cse371/math371
LOGIC

Professor Anita Wasilewska
LECTURE 4
Chapter 4
GENERAL PROOF SYSTEMS

PART 1: Introduction- Intuitive definitions
PART 2: Formal Definition of a Proof System
PART 3: Formal Proofs and Simple Examples
PART 4: Consequence, Soundness and Completeness
PART 5: Decidable and Syntactically Decidable Proof Systems
PART 1: General Introduction
Proof Systems - Intuitive Definition

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 $L$.

Proof systems are inference machines with statements called provable statements being their final products.
Semantical Link

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.
We always choose as a set of **logical axioms** $\text{LA}$ 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.
Semantical Link

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

**Specific axioms** $SA$ of the proof system $S$ is any finite set of formulas that are not **tautologies**, and hence they are always disjoint with the set of **logical axioms** $LA$ of $S$.

The **proof system** $S$ with added set of **specific axioms** $SA$ is called a **formal theory** based on $S$. 
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 system with axioms as a starting point.

We depict it informally on the next slide.
Inference Machine

AXIOMS

↓ ↓ ↓

RULES applied to AXIOMS

↓ ↓ ↓

RULES applied to any expressions above

↓ ↓ ↓

Provable formulas
Semantical Link

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 semantics $M$.

Rules of inference can be sound under one semantics and not sound under another.
Goal 1
When developing a proof system $S$ the first goal is 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 $\text{LA}$ of $S$ only, then $A$ is a tautology under the semantics $M$.
Propositional Proof Systems

We discuss here first only proof systems for propositional languages and call them *proof systems* for different propositional logics.

**Remember**

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 classical logic semantics might not be sound under Ł logic semantics, or K logic semantics, or others.
Completeness of the Proof Systems

In general there are many proof systems that are sound under a given semantics, i.e. there are many sound proof systems for a given logic semantically defined.

Given a proof system $S$ with logical axioms $LA$ that is sound under a semantics $M$.

**Notation**

Denote by $T_M$ the set of all tautologies defined by the semantics $M$, i.e. we have that

$$T_M = \{ A \in \mathcal{F} : \vdash_M A \}$$
Completeness Property

A natural question arises:
Are all tautologies i.e formulas $A \in T_M$ provable in the system $S$?

We assume that we have already proved that $S$ is sound under the semantics $M$.

The positive answer to this question is called completeness property of the system $S$. 
Completeness Theorem

Goal 2
Given for a sound proof system $S$ under its semantics $M$, our the 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 $S$ iff $A$ is a tautology under the semantics $M$.
We write the Completeness Theorem symbolically as

$$\vdash_S A \iff \models_M A$$

Completeness Theorem is composed of two parts: Soundness Theorem and the Completeness Part that proves the completeness property of a sound proof system.
Proving Soundness and Completeness

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, and then we prove that all inference rules of the system $S$ preserve the notion of the truth.

Proving the completeness part of the Completeness Theorem is always a crucial, difficult and sometimes impossible task.
OUR PLAN

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

We will present a constructive proofs of Completeness Theorem for two different Gentzen style automated theorem proving systems for classical Logic in Chapter 6.

We discuss the Inuitionistic Logic in Chapter 7.

Predicate Logics are discussed Chapters 8, 9, 10, 11.
PART 2
PROOF SYSTEMS: Formal Definitions
Proof System $S$

In this section we present **formal definitions** of the following notions:

- **Proof system** $S$
- **Formal proof** from **logical axioms** in a proof system $S$
- **Formal proof** from **specific axioms** in a proof system $S$
- **Formal Theory** based on a proof system $S$

We also give **examples** of different simple **proof systems**
Components: Language

Language $L$ of a proof system $S$ is any formal language $L$

$L = (A, F)$

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

The Language $L$ can be propositional or first order (predicate) but we discuss propositional languages first.
Expressions $\mathcal{E}$ of a proof system $S$

Given a set $\mathcal{F}$ of well formed formulas of the language $L$ of the system $S$

We often extend the set $\mathcal{F}$ to some set $\mathcal{E}$ of expressions build out of the language $L$ 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 different sets of expressions build out of formulas of the language $\mathcal{L}$.

In Chapters 6 and 10 we consider finite sequences of formulas instead of formulas, as basic expressions of the proof systems RS and RQ.

We also present there proof systems that use yet other kind of expressions, called original Gentzen sequents or their modifications.

Some systems use yet other expressions such as clauses, sets of clauses, or sets of formulas, others use yet still different expressions.
Semantical Link

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

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 in Resolution case) with appropriate formulas of $L$. 
For example, in the automated theorem proving system RS presented in Chapter 6 the basic expressions $\mathcal{E}$ are finite sequences of formulas of $\mathcal{L} = \mathcal{L}\{\neg, \cap, \cup, \Rightarrow\}$.

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

For any $\nu : \text{VAR} \rightarrow \{F, T\}$ and any $\Delta \in \mathcal{F}^*$, $\Delta = A_1, A_2, \ldots A_n$, we put

$$\nu^*(\Delta) = \nu^*(A_1, A_2, \ldots A_n)$$

$$= \nu^*(A_1) \cup \nu^*(A_2) \cup \ldots \cup \nu^*(A_n)$$

i.e. in a shorthand notation

$$\Delta \equiv (A_1 \cup A_2 \cup \ldots \cup A_n)$$
Components: Logical Axioms

Logical axioms $LA$ of $S$ form a non-empty subset of the set $\mathcal{E}$ of expressions of the proof system $S$, i.e.

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

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

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

We assume here that the set $LA$ of logical axioms is always finite, i.e. that we consider here finitely axiomatizable systems.

In general, we assume that the set $LA$ is primitively recursive, i.e. that there is an effective procedure to determine whether a given expression $E \in \mathcal{E}$ is or is not in $AL$. 
Components: Axioms

Semantical link
Given a semantics $M$ for $L$ and its extension to the set $E$ of all expressions.

We extend the notion of tautology to the expressions and write

$$\models_M E$$

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

$$T_M = \{ E \in E : \models_M E \}$$

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

$$LA \subseteq T_M$$
Components: Rules of Inference

Rules of inference $\mathcal{R}$
We assume that a proof system 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 a given string of expressions form its premisses and conclusion or do not, i.e. that all rules $r \in \mathcal{R}$ are primitively recursive.
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.

All \( r \in \mathcal{R} \) are primitively recursive relations.
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

\[
(r) \quad \frac{P_1 \quad P_2}{C}
\]

\(m\) premisses rule

\[
(r) \quad \frac{P_1 \quad P_2 \quad \ldots \quad P_m}{C}
\]
Semantic Link: Sound Rules of Inference

**Given** some \( m \) premisses rule

\[
(r) \quad \frac{P_1 ; P_2 ; \ldots ; P_m}{C}
\]

**Semantical link**

Given a semantics \( M \) for the language \( L \) and for the set of expressions \( E \)

We want the rules of inference \( r \in R \) to preserve truthfulness i.e. to be **sound** under the semantics \( M \)
General Definition: Sound Rule of Inference

Definition

Given an inference rule \( r \in \mathcal{R} \)

\[
(r) \quad \frac{P_1 ; P_2 ; \ldots ; P_m}{C}
\]

We say that the inference rule \( r \in \mathcal{R} \) is sound under a semantics \( \mathcal{M} \) if and only if all \( \mathcal{M} \)-models of the set \( \{P_1, P_2, \ldots, P_m\} \) of its premisses are also \( \mathcal{M} \)-models of its conclusion \( C \).
Propositional Definition: Sound Rule of Inference

In propositional languages case, the semantics $M$, and hence the $M$-models are defined in terms of the truth assignment $v : VAR \rightarrow LV$, where $LV$ is the set of logical values for the semantics $M$

Definition
An inference rule $r \in R$, such that

\[
\begin{align*}
(r) \quad P_1 ; P_2 ; \ldots ; P_m \\
\quad C
\end{align*}
\]

is sound under a semantics $M$ if and only if

the condition below holds or any $v : VAR \rightarrow LV$

If $v \models_M \{P_1, P_2, \ldots, P_m\}$, then $v \models_M C$
Example

Given a rule of inference

\[
(r) \quad \begin{array}{c}
(A \Rightarrow B) \\
(B \Rightarrow (A \Rightarrow B))
\end{array}
\]

Prove that \( (r) \) is sound under classical semantics

Let \( v \) be any truth assignment, such that \( v \models (A \Rightarrow B) \), i.e.
by definition \( v^*(A \Rightarrow B) = T \)

We evaluate logical value of the conclusion under \( v \) as follows

\[
v^*(B \Rightarrow (A \Rightarrow B)) = v^*(B) \Rightarrow T = T
\]

for any \( B \) and any value of \( v^*(B) \)

This proves that \( v \models (B \Rightarrow (A \Rightarrow B)) \) and hence the soundness of \( (r) \)
Formal Definition: Proof System

Definition
By a proof system we understand a quadruple

$$S = (L, E, LA, R)$$

where
$L = \{A, F\}$ is a language of $S$ with a set $F$ of formulas
$E$ is a set of expressions of $S$
In particular case $E = F$
$LA \subseteq E$ is a non-empty, finite set of logical axioms of $S$
$R$ is a non-empty, finite set of rules of inference of $S$
PART 3: Formal Proofs
Simple Examples of Proof Systems
Provable Expressions

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
Definition: Direct Consequence

Formal definitions are as follows

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

\[
\begin{array}{c}
(r) \quad P_1 ; P_2 ; .... ; P_m \\
\hline \\
C
\end{array}
\]

$C$ is called a \textbf{direct consequence} of $P_1, ... P_m$ by virtue of the rule $r \in \mathcal{R}$
Definition: Formal Proof

Formal Proof of an expression \( E \in \mathcal{E} \) in a proof system \( S = (\mathcal{L}, \mathcal{E}, \mathcal{LA}, \mathcal{R}) \) is a sequence

\[ A_1, A_2, \ldots, A_n \quad \text{for} \quad n \geq 1 \]

of expressions from \( \mathcal{E} \), such that

\[ A_1 \in \mathcal{LA}, \quad A_n = E \]

and for each \( 1 < i \leq n \), either \( A_i \in \mathcal{LA} \) 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 \( A_1, A_2, \ldots, A_n \)
Formal Proof Notation

We write

$$\vdash_S E$$

to denote that $E \in \mathcal{E}$ has a proof in $S$

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

Any $E \in \mathcal{E}$, such that $\vdash_S E$ is called a provable expression of $S$

The set of all provable expressions of $S$ is denoted by $P_S$, i.e. we put

$$P_S = \{ E \in \mathcal{E} : \vdash_S E \}$$
PART 4: Hypothesis, Consequence, Soundness and Completeness
Proof from Hypothesis

While proving expressions we often use some extra information available, besides the axioms of the proof system. This extra information is called hypothesis in the proof.

Let $\Gamma \subseteq \mathcal{E}$ be a set expressions called hypothesis.

A proof of $E \in \mathcal{E}$ from the set of hypothesis $\Gamma$ in $S$ is a formal proof in $S$, where the expressions from $\Gamma$ are treated as additional hypothesis added to the set $\text{LA}$ of the logical axioms of the system $S$.

Notation: $\Gamma \vdash_S A$

We read it: $A$ has a proof in $S$ from the set $\Gamma$ (and logical axioms $\text{LA}$).
Definition: Proof from Hypothesis

Definition

We say that $A$ has a proof in $S$ from the set $\Gamma$ (and logical axioms $LA$) if and only if there is a sequence $A_1, \ldots, A_n$ of expressions from $\mathcal{E}$, such that

$A_1 \in LA \cup \Gamma, \quad A_n = A$

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

We denote it as $\Gamma \vdash_S A$.
Special Cases

We usually consider and use the case when the set of hypothesis is finite.

Case of $\Gamma \subseteq \mathcal{E}$ finite set and $\Gamma = \{B_1, B_2, ..., B_n\}$

We use notation

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

for $\{B_1, B_2, ..., B_n\} \vdash_S A$

Case of $\Gamma = \emptyset$ is also a special one.

By the definition of a proof of $A$ from $\Gamma$, $\emptyset \vdash A$ means that in the proof of $A$ we use only axioms LA of $S$

We hence use notation $\vdash_S A$ to denote that $A$ has a proof from empty $\Gamma$; i.e. $A$ has a proof from logical axioms only
Definition: Consequences of $\Gamma$

Definition
For any $\Gamma \subseteq \mathcal{E}$, and $A \in \mathcal{E}$,
If $\Gamma \vdash_S A$, then $A$ is called a consequence of $\Gamma$ in $S$.

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

$$\text{Cn}_S(\Gamma) = \{ A \in \mathcal{E} : \Gamma \vdash_S A \}$$
Definition: Consequence Operation

Observe that by defining a consequence of $\Gamma$ in $S$, we define in fact a function which to every set $\Gamma \subseteq \mathcal{E}$ assigns a set of all its consequences $C_{nS}(\Gamma)$.

We denote this function by $C_{nS}$ and adopt the following Definition:

Any function

$$C_{nS} : 2^\mathcal{E} \rightarrow 2^\mathcal{E}$$

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

$$C_{nS}(\Gamma) = \{A \in \mathcal{E} : \Gamma \vdash_S A\}$$

is called the consequence operation in $S$. 
Consequence Operation: Monotonicity

Take any consequence operation

$\text{Cn}_S : 2^E \rightarrow 2^E$

Monotonicity Property
For any sets $\Gamma, \Delta$ of expressions of $S$,
if $\Gamma \subseteq \Delta$ then $\text{Cn}_S(\Gamma) \subseteq \text{Cn}_S(\Delta)$

Exercise: write the proof;
it follows directly from the definition of $\text{Cn}_S$ and definition of the formal proof
Consequence Operation: Transitivity

Take any consequence operation

\[ \text{Cn}_S : 2^E \rightarrow 2^E \]

Transitivity Property

For any sets \( \Gamma_1, \Gamma_2, \Gamma_3 \) of expressions of \( S \),

if \( \Gamma_1 \subseteq \text{Cn}_S(\Gamma_2) \) and \( \Gamma_2 \subseteq \text{Cn}_S(\Gamma_3) \), then \( \Gamma_1 \subseteq \text{Cn}_S(\Gamma_3) \)

Exercise: write the proof;

it follows directly from the definition of \( \text{Cn}_S \) and definition of the formal proof
Consequence Operation: Finiteness

Take any consequence operation

\[ \text{Cn}_S : 2^\mathcal{E} \rightarrow 2^\mathcal{E} \]

Finiteness Property

For any expression \( A \in \mathcal{E} \) and any set \( \Gamma \subseteq \mathcal{E} \),
\[ A \in \text{Cn}_S(\Gamma) \text{ if and only if there is a finite subset } \Gamma_0 \text{ of } \Gamma \text{ such that } A \in \text{Cn}_S(\Gamma_0) \]

Exercise: write the proof;
it follows directly from the definition of \( \text{Cn}_S \) and definition of the formal proof
Definition: Sound $S$

Definition
Given a proof system

$$S = (L, E, LA, R)$$

We say that the system $S$ is **sound** under a semantics $M$ iff the following conditions hold

1. $LA \subseteq T_M$
2. Each rule of inference $r \in R$ is **sound**
Example

Given a proof system:

\[ S = (\mathcal{L}_{\neg,\Rightarrow}, \mathcal{F}, \{(A \Rightarrow A), (A \Rightarrow (\neg A \Rightarrow B))\}), \quad (r) \frac{(A \Rightarrow B)}{(B \Rightarrow (A \Rightarrow B))} \]

1. Prove that \( S \) is **sound** under **classical semantics**
2. Prove that \( S \) is **not sound** under \( K \) semantics
Example

1. Both axioms of $S$ are basic classical tautologies and we have just proved that the rule of inference (r) is sound, hence $S$ is sound

2. Axiom $(A \Rightarrow A)$ is not a $K$ semantics tautology
   Any truth assignment $v$ such that $v^*(A) = \bot$ is a counter-model for it
   This proves that $S$ is not sound under $K$ semantics
Soundness Theorem

Let $P_S$ be the set of all provable expressions of $S$ i.e.

$$P_S = \{ A \in \mathcal{E} : \vdash_S A \}$$

Let $T_M$ be a set of all expressions of $S$ that are tautologies under a semantics $M$, i.e.

$$T_M = \{ A \in \mathcal{E} : \models_M A \}$$

**Soundness Theorem** for $S$ and semantics $M$

$$P_S \subseteq T_M$$

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

If $\vdash_S A$, then $\models_M A$.

**Exercise:** prove by Mathematical Induction over the length of a proof that if $S$ is sound, the Soundness Theorem holds for $S$
Completeness Theorem

Completeness Theorem for $S$ and semantics $M$

\[ P_S = T_M \]

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

\[ \vdash_S A \quad \text{if and only if} \quad \models_M A \]

The Completeness Theorem consists of two parts:

Part 1: Soundness Theorem

\[ P_S \subseteq T_M \]

Part 2: Completeness Part of the Completeness Theorem

\[ T_M \subseteq P_S \]
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 $L$. We sometimes consider formal theories based on proof systems with a propositional language $L$ 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. $L$ of $S$ is a predicate language
We define the language $L_{SA}$ by restricting the sets of constant, functional, and predicate symbols of $L$ to constant, functional, predicate symbols appearing in the set $SA$ of specific axioms
Both languages $L_{SA}$ and $L$ share the same set of propositional connectives

c2  $S$ is a propositional proof system, i.e. $L$ of $S$ is a propositional language $L_{SA}$ is defined by restricting $L$ 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

\[ T = (\mathcal{L}, \mathcal{E}, \text{LA}, \text{SA}, \mathcal{R}) \]

is consistent if and only if there exists an expression

\[ E \in \mathcal{E}_{\text{SA}} \] such that \( E \notin T(\text{SA}) \), i.e. such that

\[ \text{SA} \not\vdash_E E \]

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

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

is **consistent** if and only if there exists $E \in \mathcal{E}$ such that $E \not\in P_S$, i.e. such that

$$ \kappa_S E $$

otherwise $S$ is **inconsistent**
Given a proof system $S = (\mathcal{L}, \mathcal{E}, LA, R)$. Let a set $SA \subseteq \mathcal{E}$ be such that

$$SA \cap T_M = \emptyset$$

A formal theory with the set of specific axioms $SA$ is denoted by $T(SA)$ and defined as follows

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

The set of all expressions of the language $\mathcal{L}_{SA}$ provable from the set specific axioms $SA$ (and logical axioms $LA$) i.e. the set

$$T(SA) = \{ A \in \mathcal{E} : SA \vdash_S A \}$$

is called the set of all theorems of the theory $T(SA)$
Soundness of the Theory

**Soundness Theorem** for a formal theory $T(SA)$ based on a proof system $S$ says:

For any formula $A$ of the language $L_{SA}$ of the theory $T(SA)$, if a formula $A$ is **provable** in the theory $T(SA)$, then $A$ is **true** in any **model** of the set of specific axioms $SA$ of $T(SA)$.
Syntactic Completeness of Formal Theories

The **Completeness Theorem** for the proof system $S$ established equivalency of the notion of provability and tautology:

$$ P_S = T_M $$

**Observe** the equation $P_S = T_M$ holds for a theory $T(SA)$ only when the set of its specific axioms $SA = \emptyset$

We nevertheless talk about **Complete/Incomplete** theories as the **final goal** of the course (and the book) is to prove the **Gödel Incompleteness Theorem** for the **Peano formal theory** of the **Arithmetic of Natural Numbers**
Complete Formal Theory

Definition

A formal theory $T(SA)$ based on a language with negation $\neg$ is complete if and only if for any $A$ of the language of the theory $T(SA)$ the following holds

$$A \in T(SA) \text{ or } \neg A \in T(SA)$$

Otherwise a theory $T(SA)$ is incomplete

The completeness of a theory means that we can prove or disapprove any statement formulated within it

It hence corresponds to the natural meaning of the notion of a complete information
Syntactic Consistence

Definition
A formal theory $T(SA)$ based on a language with negation $\neg$ is consistent if and only if there is no formula $A$ of the language of the theory $T(SA)$ such that $A \in T(SA)$ and $\neg A \in T(SA)$.

Otherwise $T(SA)$ is inconsistent.

The notions of consistency, inconsistency and completeness, incompleteness describe are the most important properties of any theory.
PART 5: Decidable and Syntactically Decidable Proof Systems
Decidable and Syntactically Decidable Proof Systems

A proof system $S$ is called **decidable** when there is a finite, mechanical method for determining, given any expression $A \in \mathcal{E}$ whether there is a proof of $A$ in $S$; i.e. whether $A \in \mathcal{P}_S$ otherwise $S$ is called **undecidable**

**Observe** that the above notion of decidability of the system does not require to find a proof. It requires only a mechanical procedure of deciding whether a proof exists for any expression of the system.
Example

We prove now that a Hilbert style proof system $S$ for classical propositional logic presented in Chapter 9 is decidable. We first prove the Completeness Theorem for it

$$P_S = T_M$$

We get that for any $A \in \mathcal{E}$

$$A \notin P_S \iff A \notin T_M$$

We have proved already that the notion of classical propositional tautology, i.e. the statement $A \notin T_M$ is decidable.

We conclude: the system $S$ is decidable.
Syntactically Decidable Systems

A proof system $S$ is **syntactically decidable** if it is possible to define for it a **finite, mechanical method** that generates a **proof** for any given expression $A$ of $S$. Otherwise, the system $S$ is **not syntactically decidable**. We call such syntactically decidable systems **automated theorem proving** systems.
Syntactically Decidable Systems

All Gentzen type proof systems presented here are both decidable or semi-decidable and syntactically decidable or syntactically semi-decidable.

We usually call them **automated theorem proving** systems for different logics under consideration.

Resolution based proof systems are also wildly known examples of the syntactically decidable, or semi-decidable systems.

**Finding** a Gentzen Type, or Resolution type formalization for a given logic is a **standard question** one asks about any logic being developed.
Formal Proofs

**Remember** that the notion of a formal proof in a system $S$ is purely syntactical in its nature.

Formal Proof carries a semantical meaning via established semantics and the Soundness Theorem.

The rules of inference of a proof system define only how to transform strings of symbols of the language into another string of symbols.

The formal proof, by the definition says that in order to prove an expression $A$ in a system $S$ one has to construct of a sequence of proper transformations, defined by the rules of inference.
Consider a very simple proof system $S_1$ with $\mathcal{E} = \mathcal{F}$

$$S_1 = (L_{\{P, \Rightarrow\}}, \mathcal{F}, LA1 = \{(A \Rightarrow A)\}, (r) \frac{B}{PB}),$$

where $A, B \in \mathcal{F}$ are any formulas and $P$ 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 **axiom schema**
Simple System $S_2$

Consider now a system $S_2$

$$S_2 = (\mathcal{L}_{\{P, \Rightarrow\}}, \mathcal{F}, \quad LA2 = \{(a \Rightarrow a)\}, (r) \frac{B}{PB})$$

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

Observe that even the system $S_1$ has only one axiom, it is also an **axiom schema**

Observe that 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 the system $S_2$
Some Provable Formulas

We have that

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

because

$$((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c))) \in LA1$$

other provable formulas are

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

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

Formal proofs in both systems of above formulas are identical and are as follows.

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

$A_1 = (a \Rightarrow a)$, $A_2 = P(a \Rightarrow a)$

axiom rule application

for $B = (a \Rightarrow a)$
Formal Proofs

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

\[
\begin{align*}
A_1 &= (a \Rightarrow a), \\
A_2 &= P(a \Rightarrow a), \\
A_3 &= PP(a \Rightarrow a)
\end{align*}
\]

axiom \quad \text{rule application} \quad \text{rule application}

for \( B = (a \Rightarrow a) \) \quad \text{for} \quad B = P(a \Rightarrow a)
Let’s search for a proof (if exists) of the formula $A$ below in $S_2$

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

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

This formula, in turn, if it had the proof, the only last step in its proof would be the application of the rule $r$ to the formula $((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c)))$

The search process stops here
Proof Search

Observe that

$$(Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c)) \notin LA2$$

what means that our search for the proof has failed; i.e. our found sequence of formulas does not constitute a proof

Moreover, the search was, at each step unique what proves that the proof of $A$ in $S_2$ does not exist, i.e.

$$\kappa_{S_2} PP((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c)))$$
Proof Search Procedure

We easily generalize above example to a proof search procedure to any formula $A$ of $S1$ or $S2$ 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

The Procedure SP is a finite, effective, automatic procedure of searching for a proof of formulas in both our proof systems. This proves the following.

Fact Proof systems $S_1$ and $S_2$ are syntactically decidable
Remark that we haven’t defined a semantics for the language $L_{\{\Rightarrow, P\}}$ of systems $S_1, S_2$.
We can’t talk about the soundness of these systems yet but we can think how to define a sound semantics for our systems.
If we want to understand statement PA as ”A is possible” we need to define some kind of modal semantics.
Semantical link

All known modal semantics extend the classical semantics, i.e. they are the same as classical one on non-modal connectives.

Hence under any possible modal semantics axioms $S1, S2$ of would be a sound axiom under standard modal logics semantics, as they are classical tautologies.

To assure the soundness of both systems we must have a modal semantics $M$ that makes the rule

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

sound under the modal semantics $M$. 
General Question 1

General Q1: Are all proof systems decidable?
Answer Q1: No, not all proof systems are decidable.

The most "natural" and historically first developed proof system for classical predicate logic is not decidable.
General Question 2

**General Q2**  Can we give an example of a logic and its complete proof system which is not decidable, but the logic does have another complete, syntactically decidable proof system?

**Answer Q2:**  Hilbert style proof system for classical propositional logic presented in chapter 5 is complete and decidable but is not syntactically decidable.

We present in chapter 6 some complete proof systems for classical propositional logic that are syntactically decidable.