Chapter 7: Proof Systems: Soundness and Completeness
Introduction

Proof systems are built to prove statements.

Proof systems are an *inference machine* with special statements, called *provable statements* being its final products.

The starting points of the inference are called *axioms of the system*.

We distinguish two kinds of axioms: *logical AL* and *specific SP*.

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

First step : we choose as a set of logical axioms AL some subset of tautologies, i.e. statements always true.

A proof system with only logical axioms AL is called *a logic proof system*.

Building a proof system for which there is no known semantics we think about the logical axioms as statements universally true.

We choose as axioms a finite set the statements we for sure want to be universally true, and whatever semantics follows they must be tautologies with respect to it.

Logical axioms are hence not only tautologies under an established semantics, but they also guide us how to establish a semantics, when it is yet unknown.

The specific axioms SP are these formulas of the language that describe our knowledge of a universe we want to prove facts about.

Specific axioms are not universally true, they are true only in the universe we are interested to describe and investigate.

A proof system with logical axioms AL and specific axioms SP is called *a formal theory*.

The inference machine is defined by a finite set of rules, called *inference rules*.

The inference rules describe the way we are allowed to transform the information within the system with axioms as a staring point.

The transformation process is called *a formal proof* and can be depicted as follows:

$$\boxed{\text{AXIOMS}}$$

$$\downarrow \ \downarrow \ \downarrow$$

$$\boxed{\text{RULES applied to AXIOMS}}$$

$$\downarrow \ \downarrow \ \downarrow$$

$$\boxed{\text{Provable formulas}}$$

$$\downarrow \ \downarrow \ \downarrow$$

$$\boxed{\text{RULES applied to any expressions above}}$$

$$\downarrow \ \downarrow \ \downarrow$$

$$\boxed{\text{NEW Provable formulas}}$$

$$\downarrow \ \downarrow \ \downarrow$$

. . . . etc. . . . . . .

The provable formulas are those for which we have a formal proof are called *consequences* of the axioms.

Semantical link : the rules have to preserve the truthfulness of what we are proving.

Rules with this property are called *sound rules* and the system *a sound proof system*.

Soundness Theorem : for any formula A of the language of the system S,

If a formula A is provable in a logic proof system S, then A is a tautology.

Formal theory with specific axioms SP, based on a logic defined by the axioms AL is a proof system S with logical axioms AL and specific axioms SP.

Notation : $TH_S(SP)$.

Soundness Theorem for formal theory says: for any formula A of the language of the theory $TH_S(SP)$,

If a formula A is provable in the theory $TH_S(SP)$, then A is true in any model of the set of specific axioms SP.

We discuss here only logic proof systems and call them proof systems for different logics.

Any proof system can be sound under one semantics, and not sound under the other.

For example a set of axioms and rules sound under classical logic semantics might not be sound under Ł logic semantics, or K logic semantics, or others.

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

Given a logic system S with logical axioms AL that is sound under a given semantics M. Let $\mathbf{T}_M$ be a set of all tautologies defined by the semantics M, i.e.

$$\mathbf{T}_M = \{A : \quad \models_M A\}.$$

A natural questions arises : are all tautologies defined by the semantics M, provable in the system S that is sound under the semantics M.

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

Completeness Theorem for a logic system S, under a semantics M. For any A (of the language of S),

A is provable in S iff   A is a tautology under the semantics M.

Symbolically :

$$\vdash_S A \quad iff \quad \models_M A.$$

Completeness Theorem is composed from two parts: the **Soundness Theorem** and the *completeness part* that proves the completeness property of a sound system.

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

We first prove that all logical axioms are tautologies, and then that all inference rules of the system preserve the notion of the truth (model).

Proving the *completeness part* of the **Completeness Theorem** is always a very difficult, and a crucial task.

We will study two proofs of the Completeness Theorem for classical propositional Hilbert style proof system in the next chapter, and a constructive proofs for automated theorem proving systems for classical logic the the following chapter.

Predicate Logics case will be discussed in the second part of the book.

Formal Definitions : In this section we formulate a general definition of a proof system, formal proof, set of consequences and give simple examples of different proof systems.

Language $\mathcal{L}$ of S

Usually, as in the propositional case, the language $\mathcal{L}$ consists of its alphabet $\mathcal{A}$ and a set of formulas, denoted here by $\mathcal{F}$, i.e.

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

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

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

For example sometimes we consider the set of all finite sequences of formulas, or sets of formulas, or other expressions, called Gentzen sequents, or sets of clauses in the case of the resolution based systems as the basic *expressions* of our proof system S under consideration.

Expressions $\mathcal{E}$

We assume that $\mathcal{E}$ is enumerable and primitively recursive i.e. that there is an effective procedure to determine whether a given expression is in $\mathcal{E}$.

Semantical Equivalency We always have to prove prove a semantic equivalency of $\mathcal{E}$ and the set of all formulas $\mathcal{F}$ of $\mathcal{L}$.

It means that we prove that for a given semantics M under which we build our proof system S. This is also called an *extension* of the semantics M for $\mathcal{L}$ to the set of expressions $\mathcal{E}$.

Example In automated theorem proving system **RS** we study later, our basic expressions are *finite sequences of formulas* of $\mathcal{L} = \mathcal{L}_{\neg, \cap, \cup, \Rightarrow}$.

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

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

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

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

i.e.

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

Axioms AL of S

We assume that the axioms AL of S form a
   proper, non-empty subset of the set $\mathcal{E}$ of
   expressions of our language.

The set AL is primitively recursive i.e. there
   is an effective procedure to determine whether
   a given expression $A \in \mathcal{E}$ is in AX or not.

Axioms AL called logical axioms of S.

Semantical link : For a given semantics M
   for $\mathcal{L}$ and $\mathcal{E}$, AL is always a subset of ex-
   pressions that are tautologies of under the
   semantics M.

Rules of inference $\mathcal{R}$ We assume that a proof system contains only a finite number of inference rules.

Each rule has a finite number of premisses and one conclusion.

We also assume that one can effectively decide, for any rule, whether a given string of expressions form its premisses and conclusion or not, i.e. that

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

Formally : each $r \in \mathcal{R}$ is a relation defined in $\mathcal{E}^m$ with values in $\mathcal{E}$, i.e.

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

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

We write the inference rules in a following convenient way.

One premiss rule :

$$(r) \quad \frac{A}{B},$$

Two premisses rule :

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

Three premisses rule :

$$(r) \quad \frac{P_1 \quad ; \quad P_2 \quad ; \quad P_3}{A},$$

m-premisses rule :

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

Semantical link : For a given semantic M for $\mathcal{L}$, and $\mathcal{E}$, we chose for S rules which preserve truthfulness under the semantics M, i.e. we prove that the rules of the system S are *sound*.

Proof system S

By a (logic) proof system we understand a triple

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

Language : $\mathcal{L} = \{\mathcal{A}, \mathcal{F}\}$ is a formal language, called the language of S with a set $\mathcal{F}$ of formulas.

Expressions : $\mathcal{E}$ is a set of expressions of $\mathcal{L}$.

Logical Axioms : AL is a non-empty, proper, primitively recursive subset of the set of expressions $\mathcal{E}$.

Rules of inference : $\mathcal{R}$ is a finite set of rules of inference and all $r \in \mathcal{R}$ are primitively recursive relations.

Provable expressions of a system S are final product of single or multiple use of the inference rules, with axioms taken as a starting point.

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

A multiple application of rules of inference is called a proof.

Formal definitions follow.

Direct consequence For any rule of inference $r \in \mathcal{R}$, if

$$(P_1, ... P_n, A) \in r,$$

then A is called a direct consequence of $P_1, ... P_n$ by virtue of r.

Proof of A in S is a sequence

$$A_1, ... A_n$$

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

$$A_1 \in AL, \quad A_n = A$$

and for each i, $1 \leq i \leq n$, either A_i is an axiom of the system or A_i is a direct consequence of some of the preceding expressions by virtue of one of the rules of inference.

Notation :

$$\vdash_S A$$

denotes that A has a proof in S.

When the proof system S is fixed we write

$$\vdash A.$$

Provable expressions of S

Any $A \in \mathcal{E}$, such that

$$\vdash_{\mathcal{S}} A$$

is called a provable expression of S.

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

$$\mathbf{PR}_S = \{A \in \mathcal{E} : \quad \vdash_{\mathcal{S}} \mathcal{A}\}.$$

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

Proof of A **from** Γ **in** S is a proof where the expressions from Γ are added to the set AL of the axioms of the system.

Notation :

$$\Gamma \vdash_S A.$$

Hypothesis

If $\Gamma \vdash_S A$ then the expressions of Γ are called hypotheses of the proof of A from Γ.

Finite Γ

If Γ is a finite set and $\Gamma = \{B_1, B_2, ..., B_n\}$, then we write

$$B_1, B_2, ..., B_n \vdash_\mathcal{S} A$$

instead of $\{B_1, B_2, ..., B_n\} \vdash_\mathcal{S} A$.

Empty Γ

The case of $\Gamma = \emptyset$ is a special one. By the definition of a proof of A from Γ, $\emptyset \vdash A$ means that in the proof of A only axioms AL of S were used. We hence write

$$\vdash_S A$$

to denote that A has a proof from empty Γ.

Consequence of Γ

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

Consequence operation in S is a function $\mathbf{Cn}_S$
which to every set $\Gamma \subset \mathcal{E}$, assigns a set of
all its consequences. I.e.

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

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

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

is called the consequence operation in S.

Properties of consequence operation $\mathbf{Cn}_S$.

Monotonicity

For any sets Γ, Δ of expressions of S,

if $\Gamma \subset \Delta$ then $\mathbf{Cn}_S(\Gamma) \subset \mathbf{Cn}_S(\Delta)$.

Transitivity

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

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

Finiteness

For any expression $A \in \mathcal{F}$ and any set $\Gamma \subset \mathcal{F}$,

$A \in \mathbf{Cn}_S(\Gamma)$ if and only if there is a finite subset Γ_0 of Γ such that $A \in \mathbf{Cn}_S(\Gamma_0)$.

Decidable system is a proof system, for which there is a mechanical method for determining, given any expression A of the system, whether there is a proof of A; otherwise it 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 *there is* a proof of any expression of the system.

Example : A Hilbert style proof system for classical propositional logic decidable, but not syntactically, or automatically decidable proof system.

We conclude its decidability from the Completeness Theorem and the decidability of the notion of classical tautology.

Syntactically decidable system is a proof system S, for which there is a mechanical method for determining, given any expression A of the system, not only whether there is a proof of A, but which also generates a proof; otherwise S is not not syntactically decidable.

Other names : *automatically decidable*, or *an automated system*

Example : Gentzen proof system, the **RS** system presented here later, and Resolution style proof systems for classical propositional logic are both decidable and syntactically decidable proof systems, and we call them always *automated proof systems*.

The notion of a proof in a system S usually carries a semantical meaning via the *Soundness Theorem* but it is nevertheless purely *syntactical* in its nature.

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

The definition of a formal proof says that in order to prove an expression A of a system one has to construct of s sequence of proper transformations, defined by the rules of inference.

Example: System S_1

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

where A, B are any formulas of the propositional language $\mathcal{L}_{\{P,\Rightarrow\}}$.

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

We call such an axiom *axiom schema*.

Example: System S_2

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

where $a \in VAR$ and B is any formulas of the propositional language $\mathcal{L}_{\{P, \Rightarrow\}}$.

Example :

$$((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c))) \in AL(S1)$$

But is not an axiom of the system S_2.

Provable formulas :

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

Proof :

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

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

$$\vdash_{S_1} PP(a \Rightarrow a), \ \vdash_{S_1} PP(a \Rightarrow a).$$

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:

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

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

$(a \Rightarrow a),$ $P(a \Rightarrow a),$ $PP(a \Rightarrow a).$
   axiom   rule application   rule application
            for $B = (a \Rightarrow a)$ for $B = P(a \Rightarrow a)$

More formulas :

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

$$\nvdash_{S_1} PPP((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c))).$$

Formal proof of
$$PPP((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c)))$$
in S_1 is:

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

$P((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c))),$
rule r application

$PP((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c))),$
rule r application

$PPP((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c))).$
rule r application

Search for a proof of the formula

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

in S_2.

Observe, that if it had the proof, the only last step in this proof would be the application of the rule r to the formula

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

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

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

And again, this one could be obtained only from the formula

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

by the virtue of the rule r.

The search process stops here.

The formula :

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

isn't an axiom of S_2, what means that the only possible way of finding the proof has failed, i.e. we have proved that $\nvdash_{S_1} PPP((Pa \Rightarrow (b \Rightarrow c)) \Rightarrow (Pa \Rightarrow (b \Rightarrow c)))$.

We easily generalize above procedure to any formula of $S1$ or $S2$.

It is an effective, automatic procedure of searching for a proof of our formula in both our proof systems.

If the search ends with an axiom, we have a proof.

If it doesn't end with an axiom it means that the proof does not exist. We have described it, as an example, for one particular formula. It can be easily extended to any formula A of $\mathcal{L}_{\{P, \Rightarrow\}}$ by the re-iteration of the following steps.

We easily generalize above procedure to any formula of $S1$ or $S2$ as follows.

Step : Check the main connective of A.

If main connective is P, it means that A was obtained by the rule r.

Erase the main connective P.

Repeat until no P left.

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

If it is an axiom , STOP and YES, we have a proof.

If it is not an axiom , STOP and NO, proof does not exist.

It is an effective, automatic procedure of searching for a proof of our formula in both our proof systems. This proves the following.

Fact Proof systems S_1 and S_2 are syntactically decidable.

General form of provable formulas:

$$PR_{S_1} = \{P^n(A \Rightarrow A) : n \in N, A \in \mathcal{F}\},$$

$$PR_{S_2} = \{P^n(a \Rightarrow a) : n \in N, a \in VAR\}.$$

Fact :

$$PR_{S_1} \neq PR_{S_2}, \quad PR_{S_2} \subset PR_{S_1}.$$

Semantical link : We haven't defined a semantics for the language $\mathcal{L}_{\{\Rightarrow,P\}}$ of systems $S1, S2$, so we can't talk about the soundness of these systems yet.

All known modal semantics *extend the classical semantics*, i.e. are the same as the classical one on non-modal connectives so the axiom in both cases would be a sound axiom under standard modal logics semantics.

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

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

is sound, i.e. such that

$$\models_M (B \Rightarrow PB).$$

Otherwise they will not be sound.

Example: Systems S_3 and S_4.

$$S_3 = (\mathcal{L}_{\{\cup,\neg\}}, \quad \mathcal{F}, \ \{(A \cup \neg A)\},$$

$$\frac{B}{(B \cup (A \cup \neg A))},$$

for any $A, \ B \ \in \mathcal{F}$.

$$S_4 = (\mathcal{L}_{\{\cup,\neg\}}, \ \mathcal{F}, \quad \{(A \cup \neg A)\},$$

$$\frac{(A \cup \neg A)}{(B \cup (A \cup \neg A))},$$

for any $A, B \in \mathcal{F}$.

Question 1: describe the sets of provable formulas of S_3 and S_4.

Question 2: do they produce the same sets of provable formulas? I.e. is it *true/ false* that

$$\{A : \vdash_{S_3} A\} = \{A : \vdash_{S_4} A\}.$$

If yes, prove it.

if not true, give a formula which is a theorem of one system and is not a theorem of other system.

Question 3: Are the systems S_3 and S_4 decidable, syntactically decidable?

Answer Question 1: We first describe the set of provable formulas of both systems.

System S_3 Obviously, $\vdash_{S_3} (A \cup \neg A)$.

One application of the inference rule gives us the proof of $((A \cup \neg A) \cup (A \cup \neg A))$.

The next application of the rule (to the already produced formula) will give us the proof of $(((A \cup \neg A) \cup (A \cup \neg A)) \cup (A \cup \neg A))$.

It is easy to see that all provable formulas of S_3 will be of the form of the disjunction of the axiom of S_3, i.e.

$$PR_{S_3} = \{\textstyle\bigcup_n (A \cup \neg A)^n : n \in N, A \in \mathcal{F}\},$$

where $\bigcup_n (A \cup \neg A)^n$ denotes a disjunction of n formulas of the form $(A \cup \neg A)$.

System S_4 Obviously, as before, $\vdash_{S_4}(A \cup \neg A)$.

One application of the inference rule gives us the proof of $(B \cup (A \cup \neg A))$, where B is a certain formula from $\mathcal{F}$.

Observe that The rule r can't be, by its definition, applied to already produced formula (except the axiom).

Multiple application of the rule means multiple application to the axiom and producing all possible formulas of the form $(B \cup (A \cup \neg A))$, where B's are different for different applications.

Provable formulas of $S4$:

$$PR_{S_4} = \{(B \cup (A \cup \neg A)) : A, B \in \mathcal{F}\}$$

$$\cup \{(A \cup \neg A) : A \in \mathcal{F}\}.$$

Provable formulas of S_3 and S_4. Obviously, $PR_{S_3} \subseteq PR_{S_4}$ as we have, by definition, that

$$\bigcup_n (A \cup \neg A)^n = \bigcup_{n-1} (A \cup \neg A)^{n-1} \cup (A \cup \neg A),$$

Denote: $\bigcup_{n-1} (A \cup \neg A)^{n-1} = B$

Provable formulas of S_3 are axioms or have a form

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

for a certain $B \in \mathcal{F}$.

Answer to Question 2:

$$\vdash_{S_4}((a \cup \neg b) \cup (a \cup \neg a)),$$

but obviously

$$\nvdash_{S_3}((a \cup \neg b) \cup (a \cup \neg a)).$$

The above proves that

$$PR_{S_3} \subseteq PR_{S_4} \quad and \quad PR_{S_3} \neq PR_{S_4}.$$

Answer Question 3 It follows immediately form the form of the sets PR_{S_3} and PR_{S_4} that both systems are syntactically decidable. The design of the proper proof searching procedure is left to the reader as an exercise.

Semantical link 1 : Both systems are *sound under classical semantics.*

It follows from the fact that

$$\models (A \cup \neg A),$$

and the rules

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

are sound because

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

Semantical link 2 : Both systems are *not sound under* **Ł, K, H, B** *semantics.*

It follows from the fact that

$$\not\models_M (A \cup \neg A)$$

for M = **Ł, K, H, B**.

General Q1 Are all proof systems decidable?

Answer Q1 : No, not all the systems are decidable. The most "natural" and historically first developed proof system for classical predicate logic is not decidable.

General Q2 Can we give an example of a proof system for a given logic which is not decidable, but the logic does have (another) syntactically decidable system?

Answer Q2 : Hilbert style proof system for classical logic presented in the next chapter is decidable, but not syntactically decidable.

RS system for classical propositional logic, described in chapter 9 is syntactically decidable.