Gödel's First Incompleteness Theorem
Unit 4: The Power of Arithmetic

Lavinia Picollo

1 Expressing primitive recursive functions

Proposition 1. *L_A can express the factorial function.*

Proof. To prove this, we need to find a wff $\mathsf{F}(\mathsf{x},\mathsf{y})$ of L_A s.t., for every m, n, $\mathsf{F}(\overline{\mathsf{m}},\overline{\mathsf{n}})$ is true just in case $m! = n$. Given the recursive definition we gave of the function, $x! = y$ is equivalent to:

(A!) There is a sequence of numbers $k_0, k_1, \ldots, k_x$ s.t. $k_0 = 1$; if $v < x$, then $k_{Sv} = k_v \times Sv$; and $k_x = y$.

If we could talk about finite sequences of numbers in L_A, our job would be done. And we can, via coding them by numbers. Actually, we will code them using a pair of numbers, i.e. we will assign a unique ordered pair of natural numbers to each finite sequence using a function *decode* that we can already express in L_A.

We choose Gödel's β-function: $\beta(x, y, z) =$ the remainder left when x is divided by $y(z + 1) + 1$.

Proposition 2. *For any finite sequence of natural numbers $k_0, k_1, \ldots, k_n$ there is a pair of (code) numbers c, d s.t. for every $i \leq n$, $\beta(c, d, i) = k_i$.*

Therefore, we can reformulate (A!) – and thus $x! = y$ – as follows:

(B!) There is some pair of numbers c, d s.t. $\beta(c, d, 0) = 1$; if $v < x$, then $\beta(c, d, Sv) = \beta(c, d, v) \times Sv$; and $\beta(c, d, x) = y$.

Note that β is expressible in L_A by the Δ_0 formula

$$\mathsf{B}(\mathsf{x},\mathsf{y},\mathsf{z},\mathsf{w}) =_{def} \exists \mathsf{v} \leq \mathsf{x}\, (\mathsf{x} = \mathsf{v} \times \mathsf{S}(\mathsf{y} \times \mathsf{Sz}) + \mathsf{w} \wedge \mathsf{w} \leq \mathsf{y} \times \mathsf{Sz})$$

So, we can rephrase (B!) – and thus $x! = y$ – in L_A as follows:

(C!) $\exists \mathsf{z} \exists \mathsf{w}\, (\mathsf{B}(\mathsf{z},\mathsf{w},0,\overline{1}) \wedge \forall \mathsf{v} \leq \mathsf{x}\, (\mathsf{v} \neq \mathsf{x} \to \exists \mathsf{u}_1 \exists \mathsf{u}_2\, (\mathsf{B}(\mathsf{z},\mathsf{w},\mathsf{v},\mathsf{u}_1) \wedge \mathsf{B}(\mathsf{z},\mathsf{w},\mathsf{Sv},\mathsf{u}_2) \wedge \mathsf{u}_2 = \mathsf{u}_1 \times \mathsf{Sv})) \wedge \mathsf{B}(\mathsf{z},\mathsf{w},\mathsf{x},\mathsf{y}))$.

□

Proposition 3. *All initial functions can be expressed in L_A*

Proof. For the successor function the case is trivial, as we have a function symbol in the language for it: it can be expressed by the formula $\mathsf{Sx} = \mathsf{y}$.

For the zero function we have the formula $\mathsf{Z}(\mathsf{x},\mathsf{y}) =_{def} \mathsf{x} = \mathsf{x} \wedge \mathsf{y} = 0$.

Finally, for each projection I_i^k we have the following formula:

$$\mathsf{I}_\mathsf{i}^\mathsf{k}(\mathsf{x}_1, \ldots, \mathsf{x}_\mathsf{k}, \mathsf{y}) =_{def} \mathsf{x}_1 = \mathsf{x}_1 \wedge \cdots \wedge \mathsf{x}_\mathsf{k} = \mathsf{x}_\mathsf{k} \wedge \mathsf{x}_\mathsf{i} = \mathsf{y}$$

$\square$

Proposition 4. *If two functions are expressible in L_A, then so are their compositions.*

Proof. Let $\mathsf{G}(\vec{\mathsf{y}}, \mathsf{v})$ (possibly a numeral) and $\mathsf{H}(\vec{\mathsf{x}}, \mathsf{u}, \vec{\mathsf{z}}, \mathsf{v})$ express in L_A the functions $g(\vec{y})$ (possibly a constant) and $h(\vec{x}, u, \vec{z})$ resp. Then, the following wff expresses $f(\vec{x}, \vec{y}, \vec{z}) := h(\vec{x}, g(\vec{y}), \vec{z})$:

$$\mathsf{F}(\vec{\mathsf{x}}, \vec{\mathsf{y}}, \vec{\mathsf{z}}, \mathsf{v}) =_{def} \exists \mathsf{u}\, (\mathsf{G}(\vec{\mathsf{y}}, \mathsf{u}) \wedge \mathsf{H}(\vec{\mathsf{x}}, \mathsf{u}, \vec{\mathsf{z}}, \mathsf{v}))$$

$\square$

Proposition 5. *If a function is defined by primitive recursion from two functions expressible in L_A, then it is also expressible in L_A.*

Proof. Let $f(\vec{x})$ be defined by primitive recursion from the functions (or constant) $g(\vec{x})$ and $h(\vec{x}, y, z)$. Then, we have that $f(\vec{x}, 0) = g(\vec{x})$, and $f(\vec{x}, Sy) := h(\vec{x}, y, f(\vec{x}, y))$. Thus, we can say that $f(\vec{x}, y) = z$ as follows:

(A_f) There is a sequence of numbers $k_0, k_1, \ldots, k_y$ s.t. $k_0 = g(\vec{x})$; if $v < y$, then $k_{Sv} = h(\vec{x}, v, k_v)$; and $k_y = z$.

which, in turn, can be rephrased as

(B_f) There is some pair of numbers c, d s.t. $\beta(c, d, 0) = g(\vec{x})$; if $v < y$, then $\beta(c, d, Sv) = h(\vec{x}, v, \beta(c, d, v))$; and $\beta(c, d, y) = z$.

Let $\mathsf{G}(\vec{\mathsf{x}}, \mathsf{v})$ and $\mathsf{H}(\vec{\mathsf{x}}, \mathsf{y}, \mathsf{z}, \mathsf{v})$ express in L_A the functions g and h resp. Thus, $f(\vec{x}, y)$ can be expressed in L_A too, by the following formula $\mathsf{F}(\tilde{\mathsf{x}}, \mathsf{y}, \mathsf{z})$:

(C_f) $\exists \mathsf{w}_1 \exists \mathsf{w}_2\, (\exists \mathsf{v}\, (\mathsf{B}(\mathsf{w}_1, \mathsf{w}_2, 0, \mathsf{v}) \wedge \mathsf{G}(\vec{\mathsf{x}}, \mathsf{v})) \wedge$
$\forall \mathsf{v} \leq \mathsf{y}\, (\mathsf{v} \neq \mathsf{y} \rightarrow \exists \mathsf{u}_1 \exists \mathsf{u}_2\, (\mathsf{B}(\mathsf{w}_1, \mathsf{w}_2, \mathsf{v}, \mathsf{u}_1) \wedge \mathsf{B}(\mathsf{w}_1, \mathsf{w}_2, \mathsf{S}\mathsf{v}, \mathsf{u}_2) \wedge \mathsf{H}(\vec{\mathsf{x}}, \mathsf{v}, \mathsf{u}_1, \mathsf{u}_2))) \wedge \mathsf{B}(\mathsf{w}_1, \mathsf{w}_2, \mathsf{y}, \mathsf{z}))$

$\square$

Theorem 23. *L_A can express all p.r. functions.*

Proof. By Propositions 3, 4, and 5. $\square$

Theorem 24. *L_A can express every p.r. function by a Σ_1 wff.*

Exercise 1 (*)**.** *Prove Theorem 24 by induction on the class of p.r. functions. (Check Unit 2, Proposition 3).*

2 Capturing primitive recursive functions

Definition (caputring/representing functions)**.** $\varphi(\mathsf{x}_1, \ldots, \mathsf{x}_\mathsf{m}, \mathsf{y})$ *captures the $m-$place function f in T iff, for every $n+1$-tuple of natural numbers $n_1, \ldots, n_m, k$:*

1. *if $f(n_1, \ldots, n_m) = k$, then $T \vdash \varphi(\overline{\mathsf{n}_1}, \ldots, \overline{\mathsf{n}_\mathsf{m}}, \overline{\mathsf{k}})$;*
2. *if $f(n_1, \ldots, n_m) \neq k$, then $T \vdash \neg\varphi(\overline{\mathsf{n}_1}, \ldots, \overline{\mathsf{n}_\mathsf{m}}, \overline{\mathsf{k}})$;*
3. *$T \vdash \exists ! \mathsf{y}\, \varphi(\overline{\mathsf{n}_1}, \ldots, \overline{\mathsf{n}_\mathsf{m}}, \mathsf{y})$*

Proposition 6. *Let* $\mathsf{Q} \subseteq \mathsf{T}$*,* f *be an* m*-place function, and* $\varphi(\mathsf{x_1}, \ldots, \mathsf{x_m}, \mathsf{y})$ *be a wff of* L_A *s.t., if* $f(n_1, \ldots, n_m) = k$*, then* $T \vdash \varphi(\overline{\mathsf{n_1}}, \ldots, \overline{\mathsf{n_m}}, \overline{\mathsf{k}})$*, and if* $f(n_1, \ldots, n_m) \neq k$*, then* $T \vdash \neg\varphi(\overline{\mathsf{n_1}}, \ldots, \overline{\mathsf{n_m}}, \overline{\mathsf{k}})$*. Then, there is a wff* $\tilde{\varphi}(\mathsf{x_1}, \ldots, \mathsf{x_m}, \mathsf{y})$ *that captures* f *in* T*.*

Proof. Assume $\varphi(\mathsf{x_1}, \ldots, \mathsf{x_m}, \mathsf{y})$ captures f in T and let

$$\tilde{\varphi}(\mathsf{x_1}, \ldots, \mathsf{x_m}, \mathsf{y}) =_{def} \varphi(\mathsf{x_1}, \ldots, \mathsf{x_m}, \mathsf{y}) \wedge \forall \mathsf{z} \leq \mathsf{y}\, (\varphi(\mathsf{x_1}, \ldots, \mathsf{x_m}, \mathsf{z}) \to \mathsf{z} = \mathsf{y})$$

Clearly, conditions 1 and 2 of the previous definition are satisfied by $\tilde{\varphi}$. We want to derive $\exists \mathsf{y}\, (\tilde{\varphi}(\overline{\mathsf{n_1}}, \ldots, \overline{\mathsf{n_m}}, \mathsf{y}) \wedge \forall \mathsf{w}\, (\tilde{\varphi}(\overline{\mathsf{n_1}}, \ldots, \overline{\mathsf{n_m}}, \mathsf{w}) \to \mathsf{w} = \mathsf{y}))$ in T. Let $f(n_1, \ldots, n_m) = k$. By hypothesis, we have that $T \vdash \varphi(\overline{\mathsf{n_1}}, \ldots, \overline{\mathsf{n_m}}, \overline{\mathsf{k}})$, but also that $T \vdash \neg\varphi(\overline{\mathsf{n_1}}, \ldots, \overline{\mathsf{n_m}}, \overline{\mathsf{i}})$ for each $i < k$. Since T includes Q, this means that $T \vdash \forall \mathsf{z} \leq \overline{\mathsf{k}}\, (\varphi(\mathsf{x_1}, \ldots, \mathsf{x_m}, \mathsf{z}) \to \mathsf{z} = \overline{\mathsf{k}})$ (cf. Unit 2, Proposition 3). Thus, $T \vdash \tilde{\varphi}(\overline{\mathsf{n_1}}, \ldots, \overline{\mathsf{n_m}}, \overline{\mathsf{k}})$. We reason in T:

1.	$\tilde{\varphi}(\overline{\mathsf{n_1}}, \ldots, \overline{\mathsf{n_m}}, \mathsf{w})$	assumption
2.	$\varphi(\overline{\mathsf{n_1}}, \ldots, \overline{\mathsf{n_m}}, \mathsf{w}) \wedge \forall \mathsf{z} \leq \mathsf{w}\, (\varphi(\overline{\mathsf{n_1}}, \ldots, \overline{\mathsf{n_m}}, \mathsf{z}) \to \mathsf{z} = \mathsf{w})$	
3.	$\mathsf{w} \leq \overline{\mathsf{k}} \vee \overline{\mathsf{k}} \leq \mathsf{w}$	Unit 2, Prop. 3
4.	$\mathsf{w} \leq \overline{\mathsf{k}} \to \mathsf{w} = \overline{\mathsf{k}}$	Check!
5.	$\overline{\mathsf{k}} \leq \mathsf{w}$	assumption
6.	$(\varphi(\overline{\mathsf{n_1}}, \ldots, \overline{\mathsf{n_m}}, \overline{\mathsf{k}}) \to \overline{\mathsf{k}} = \mathsf{w})$	2
7.	$\mathsf{w} = \overline{\mathsf{k}}$	6
8.	$\overline{\mathsf{k}} \leq \mathsf{w} \to \mathsf{w} = \overline{\mathsf{k}}$	5-7
9.	$\mathsf{w} = \overline{\mathsf{k}}$	3, 4, 8
10.	$\forall \mathsf{w}\, (\tilde{\varphi}(\overline{\mathsf{n_1}}, \ldots, \overline{\mathsf{n_m}}, \mathsf{w}) \to \mathsf{w} = \overline{\mathsf{k}}))$	1-9
11.	$\tilde{\varphi}(\overline{\mathsf{n_1}}, \ldots, \overline{\mathsf{n_m}}, \overline{\mathsf{k}}) \wedge \forall \mathsf{w}\, (\tilde{\varphi}(\overline{\mathsf{n_1}}, \ldots, \overline{\mathsf{n_m}}, \mathsf{w}) \to \mathsf{w} = \overline{\mathsf{k}}))$	10
12.	$\exists \mathsf{y}\, (\tilde{\varphi}(\overline{\mathsf{n_1}}, \ldots, \overline{\mathsf{n_m}}, \mathsf{y}) \wedge \forall \mathsf{w}\, (\tilde{\varphi}(\overline{\mathsf{n_1}}, \ldots, \overline{\mathsf{n_m}}, \mathsf{w}) \to \mathsf{w} = \mathsf{y}))$	11

□

Proposition 7. Q *can capture every initial function by a* Σ_1 *formula.*

Exercise 2. *Prove Proposition 7.*

Proposition 8. *If* Q *can capture the functions* g *and* h *by* Σ_1 *formulas, it can also capture a function* f *defined by composition from* g *and* h *by a* Σ_1 *formula.*

Exercise 3. *Prove Proposition 8.*

Proposition 9. *If* Q *can capture the functions* g *and* h *by* Σ_1 *formulas, it can also capture a function* f *defined by primitive recursion from* g *and* h *by a* Σ_1 *formula.*

Exercise 4 (*). *Prove Proposition 9.*

Theorem 25. Q *can capture any p.r. function by a* Σ_1 *formula.*

3 Expressing and capturing properties and relations

Theorem 26. L_A *can express every p.r. property and relation by a* Σ_1 *wff.*

Exercise 5. *Prove Theorem 26.*

Theorem 27. Q *can capture every p.r. property and relation by a* Σ_1 *wff.*

Exercise 6. *Prove Theorem 27.*