

Soundness Theorem

We connect “proof” and “truth” through the Soundness Theorem, which states that every provable formula is true.

1. The Soundness Theorem

1.1. Statement

Let $\mathcal{L}$ be a fixed first-order language.

Theorem 1 (Soundness Theorem).

$$\boxed{\text{If } \mathcal{P}_1, \dots, \mathcal{P}_l \vdash \mathcal{Q} \text{ then } \mathcal{P}_1, \dots, \mathcal{P}_l \models \mathcal{Q}.}$$

More generally, if Γ is a (possibly infinite) set of formulas:

$$\text{If } \Gamma \vdash \mathcal{Q} \text{ then } \Gamma \models \mathcal{Q}.$$

This is a common-sense theorem: a proven formula is a true formula! This result is implicit in Gödel’s 1930 thesis, in which he focuses on the converse (the Completeness Theorem). The idea is that the inference rules for proofs were well chosen, since each rule preserves the true/false nature of first-order logic.

1.2. Review

Recall that

$$\mathcal{P}_1, \dots, \mathcal{P}_l \vdash \mathcal{Q}$$

means that $\mathcal{Q}$ can be deduced from the formulas $\mathcal{P}_1, \dots, \mathcal{P}_l$ in finitely many steps, each of which consists in applying an inference rule (from a finite, precisely specified collection).

Also recall that

$$\mathcal{P}_1, \dots, \mathcal{P}_l \models \mathcal{Q}$$

means that, for every structure $\mathcal{S}$ and every assignment s , if $\mathcal{P}_1, \dots, \mathcal{P}_l$ are true under this interpretation, then $\mathcal{Q}$ is also true under this interpretation.

Thus, the Soundness Theorem states that a provable formula is true under every interpretation, independently of the chosen structure.

It can also be restated as follows: if $\mathcal{P} \vdash \perp$, then $\mathcal{P}$ is unsatisfiable (that is, $\mathcal{P}$ is a contradiction); in other words, regardless of $\mathcal{S}$ and s , the interpretation of $\mathcal{P}$ is false. Recall that $\perp$ denotes the formula that is always false. Here is a proof of this restatement: suppose there are $\mathcal{S}$ and s for which $\mathcal{P}$ is true. Since $\perp$ is false under every interpretation, this contradicts $\mathcal{P} \models_{\mathcal{S},s} \perp$. Thus, $\mathcal{P}$ is not satisfiable.

1.3. Applications

Let us explain the usefulness of the Soundness Theorem.

1. In one sense, $\mathcal{P} \vdash \mathcal{Q}$ is easy. Indeed, the task is to construct a proof. Although finding a proof is by no means always easy, it is still a process with finitely many steps and finitely many possible rules.
2. By contrast, showing that no proof establishes that $\mathcal{P}$ implies $\mathcal{Q}$, that is, that $\mathcal{P} \not\vdash \mathcal{Q}$, is difficult. Failing to find a proof does not mean that none exists.
3. On the other hand, showing that $\mathcal{P} \models \mathcal{Q}$ is not straightforward, because this would require checking infinitely many conceivable structures and assignments.
4. Conversely, establishing $\mathcal{P} \not\models \mathcal{Q}$ is easy. It is enough to provide a counterexample by specifying a structure and an assignment such that $\mathcal{P}$ is true and $\mathcal{Q}$ is false under this interpretation.

In summary, the Soundness Theorem can help with the difficult cases, either directly or through its contrapositive:

$$\begin{array}{ll} \text{If } \mathcal{P} \vdash \mathcal{Q} \text{ then } \mathcal{P} \models \mathcal{Q}. \\ \text{If } \mathcal{P} \not\vdash \mathcal{Q} \text{ then } \mathcal{P} \not\models \mathcal{Q}. \end{array}$$

2. Proof of the Soundness Theorem

We are going to prove that each inference rule preserves true/false logic and the satisfaction of formulas. Since a proof is a finite sequence of applications of these rules, the Soundness Theorem will then follow by induction.

2.1. Rules of true/false logic

See the chapter “Proofs” in Part One for the complete list of these rules. Here we give only a few examples.

- We have the rule “ $\mathcal{P} \wedge \mathcal{Q} \vdash \mathcal{P}$ ”, and indeed the logical consequence “ $\mathcal{P} \wedge \mathcal{Q} \models \mathcal{P}$ ” holds.
- We have the rule “ $\mathcal{P}, \mathcal{Q} \vdash \mathcal{P} \wedge \mathcal{Q}$ ”, and indeed the logical consequence “ $\mathcal{P}, \mathcal{Q} \models \mathcal{P} \wedge \mathcal{Q}$ ” holds.
- We have the rule “ $\neg\neg\mathcal{P} \vdash \mathcal{P}$ ”, and indeed the logical consequence “ $\neg\neg\mathcal{P} \models \mathcal{P}$ ” holds.
- We have the rule “ $\perp \vdash \mathcal{P}$ ”, and indeed the logical consequence “ $\perp \models \mathcal{P}$ ” holds.
- We have the rule “ $\mathcal{P}, \mathcal{P} \rightarrow \mathcal{Q} \vdash \mathcal{Q}$ ”, and indeed the logical consequence “ $\mathcal{P}, \mathcal{P} \rightarrow \mathcal{Q} \models \mathcal{Q}$ ” holds.

These logical consequences are easily justified using truth tables. For example, to prove $\mathcal{P} \wedge \mathcal{Q} \models \mathcal{P}$, fix a structure $\mathcal{S}$ and an assignment s , and draw up the truth table:

$\mathcal{P}$	$\mathcal{Q}$	$\mathcal{P} \wedge \mathcal{Q}$	$\mathcal{P}$
T	T	T	T
T	F	F	—
F	T	F	—
F	F	F	—

Whenever $\mathcal{P} \wedge \mathcal{Q}$ is true, $\mathcal{P}$ is also true. Thus,

$$\mathcal{P} \wedge \mathcal{Q} \models_{\mathcal{S},s} \mathcal{P}$$

and since this is true for all $\mathcal{S}$ and s :

$$\mathcal{P} \wedge \mathcal{Q} \models \mathcal{P}$$

The same can be done for *modus ponens*:

$\mathcal{P}$	$\mathcal{Q}$	$\mathcal{P} \rightarrow \mathcal{Q}$	$\mathcal{Q}$
T	T	T	T
T	F	F	—
F	T	T	—
F	F	T	—

Whenever $\mathcal{P}$ is true and $\mathcal{P} \rightarrow \mathcal{Q}$ is also true, $\mathcal{Q}$ is true as well.

2.2. Rules for quantifiers

The rules involving quantifiers are also compatible with satisfaction. This again follows from the fact that the inference rules were chosen sensibly. However, establishing their validity involves manipulations that are not always easy to set out in detail. On a first reading, you may skip this step and go directly to the end of the proof. Studying these proofs, however, helps clarify the conditions under which the inference rules may be applied.

Recall the definitions and notions from the chapter “Satisfaction”.

$$\models_{\mathcal{S},s} \forall x \mathcal{P}(x) \quad \text{iff} \quad \text{for every } d \in E, \models_{\mathcal{S},s[x \mapsto d]} \mathcal{P}(x)$$

The notation $s[x \mapsto d]$ denotes the assignment that maps x to d and agrees with s on all other variables.

$$\models_{\mathcal{S},s} \exists x \mathcal{P}(x) \quad \text{iff} \quad \text{there exists } d \in E, \models_{\mathcal{S},s[x \mapsto d]} \mathcal{P}(x)$$

Recall the relationship between assignments and substitution:

$$\models_{\mathcal{S},s} \mathcal{P}[c/x] \quad \text{iff} \quad \models_{\mathcal{S},s[x \mapsto c^{\mathcal{S}}]} \mathcal{P}(x)$$

Finally, for a formula $\mathcal{P}$, we write $\mathcal{P}(x)$ to emphasize its dependence on the variable x , and we write:

$$\mathcal{P}(c) = \mathcal{P}[c/x]$$

Writing $\mathcal{P}(c)$ presupposes that substituting c for x is valid.

We begin with the two simple rules.

Rule I. We have the rule

$$\forall x \mathcal{P}(x) \vdash \mathcal{P}(c)$$

where c is a constant. And indeed, the corresponding logical consequence holds:

$$\forall x \mathcal{P}(x) \models \mathcal{P}(c)$$

Proof. Consider a structure $\mathcal{S}$ with domain E and an assignment s such that “ $\forall x \mathcal{P}(x)$ ” is true. By definition, this means that $\mathcal{P}(d)$ is true for every d in the domain E . In particular, $\mathcal{P}(c)$ is therefore satisfied.

Let us now give the formal proof. Let c be a constant of the language, and let $c^{\mathcal{S}}$ be its interpretation in E .

$$\begin{array}{ll} & \models_{\mathcal{S},s} \forall x \mathcal{P}(x) \\ \text{therefore} & \models_{\mathcal{S},s[x \mapsto d]} \mathcal{P}(x) \quad \text{for every } d \in E \\ \text{in particular} & \models_{\mathcal{S},s[x \mapsto c^{\mathcal{S}}]} \mathcal{P}(x) \\ \text{therefore} & \models_{\mathcal{S},s} \mathcal{P}[c/x] \\ \text{hence} & \models_{\mathcal{S},s} \mathcal{P}(c) \end{array}$$

□

Rule III. We have the rule

$$\mathcal{P}(c) \vdash \exists x \mathcal{P}(x)$$

where c is a constant, and indeed the corresponding logical consequence holds:

$$\mathcal{P}(c) \models \exists x \mathcal{P}(x)$$

Proof. Fix S with domain E , and fix s . If $\mathcal{P}(c)$ is true, then there is a $d \in E$ (namely, $d = c^S$) such that $\mathcal{P}(d)$ is true; therefore, “ $\exists x \mathcal{P}(x)$ ” is true.

Let us now give the formal proof. Let c be a constant of the language, and let c^S be its interpretation in E .

$$\begin{array}{ll} & \models_{S,s} \mathcal{P}(c) \\ \text{therefore} & \models_{S,s} \mathcal{P}[c/x] \\ \text{therefore} & \models_{S,s[x \mapsto c^S]} \mathcal{P}(x) \\ \text{hence} & \models_{S,s} \exists x \mathcal{P}(x) \quad \text{since the formula is true for } d = c^S \end{array}$$

□

Rule II. We restate this rule as follows. Let Γ be a set of formulas, and let c be a constant that does not occur in the formulas of Γ . Rule II is written:

$$\text{If } \Gamma \vdash \mathcal{P}(c) \quad \text{then} \quad \Gamma \vdash \forall x \mathcal{P}(x)$$

And indeed there is a corresponding statement in terms of satisfaction:

$$\text{If } \Gamma \models \mathcal{P}(c) \quad \text{then} \quad \Gamma \models \forall x \mathcal{P}(x)$$

Proof. We have a language $\mathcal{L}$ with a constant c . Suppose $\Gamma \models \mathcal{P}(c)$. Fix a structure S with domain E and an assignment s such that Γ is satisfied. Intuitively, we must show that $\mathcal{P}(d)$ is satisfied for every $d \in E$. Formally, we must prove that $\models_{S,s[x \mapsto d]} \mathcal{P}(x)$ for every $d \in E$.

So far, we have a structure S with domain E ; in this structure, the constant c of the language is interpreted as a value $c^S \in E$. The assumptions then imply that $\mathcal{P}(c^S)$ is satisfied. We have therefore verified $\mathcal{P}(x)$ for one value in the domain, but we want to prove it for every value.

Fix an arbitrary value $d \in E$ in the domain. We consider another structure S' with the same domain E . The structure S' is the same as S , except that the interpretation of c in this structure is our fixed value d : $c^{S'} = d$. We do not change the assignment s . Since the formulas of Γ do not depend on the constant c , and since Γ is satisfied under S and s , it is also satisfied under S' and s .

The assumption $\Gamma \models \mathcal{P}(c)$ expresses satisfaction for every structure, so in particular we have

$$\models_{S',s} \mathcal{P}(c)$$

Since Γ is satisfied under S' and s , we have:

$$\begin{array}{ll} & \models_{S',s} \mathcal{P}(c) \\ \text{therefore} & \models_{S',s} \mathcal{P}[c/x] \\ \text{therefore} & \models_{S',s[x \mapsto c^{S'}]} \mathcal{P}(x) \\ \text{therefore} & \models_{S',s[x \mapsto d]} \mathcal{P}(x) \\ \text{hence} & \models_{S,s[x \mapsto d]} \mathcal{P}(x) \end{array}$$

Note that in the last line we returned to the structure S . This is possible because the constant c does not occur in the formula $\mathcal{P}(x)$ (indeed, writing $\mathcal{P}(c)$ requires the substitution of c for x to be possible, and hence c not to occur in $\mathcal{P}(x)$).

Thus, for every $d \in E$, we have obtained:

$$\models_{S,s[x \mapsto d]} \mathcal{P}(x)$$

This is precisely the definition of:

$$\models_{S,s} \forall x \mathcal{P}(x)$$

□

Rule IV. We restate this rule as follows. Let $\mathcal{Q}$ denote a formula, let Γ be a set of formulas, and let c be a constant of the language that occurs neither in $\mathcal{Q}$ nor in the formulas of Γ . Rule IV is written:

$$\text{If } \Gamma \vdash \exists x \mathcal{P}(x) \text{ and } \mathcal{P}(c) \vdash \mathcal{Q} \text{ then } \Gamma \vdash \mathcal{Q}$$

And indeed there is a corresponding statement in terms of satisfaction:

$$\text{If } \Gamma \models \exists x \mathcal{P}(x) \text{ and } \mathcal{P}(c) \models \mathcal{Q} \text{ then } \Gamma \models \mathcal{Q}$$

Proof. First consider a structure S with domain E and an assignment s such that Γ is satisfied. Then, by assumption, “ $\exists x \mathcal{P}(x)$ ” is satisfied, so there exists $d \in E$ such that $\mathcal{P}(x)$ is satisfied under the assignment $s[x \mapsto d]$.

Change the structure S into a structure S' in which only the interpretation of the constant c is changed: in S' , define $c^{S'} = d$. Since $\mathcal{Q}$ and the formulas of Γ do not depend on c , their satisfaction in S and in S' is equivalent. Thus, Γ is satisfied in S' , and $\mathcal{P}(c)$ is satisfied as well:

$$\begin{aligned} & \models_{S',s} \exists x \mathcal{P}(x) \\ \text{therefore } & \models_{S',s[x \mapsto d]} \mathcal{P}(x) \quad \text{for the same } d \in E \text{ as above} \\ \text{therefore } & \models_{S',s[x \mapsto c^{S'}]} \mathcal{P}(x) \quad \text{since } c^{S'} = d \\ \text{therefore } & \models_{S',s} \mathcal{P}[c/x] \\ \text{therefore } & \models_{S',s} \mathcal{P}(c) \\ \text{therefore } & \models_{S',s} \mathcal{Q} \quad \text{by the assumption } \mathcal{P}(c) \models \mathcal{Q} \\ \text{hence } & \models_{S,s} \mathcal{Q} \quad \text{since } c \text{ does not occur in } \mathcal{Q} \end{aligned}$$

□

2.3. Proof of the theorem

Let $\Gamma = \{\mathcal{P}_1, \dots, \mathcal{P}_l\}$ denote the set of formulas serving as assumptions, and let $\mathcal{Q}$ be the conclusion formula. Suppose $\Gamma \vdash \mathcal{Q}$. We must prove that $\Gamma \models \mathcal{Q}$.

The statement $\Gamma \vdash \mathcal{Q}$ means that there is a proof of $\mathcal{Q}$ from the formulas in Γ . Suppose this proof is written in natural deduction as in the figure below. The formulas $\mathcal{P}_1, \dots, \mathcal{P}_l$ are the assumptions. Each formula $\mathcal{P}_k$, with $l+1 \leq k \leq q$, is obtained by applying one of the inference rules to the preceding lines $\mathcal{P}_1, \dots, \mathcal{P}_{k-1}$. The final formula obtained, $\mathcal{P}_q$, is the formula $\mathcal{Q}$.

	$\mathcal{P}_1$	
	$\vdots$	Assumptions
	$\mathcal{P}_l$	
	$\mathcal{P}_{l+1}$	
	$\vdots$	
	$\mathcal{P}_k$	Proof steps
	$\vdots$	
	$\mathcal{P}_{q-1}$	
$\vdash$	$\mathcal{P}_q = \mathcal{Q}$	Conclusion

Fix a structure $\mathcal{S}$ and an assignment s such that Γ is satisfied. We must show that $\mathcal{Q}$ is then also satisfied (under $\mathcal{S}$ and s). We prove this by induction on k (for $1 \leq k \leq q$):

$\mathcal{H}_k$: the formula $\mathcal{P}_k$ is satisfied

- *Base case.* For $k = 1, \dots, l$, the formula $\mathcal{P}_k$ is one of the formulas in Γ , so it is satisfied by assumption.
- *Induction step.* Let k satisfy $1 \leq k < q$. Suppose $\mathcal{P}_1, \dots, \mathcal{P}_k$ are satisfied (under $\mathcal{S}$ and s), and show that $\mathcal{P}_{k+1}$ is also satisfied. Indeed, $\mathcal{P}_{k+1}$ is obtained from the preceding formulas by applying one of the inference rules for proofs. We have already shown that each of these rules preserves the satisfaction of formulas, so $\mathcal{P}_{k+1}$ is satisfied (under $\mathcal{S}$ and s).
- *Conclusion.* By the principle of induction, the final formula $\mathcal{P}_q$, which is $\mathcal{Q}$, is satisfied.

Thus, $\models_{\mathcal{S}, s} \mathcal{Q}$, and since $\mathcal{S}$ and s are arbitrary, $\Gamma \models \mathcal{Q}$.

Exercises

The Soundness Theorem

Exercise 1. Let Γ be a set of formulas such that $\Gamma \vdash \perp$. Show that Γ is a contradiction; that is, there is no structure $\mathcal{S}$ and no assignment s under which all the formulas in Γ are satisfied.

Exercise 2. Prove the unprovability of the following statements:

1. $\exists x P(x) \not\vdash \forall x P(x)$
2. $\forall x \exists y Q(x, y) \not\vdash \exists y \forall x Q(x, y)$
3. $\exists x (P(x) \rightarrow Q(x)) \not\vdash \exists x (Q(x) \rightarrow P(x))$
4. $\forall x (P(x) \vee Q(x)) \not\vdash (\forall x P(x)) \vee (\forall x Q(x))$

Exercise 3. Prove the following logical consequences:

1. $\forall x (P(x) \rightarrow Q(x)), P(c) \models Q(c)$
2. $\forall x (P(x) \rightarrow \perp) \models \neg(\exists x P(x))$
3. $\forall x (P(x) \leftrightarrow Q(x)), \forall x (Q(x) \leftrightarrow R(x)) \models \forall x (P(x) \leftrightarrow R(x))$
4. $\forall x (P(x) \wedge Q(x)) \models \forall x P(x)$

Proofs of the Soundness Theorem

Exercise 4. Verify that all the basic rules of true/false logic are compatible with satisfaction.