Incompleteness and Undecidability of Predicate Logic

Additional Course Notes for Semantiek en Logica 2

Herman Geuvers

January 8, 2007

1 Completeness and Hilbert's program

We have seen in S&L1 that predicate logic is sound and complete:

Theorem 1.1

Soundness *If* $\Gamma \vdash \varphi$, *then* $\Gamma \models \varphi$

Completeness *If* $\Gamma \models \varphi$, *then* $\Gamma \vdash \varphi$

We recall that $\vdash$ refers to *derivability* e.g. in the natural deduction system of predicate logic as given in [1]. We also recall that $\models$ refers to *truth in all models*: for all models (of the right signature) $\mathcal{M}$, if all formulas of Γ are true in $\mathcal{M}$, then φ is true in $\mathcal{M}$.

The completeness referred to above is also called *semantical completeness*. It relates derivability to truth in models.

There's also a notion of *syntactical completeness*, which refers to the issue whether a *theory* can decide upon all formulas. A *theory* is a set of closed formulas.

Definition 1.2 *A set of closed formulas T is called* syntactically complete, *if for each closed formula φ, either $T \vdash \varphi$ or $T \vdash \neg\varphi$.*

A particularly interesting theory is that of Peano arithmetic that describes the natural numbers, as it lays on the basis of many other mathematical theories.

Definition 1.3 *PA is the following theory (over the appropriate signature)*

$$\forall x(0 \neq S(x))$$
$$\forall x, y(S(x) = S(y) \rightarrow x = y)$$
$$\forall x(x + 0 = x)$$
$$\forall x, y(x + S(y) = S(x + y))$$
$$\forall x(x \cdot 0 = 0)$$
$$\forall x, y(x \cdot S(y) = x \cdot y + x)$$
$$\varphi(0) \wedge \forall x(\varphi(x) \rightarrow \varphi((S(x)))) \rightarrow \forall x \varphi(x)$$

The last formula is actually a formula scheme *representing infinitely many formulas, one for each $\varphi(x)$. It is called the* induction scheme.

In 1900 the famous mathematician Hilbert posed a list of 10 problems at the Second International Mathematical Congress in Paris on August 8. In print, this list was extended to 23 problems.

Hilbert's second problem is:

> Can it be proven that the axioms of logic are consistent?

Hilbert wasn't precise about which axioms or which logic he was referring to, as by that time those things hadn't been made precise yet. A concretization of the problem is the question:

> Do the Peano axioms form a consistent theory (in first order predicate logic)?

Here consistency means that one can not prove everything (or stated differently, one can not prove $\bot$). So a theory T is *consistent* if there is a formula φ such that $T \nvdash \varphi$.

In terms of foundations and philosophy of mathematics, Hilbert was a *formalist*. He devised what was later called *Hilbert's program*. The main goal of Hilbert's program was to provide secure foundations for all mathematics. In particular this should include (taken from Wikipedia):

1. A formalization of all mathematics; in other words all mathematical statements should be written in a precise formal language, and manipulated according to well defined rules.
2. Completeness: a proof that all true mathematical statements can be proved in the formalism.
3. Consistency: a proof that no contradiction can be obtained in the formalism of mathematics. This consistency proof should preferably use only "finitistic" reasoning about finite mathematical objects.
4. Conservation: a proof that any result about "real objects" obtained using reasoning about "ideal objects" (such as uncountable sets) can be proved without using ideal objects.
5. Decidability: there should be an algorithm for deciding the truth or falsity of any mathematical statement.

2 Incompleteness

Gödel's incompleteness theorems showed that most of the goals of Hilbert's program were impossible to achieve, at least if interpreted in the most obvious way. His second incompleteness theorem stated (roughly) that any consistent theory powerful enough to encode addition and multiplication of integers cannot prove its own consistency. This wipes out most of Hilbert's program.

Theorem 2.1 (Gödel's first incompleteness theorem) *If PA is consistent, then PA is syntactically incomplete,*
i.e. there is a formula φ such that $PA \nvdash \varphi$ and $PA \nvdash \neg\varphi$.

Adding more axioms to PA doesn't improve the situation: PA is *essentially incomplete*: for any "sensible" consistent extension T of PA there is a formula φ such that $T \nvdash \varphi$ and $T \nvdash \neg\varphi$. Here "sensible" means that the set of theorems in T is recursively enumerable. NB. Some weaker theories than PA are already incomplete.

Another issue is whether one can prove the consistency of PA within PA. the answer is NO, where we first have to wonder of course how to formulate the consistency of PA as a fist order formula. We will come to this a little bit later. Let's assume that $\mathrm{Con}(T)$ denotes the consistency of theory T.

Theorem 2.2 (Gödel's second incompleteness theorem) *If PA is consistent, then $PA \nvdash Con(PA)$.*

3 Proof sketch of incompleteness

We code the syntax of first order logic and the natural deduction derivations from the axioms of PA as natural numbers. For this, one uses Gödel coding of finite sequences of numbers as natural numbers, as known from the course T1b. We skip the details and just denote the coding by $\lceil - \rceil$. We can then write a formula $\mathrm{Prov}(x, y)$ which denotes (informally speaking) that x is a derivation with conclusion y. For now we just suppose that

> $\mathrm{Prov}(n, m)$ denotes that n is the code of a derivation from axioms of PA with conclusion φ, where $m = \lceil \varphi \rceil$.

Define

$$\mathrm{Thm}(x) := \exists z(\mathrm{Prov}(z, x))$$

So $\mathrm{Thm}(\lceil \varphi \rceil)$ denotes that φ is derivable in PA.

Theorem 3.1 (Fixpoint theorem) *For $\varphi(x)$ a formula with $FV(\varphi(x)) = \{x\}$, there is a closed formula A such that*

$$PA \vdash \varphi(\lceil A \rceil) \leftrightarrow A$$

This is also called the Diagonalization Lemma or the Selfreference Lemma.

Proof: One uses the fact that we can define in PA a *substitution function* s that satisfies

$$s(\lceil \psi(x) \rceil, n) = \lceil \psi(n) \rceil.$$

So s takes the code of a formula (with one free variable x) and a number n and returns the code of the same formula with now n substituted for x.

Now given a formula $\varphi(x)$ with $\mathrm{FV}(\varphi(x)) = \{x\}$, define

$$\begin{aligned} D(x) &:= \varphi(s(x,x)) \\ m &:= \ulcorner D(x) \urcorner \end{aligned}$$

Then we can prove that

$$\mathrm{PA} \vdash D(m) \leftrightarrow \varphi(\ulcorner D(m) \urcorner)$$

QED

Exercise 3.2 *Prove that PA $\vdash D(m) \leftrightarrow \varphi(\ulcorner D(m) \urcorner)$, as claimed in the proof above.*

Gödel first proved a weaker variant of the first incompleteness theorem, which uses ω-consistency.

Definition 3.3 *A theory T is ω-consistent if there is no formula $\varphi(x)$ such that*

- $T \vdash \exists x \varphi(x)$
- $T \vdash \neg\varphi(n)$ *for all* $n \in \mathbb{N}$.

Exercise 3.4 *Prove that ω-consistency implies consistency.*

Theorem 3.5 (Gödel's first incompleteness theorem) *If PA is ω-consistent, then PA is syntactically incomplete,*
i.e. there is a formula φ such that PA $\nvdash \varphi$ and PA $\nvdash \neg\varphi$.

This is the original theorem due to Gödel, which was later strengthened by Rosser to the statement that we've seen above (relying only on consistency). Note hat PA may be inconsistent and then PA proves everything, so it's complete. We can prove that PA is consistent, but only using logical principles that are stronger then PA. That's basically what the second incompleteness theorem states.

Proof: Suppose that PA is ω-consistent. (Then it is also consistent.) Define

$$B(x) := \neg\mathrm{Thm}(x)$$

So $B(x)$ states that the formula coded by x is not provable in PA. Now there is (by Diagonalization) a formula G such that

$$\mathrm{PA} \vdash B(\ulcorner G \urcorner) \leftrightarrow G.$$

This is the famous "Gödel sentence", which says "*I am not provable*". Note that

$$G \leftrightarrow B(\ulcorner G \urcorner) \leftrightarrow \neg\mathrm{Thm}(\ulcorner G \urcorner).$$

If PA $\vdash G$, then PA $\vdash \text{Prov}(n, \ulcorner G \urcorner)$ for some n (because Prov is a sound encoding of provability), so PA $\vdash \exists x \text{Prov}(x, \ulcorner G \urcorner)$, i.e. PA $\vdash \text{Thm}(\ulcorner G \urcorner)$. But this means that PA $\vdash \neg G$, which contradicts the consistency of PA, so

$$\text{PA} \nvdash G.$$

If PA $\vdash \neg G$, then PA $\vdash \exists x \text{Prov}(x, \ulcorner G \urcorner)$ (check this). Also, because PA is consistent, we know that there is no derivation of G in PA, so $PA \vdash \neg \text{Prov}(n, \ulcorner G \urcorner)$ for all n. But this contradicts the ω-consistency of PA (check this), so

$$\text{PA} \nvdash \neg G.$$

QED

Exercise 3.6 *Verify in detail the second part of the above proof. So, especially, provide all details where it says "check this".*

We may wonder whether in the "standard model" of PA, $\mathbb{N}$, the Gödel sentence is true. It is:

$$\mathbb{N} \models G$$

Suppose $\mathbb{N} \models \text{Thm}(\ulcorner G \urcorner)$. Then $\mathbb{N} \models \exists x \text{Prov}(x, \ulcorner G \urcorner)$, so there is an $n \in \mathbb{N}$ such that $\mathbb{N} \models \text{Prov}(n, \ulcorner G \urcorner)$. So there is a derivation of G in PA, so PA $\vdash G$. Contradiction. So

$$\mathbb{N} \models \neg \text{Thm}(\ulcorner G \urcorner) \text{ and so } \mathbb{N} \models G.$$

Consistency can be defined using provability as follows.

$$\text{Con}(PA) := \neg \text{Thm}(\ulcorner 0 = 1 \urcorner).$$

4 Decidability

Is PA decidable? Is predicate logic decidable? The set $S := \{\ulcorner \varphi \urcorner \mid \text{PA} \vdash \varphi\}$ is at most recursively enumerable, because $\text{Prov}(x, y)$ is primitive recursive and hence $\text{Thm}(y) := \exists x \text{Prov}(x, y)$ is recursively enumerable. But is S recursive?

No, it is not. This can be seen as a corollary of the proof of the incompleteness theorem, but we will show undecidability of predicate logic directly, as a consequence of the undecidability of the blank tape problem for Turing Machines.

We can formalize Turing machines in predicate logic as follows. Given a TM M, we define a ternary predicate $S(q, x, y)$, where q denotes a state, x denotes the tape content left of the tape head, in reverse order, y denotes the tape content right of the tape head. Tape content is denoted by ax and by, indicating the first symbols by a and b, so we leave the function symbols that make up sequences of tape symbols implicit. We assume a special state q_f being the single final state.

We now create a theory T that represents M by taking the following formulas:

$$\forall x, y(S(q_1, x, ay) \rightarrow S(q_2, bx, y))$$

is added to T if $\delta(q_1, a) = (q_2, b, R)$ in the machine M.

$$\forall x, y(S(q_1, cx, ay) \to S(q_2, x, cby))$$

is added to T if $\delta(q_1, a) = (q_2, b, L)$ in the machine M.
Etcetera.

So $S(q, x, y)$ denotes a "state" of a TM with tape input and tape head. We can now formulate the Blank tape problem as follows.

$$S(q_0, \lambda, \lambda) \to \exists x, y S(q_f, x, y)$$

This formula is derivable in the theory T if and only if Turing machine M halts on the blank tape. So predicate logic is undecidable.

It is even the case that $T \wedge S(q_0, \lambda, \lambda)$ is a logic program and $\neg S(q_f, x, y)$ is a goal. So as a corollary we find that logic programming is undecidable.

As we can code all this in PA, we can also conclude that PA is undecidable.

We have seen that $S := \{\ulcorner\varphi\urcorner \mid \text{PA} \vdash \varphi\}$ is recursively enumerable and not recursive. That's the case for any set of derivable formulas from a theory T (if T is recursive and the the notion of "derivability" makes some sense). In contrast

$$\text{Th}(\mathbb{N}) := \{\ulcorner\varphi\urcorner \mid \mathbb{N} \models \varphi\}$$

is *not* recursively enumerable. If $\text{Th}(\mathbb{N})$ were r.e., then the complement $\overline{\text{Th}(\mathbb{N})}$ is $\{\ulcorner\varphi\urcorner \mid \mathbb{N} \models \neg\varphi\}$, which is also r.e. But that would make $\text{Th}(\mathbb{N})$ recursive, which it is not.

Exercise 4.1 *What is the best you can expect from a program that tries to decide predicate logic? Given a formula, the program may either yield "yes" (derivable), "no" (not derivable) or it may not terminate.*

Exercise 4.2 *In Section 2 it is said that Gödel's theorems "wiped out most of Hilbert's program".*

Explain how and to which extent Hilbert's program was "wiped out".

References

[1] J.F.A.K. van Benthem, H.P. van Ditmarsch, J. Ketting, J.S. Lodder en W.P.M. Meyer-Viol, *Logica voor informatica*, derde editie, Pearson Education, 2003.

[2] Dirk van Dalen, *Logic and Structure*, Fourth Edition, Universitext, Springer, 2004.

[3] K. Gödel Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I, *Monats. Math. Phys.* 38, pp. 173–198, 1931.

[4] G.T. Kneebone, *Mathematical Logic and the Foundation of Mathematics*, an introductory survey, Dover publications, Mineola New York, 2001 (original: 1963, Van Nostrand, London)