Math Logic: Model Theory & Computability
Lecture 24

Sketch of proof of Gödel Incompleteness.

Def (Informal). Let σ be a finite signature. Call a σ-theory T computable if there is a computer program such that given a σ-sentence φ as input, answers YES if $\varphi \in T$ and NO if $\varphi \notin T$.

Gödel's Incompleteness. Let $\sigma := (0, S, +, \cdot)$ and $\underline{N} := (\mathbb{N}, 0, S, +, \cdot)$. Then every computable subtheory $T \subseteq Th(\underline{N})$ is incomplete. In particular, PA is incomplete.

> Gödel's theorem is like the late paintings of Claude Monet. It is easy to perceive, but from a certain distance. A close look reveals only fastidious details that one perhaps does not want to know.
>
> *Jean-Yves Girard*

Def (Informal). For each $k \in \mathbb{N}^+$, a function $f : \mathbb{N}^k \to \mathbb{N}$ is called computable if there is a computer program such that on a given input $\vec{a} \in \mathbb{N}^k$ outputs $f(\vec{a})$. A relation $R \subseteq \mathbb{N}^k$ is called called computable if $\mathbb{1}_R : \mathbb{N}^k \to \{0,1\} \subseteq \mathbb{N}$ is computable.

Prop. Computable function/relations are arithmetical, i.e. definable in $\underline{N} := (\mathbb{N}, 0, S, +, \cdot)$ (equivalently, $\emptyset$-definable in $\underline{N}$).

Proof. Will follow easily from the def of computable, once given. $\boxtimes$

Coding of tuples of natural numbers and formulas. "Clearly" the function $n \mapsto p_n :=$ nth prime number is computable and thus the function $\langle \dots \rangle_k : \mathbb{N}^k \to \mathbb{N}^+$ is a computable injection. Furthermore, the function $(n_1, \dots, n_k) \mapsto p_1^{n_1} \cdot p_2^{n_2} \cdot \ldots \cdot p_k^{n_k}$

$\langle \dots \rangle : \mathbb{N}^{<\mathbb{N}} \to \mathbb{N}^+$ is also a injective and its left-inverse is computable
$\vec{a} \mapsto \langle lh(\vec{a}), \vec{a} \rangle_{lh(\vec{a})}$

in the following sense that the decoding functions

$$lh : \mathbb{N} \to \mathbb{N} \qquad \text{and} \qquad (\cdot)_\cdot : \mathbb{N}^2 \to \mathbb{N} \quad \text{for each } i \in \mathbb{N}$$

$n \mapsto$ the largest ℓ s.t. $p_1^\ell \mid n$ $\qquad$ $(n,i) \mapsto$ the largest ℓ s.t. $p_{i+1}^\ell \mid n$ if $n \neq 0$ and $i < lh(n)$ and 0 otherwise.

are computable.

We now use this to encode $\sigma_{arithm} := (0, S, +, \cdot)$-formulas as follows. Encode the $\mathrm{Alph}(\sigma_{arithm})$ by the following number:

$$\begin{array}{cccccccccccccc} 0, & S, & +, & \cdot, & (, &), & ;, & =, & \exists, & \neg, & \wedge, & v_0, & v_1, & v_2, \dots \\ 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 \end{array}$$

and we denote the number corresponding to a symbol $s \in \mathrm{Alph}(\sigma_{arithm})$ by $c(s)$.
E.g. $c(() = 4$ and $c(v_{17}) = 28$.

Next, we encode any word $w := (w_0, \dots, w_{\ell-1}) \in \mathrm{Alph}(\sigma_{arithm})^{<\mathbb{N}}$ by the number $\ulcorner w \urcorner := \langle c(w_0), c(w_1), \dots, c(w_{\ell-1}) \rangle$. This gives a encoding $\ulcorner \varphi \urcorner$ for each σ_{arithm}-formula φ.

Prop. Because recursive definitions are programmable, the following sets are computable: $\mathrm{Formulas}(\sigma_{arithm})$, $\mathrm{Sentences}(\sigma_{arithm})$, etc.

In particular, the following function is computable:

$$\mathrm{Sub}_0 : \mathbb{N}^2 \to \mathbb{N}$$

defined by setting $\mathrm{Sub}_0(n,m) := \ulcorner \varphi(\underline{m}/v_0) \urcorner$, if $n = \ulcorner \varphi \urcorner$ for some σ_{arithm}-formula φ, and 0 otherwise. Recall that $\underline{m} := \underbrace{S(S(\dots S(0)\dots)}_{m}$. As mentioned above, computable functions are arithmetic, and we let $\mathrm{Sub}_0(x,y,z)$

be the σ-arithm formula defining this function, i.e. defining its graph, so for any $n, m, k \in \mathbb{N}$, we have that

$$Sub_0(n,m) = k \quad \text{iff} \quad \underline{N} \models \underline{Sub_0}(n,m,k).$$

We now prove a lemma, which in a sense, implies that there is a program that prints its own code. This is made possible by the fact that we can use the symbol v_0 in two ways: as a symbol and as variable that has content. This is possible even in English:

(Quine) Write the following sentence in quotes twice, the second time with quotes:
"Write the following sentence in quotes twice, the second time with quotes:"

Fixed Point Lemma (for $\underline{N}$). For each extended σ_{arithm}-formula $\varphi(v_0)$ there is a σ_{arith}-sentence θ such that $\underline{N} \models \theta \leftrightarrow \varphi(\ulcorner\theta\urcorner/v_0)$.

Remark. This establishes a bridge between the code of θ and and the interpretation of θ, just like in the English sentence above.

Proof. Take $\psi := \exists z\,(\underline{Sub_0}(v_0, v_0, z) \wedge \varphi(z))$, so $\psi(v_0)$ is an extended formula and let $m := \ulcorner\psi\urcorner$. Let $\theta := \psi(\dot{m}/v_0)$ where $\dot{m} := \underbrace{S(S(\ldots S(0))\ldots)}_{m}$.
Note that $Sub_0(m,m) = Sub_0(\ulcorner\psi\urcorner, m) = \ulcorner\psi(\dot{m}/v_0)\urcorner = \ulcorner\theta\urcorner$.
Thus, (*) $\underline{N} \models \underline{Sub_0}(m,m,\ulcorner\theta\urcorner)$. Now watch the magic:

$$\begin{aligned}
\underline{N} \models \theta \quad & \text{iff} \quad \underline{N} \models \psi(\dot{m}/v_0) \\
& \text{iff} \quad \underline{N} \models \exists z\,(\underline{Sub_0}(\dot{m},\dot{m},z) \wedge \varphi(z)) \\
& \text{iff} \quad \text{there is } b \in \underline{N} \text{ s.t. } \underline{N} \models \underline{Sub_0}(m,m,b) \wedge \varphi(b) \\
& \text{iff} \quad \underline{N} \models Sub_0(m,m,\ulcorner\theta\urcorner) \wedge \varphi(\ulcorner\theta\urcorner) \\
\text{(by (*))} \quad & \text{iff} \quad \underline{N} \models \varphi(\ulcorner\theta\urcorner).
\end{aligned}$$

$\boxed{:-p}$