methods and techniques for doing so, even for **classical** proof systems (logic) alone

Non-classical logics usually require **new** sometimes very sophisticated methods

Proof Systems: Completeness Theorem

We present two proofs of the **Completeness Theorem** for propositional Hilbert style proof system for classical logic in chapter 5

We present constructive proofs for **automated theorem proving** systems for classical propositional logic in chapter 6

We discuss the proofs of the **Completeness Theorem** for Intuitionistic and Modal Logics in chapter 7

We provide the proofs of the **Completeness Theorem** for classical predicate logic in chapter 9 (Hilbert style) and chapter 10 (Gentzen style)

Chapter 4
General Proof Systems: Syntax and Semantics

Slides Set 2

PART 5 Exercises and Examples

Proof Systems: Exercises

Exercise

Given a proof system:

$$S = (\mathcal{L}_{\{\neg,\Rightarrow\}},\ \mathcal{F}\ \ LA = \{(A \Rightarrow A), (A \Rightarrow (\neg A \Rightarrow B))\}, \{(r)\}\ )$$

for

$$(r)\ \frac{(A \Rightarrow B)}{(B \Rightarrow (A \Rightarrow B))})$$

1. Prove that S is **sound**, but **not strongly sound** under classical semantics

2. Prove that S is **not sound** under **K** semantics

3. Write a **formal proof** in S with 2 applications of rule (r)

Proof Systems: Exercises

Solution

In order to prove **1.** and **2.** we have to verify conditions

C1 $LA \subseteq \mathbf{T_M}$

C2. Each $r \in \mathcal{R}$ is **sound**

for **soundness**, and **C1** , **C2'** for **strong soundness**, for

C2' Each $r \in \mathcal{R}$ is **strongly sound**

Observe that both axioms of S are basic classical tautologies, so **C1** holds

Solution

Consider the rule of inference of

$$(r)\ \frac{(A \Rightarrow B)}{(B \Rightarrow (A \Rightarrow B))}$$

Take any v such that $v^*((A \Rightarrow B))) = T$

We **evaluate** logical value of the conclusion under the truth assignment v (and classical semantics) as follows

$v^*(B \Rightarrow (A \Rightarrow B)) = v^*(B) \Rightarrow T = T$, for any formula B and any value of $v^*(B)$

This proves that S is **sound** under classical semantics

S is **not strongly sound** as

$$(A \Rightarrow B) \not\equiv (B \Rightarrow (A \Rightarrow B))$$

System S is **not sound** under **K** semantics because axiom $(A \Rightarrow A)$ **is not** a **K** semantics tautology

Solution

3. There are many solutions, i.e. one can construct many required **formal proofs**

Here is **one** of them, i.e. a sequence

$$A_1,\ A_2,\ A_3$$

where

$A_1 = (A \Rightarrow A)$

Axiom

$A_2 = (A \Rightarrow (A \Rightarrow A))$

Rule (r) application one for $A = A,\ B = A$

$A_3 = ((A \Rightarrow A) \Rightarrow (A \Rightarrow (A \Rightarrow A)))$

Rule (r) application one for $A = A,\ B = (A \Rightarrow A)$

Exercise

Given a proof system:

$$S = (\mathcal{L}_{\{\cup, \Rightarrow\}},\ \mathcal{F},\ LA = \{A1, A2\},\ (r)\ \frac{(A \Rightarrow B)}{(A \Rightarrow (A \Rightarrow B))}\ )$$

where $A1 = (A \Rightarrow (A \cup B))$, $A2 = (A \Rightarrow (B \Rightarrow A))$

1. Prove that S is **sound** under classical semantics and determine whether S is **sound** or **not sound** under **K** semantics.

2. Write a **formal proof** B_1, B_2, B_3 in S with **two** applications of the rule (r) that starts with axiom A1, i.e such that $B_1 = (A \Rightarrow (A \cup B))$

3. Write a **formal proof** B_1, B_2 in S with **one** application of the rule (r) that starts with axiom A2, ı.e such that $A_1 = (A \Rightarrow (B \Rightarrow A))$

Solution

1. All axioms of S are **basic** classical tautologies

The **proof** (in shorthand notation) of **soundness** of the rule

$$(r)\ \frac{(A \Rightarrow B)}{(A \Rightarrow (A \Rightarrow B))}$$

is as follows. Assume $(A \Rightarrow B) = T$. Hence the logical value of conclusion is $(A \Rightarrow (A \Rightarrow B)) = (A \Rightarrow T) = T$ for all A, and S is **sound** under classical semantics

S is **not sound** under **K** semantics

Take a truth assignment such that $A = \perp,\ B = \perp$

We evaluate logical value of axiom A1 (in shorthand notation)

$(A \Rightarrow (A \cup B)) = (\perp \Rightarrow (\perp \cup \perp)) = \perp$ and $\not\models_{\mathbf{K}} (A \Rightarrow (A \cup B))$

Solution

2. The required formal proof B_1, B_2, B_3 is as follows

$B_1 = (A \Rightarrow (A \cup B))$

Axiom

$B_2 = (A \Rightarrow (A \Rightarrow (A \cup B)))$

Rule (r) application for $A = A$ and $B = (A \cup B)$

$B_3 = (A \Rightarrow (A \Rightarrow (A \Rightarrow (A \cup B))))$

Rule (r) application for $A = A$ and $B = (A \Rightarrow (A \cup B))$

Solution

3. The required formal proof B_1, B_2 is as follows

$B_1 = (A \Rightarrow (B \Rightarrow A))$

Axiom

$B_2 = (A \Rightarrow (A \Rightarrow (B \Rightarrow A)))$

Rule (r) application for $A = A$ and $B = (B \Rightarrow A)$

Exercise

Let S be the following proof system

$$S = (\ \mathcal{L}_{\{\Rightarrow,\cup,\neg\}},\ \mathcal{F},\ A1,\ (r1),\ (r2)\ )$$

where the logical axiom A1 is $A1 = (A \Rightarrow (A \cup B))$

Rules of inference (r1), (r2) are:

$$(r1)\ \frac{A\ ;\ B}{(A \cup \neg B)}, \qquad (r2)\ \frac{A\ ;\ (A \cup B)}{B}$$

1. **Verify** whether S is **sound/not sound** under classical semantics

2. **Find** a **formal proof** of $\neg(A \Rightarrow (A \cup B))$ in S, ie. show that $\vdash_S \neg(A \Rightarrow (A \cup B))$

3. **Does** $\vdash_S \neg(A \Rightarrow (A \cup B))$ **prove** that $\models \neg(A \Rightarrow (A \cup B))$?

Solution

1. The system S is **not sound**

Take any v, such that $v^*(A) = T$ and $v^*(B) = F$

The premiss $(A \cup B$ of the rule (r2) is T under v

Its conclusion under v is $v^*(B) = F$

2. The **formal proof** of $\neg(A \Rightarrow (A \cup B))$ is as follows

B_1: $(A \Rightarrow (A \cup B))$

axiom

B_2: $(A \Rightarrow (A \cup B))$

axiom

B_3: $((A \Rightarrow (A \cup B)) \cup \neg(A \Rightarrow (A \cup B)))$

rule (r1) application to B_1 and B_2

B_4: $\neg(A \Rightarrow (A \cup B))$

rule (r2) application to B_1 and B_3

Solution

3. System S is **not sound**

In general, the existence of a **formal proof** in a not sound proof systems **does not** guarantee that what was proved is a **tautology**

Moreover, the **non-sound** rule (r2) was used in the proof of the formula

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

so we have that

$$\not\models \neg(A \Rightarrow (A \cup B))$$

Exercise

Create your pwn 3 valued extensional semantics **M** for the language

$$\mathcal{L}_{\{\neg,\ \mathbf{L},\ \cup,\ \Rightarrow\}}$$

by **defining** the connectives $\neg, \cup, \Rightarrow$ on a set $\{F, \perp, T\}$ of logical values

You must **follow** the following assumptions **a1, a2, a3**

a1 The third logical value value is intermediate between truth and falsity, i.e. the set $\{F, \perp, T\}$ of logical values is ordered as follows

$$F < \perp < T$$

a2 The value T is the **designated** value

Proof Systems: Exercises

a3 The connective **L** is one argument connective that reads "like", "likes"

The **semantics** has to **model** a situation in which one "likes" only the truth, i.e. the logical value T

It means the connective **L** must be such that

$$\mathbf{L}T = T, \quad \mathbf{L}\perp = F, \quad \text{and} \quad \mathbf{L}F = F$$

The connectives $\neg, \cup, \Rightarrow$ can be **defined** as you wish, but you have to define them in such a way to make sure that

$$\models_{\mathbf{M}} (\mathbf{L}A \cup \neg\mathbf{L}A)$$

Proof Systems: Example

Example

Here is an example of a required simple semantics

We define the logical connectives by writing functions defining connectives in form of the truth tables.

M Semantics

L	F	⊥	T
	F	*F*	T

¬	F	⊥	T
	T	*F*	F

M Semantics

$\cap$	F	$\bot$	T
F	F	F	F
$\bot$	F	$\bot$	$\bot$
T	F	$\bot$	T

$\cup$	F	$\bot$	T
F	F	$\bot$	T
$\bot$	$\bot$	T	T
T	T	T	T

$\Rightarrow$	F	$\bot$	T
F	T	T	T
$\bot$	T	$\bot$	T
T	F	F	T

We verify by simple evaluation whether the condition **s3** is satisfied, i.e. whether $\models_{\mathbf{M}} (LA \cup \neg LA)$

Let $v : VAR \longrightarrow \{F, \bot, T\}$ be any truth assignment

For any formula A, $v^*(A) \in \{F, \bot, T\}$ and

$\mathbf{L}F \cup \neg \mathbf{L}F = \mathbf{L}F \cup \neg \mathbf{L}F = F \cup \neg F \cup T = T$

$\mathbf{L} \bot \cup \neg \mathbf{L} \bot = F \cup \neg F = F \cup T = T$

$\mathbf{L}T \cup \neg \mathbf{L}T = T \cup \neg T = F \cup T = T$

Proof Systems: Exercise

Exercise

Let S be the following proof system

$$S = (\ \mathcal{L}_{\{\neg, \mathbf{L}, \cup, \Rightarrow\}},\ \mathcal{F},\ \ \{A1, A2\},\ \ \{(r1),\ (r2)\}\ )$$

where $A1:\ \ (\mathbf{L}A \cup \neg\mathbf{L}A)$, $\qquad A2:\ (A \Rightarrow \mathbf{L}A)$,

$$(r1)\ \frac{A\ ;\ B}{(A \cup B)}, \qquad\qquad (r2)\ \frac{A}{\mathbf{L}(A \Rightarrow B)}$$

1. Show, by constructing a proper formal proof that

$$\vdash_S ((\mathbf{L}b \cup \neg\mathbf{L}b) \cup \mathbf{L}((\mathbf{L}a \cup \neg\mathbf{L}a) \Rightarrow b)))$$

2. Verify whether the system S is **M**-sound under the semantics **M** developed in the previous Example

3. If the system S **is not** **M**-sound then define a new semantics **N** would make S **sound**