Forever undecided
paper for the Models of Grammar Staff Seminar

Elias Thijsse

T&I, Arts, Tilburg University

November 13, 2002

Overview

- Historical Context
- Synopsis of Proof
- Encoding
- Recursion Theory
- Axiom Systems
- Representation
- Diagonalization
- Consequences & Further Work
- (Mis)interpretations
- References

Historical Context

before 1900 Foundations of Mathematics. Cantor's set theory seemed to provide basis.

around 1900 Antinomies of set theory, e.g., the Russell paradox. Any set could be formed by implicit definition, i.e. $\{x : \varphi(x)\}$ is OK, so consider $R = \{x : x \notin x\}$. Then, however, $R \in R \Leftrightarrow R \notin R$.

$\sim$ 1910, 1920 Different schools: intuitionism (Brouwer), type theory (Whitehead & Russell), formalism (Hilbert c.s.). Hilbert's program: prove conservation and consistency of axiom systems (by finitistic means). Famous systems: PM (Principia Mathematica for typed logic), ZF (Zermelo-Fraenkel for set theory) and PA (Peano Arithmetic for simple arithmetic).

1930,1931 Gödel's incompleteness theorems: Any consistent theory containing PA leaves some sentence undecided, i.e. neither the sentence nor its negation can be proven as a consequence of the theory; moreover the consistency of the theory cannot be shown within the theory: the sentence asserting consistency is not derivable.

Synopsis of Proof

Gödel based his proof on PM, including arithmetic. We focus on the simpler first-order theory PA. PA $\vdash$ φ means that φ can be derived from PA.

The main idea of Gödel's proof is to construct a liar sentence ψ in the language of PA, expressing its own non-derivability, roughly: PA $\vdash \psi \leftrightarrow \neg$'PA $\vdash \psi$' (*). Now, we can show that neither ψ, nor $\neg\psi$ is derivable from PA: suppose PA $\vdash \psi$, then (by *) PA $\vdash \neg$'PA $\vdash \psi$', hence (by truthfulness of quotation and consistency PA) PA $\nvdash \psi$ $\lightning$. Thus (!) PA $\nvdash \psi$. Next suppose that PA $\vdash \neg\psi$, so (by *) PA $\vdash$ 'PA $\vdash \psi$', hence PA $\vdash \psi$, contradicting the previous conclusion. Thus PA $\nvdash \neg\psi$.

Main parts of the proof:

1. *Encoding* the logical syntax in numbers, expression X is coded by its Gödel number $\ulcorner X \urcorner$;
2. *Recursion Theory*: systematic development of arithmetical functions in order to get
3. *Representation* of those functions, and what they express, in PA, and finally
4. *Diagonalization*: actual construction of the liar sentence by self-applied substitution.

First-order logic

The language of first-order logic (FOL) is restricted to the (logical) symbols $=$, $\neg$, $\rightarrow$, and $\forall$. Syntax as usual.

The axioms of FOL are universal generalizations (by prefixing any string $\forall v_0 \cdots \forall v_k$) of (instantiations of) the schemes:

1. $\varphi \rightarrow (\psi \rightarrow \varphi)$
2. $(\varphi \rightarrow (\psi \rightarrow \chi) \rightarrow ((\varphi \rightarrow \psi) \rightarrow (\varphi \rightarrow \chi))$
3. $(\neg\varphi \rightarrow \neg\psi) \rightarrow (\psi \rightarrow \varphi)$.
4. $\forall v_0 \varphi \rightarrow [t/v_0]\varphi$ (with proviso on binding)
5. $\varphi \rightarrow \forall v_0 \varphi$ (provided v_0 not free in φ)
6. $\forall v_0(\varphi \rightarrow \psi) \rightarrow (\forall v_0 \varphi \rightarrow \forall v_0 \psi)$

The only inference rule is *modus ponens*: from φ and $\varphi \rightarrow \psi$ infer ψ ($\varphi, \varphi \rightarrow \psi \vdash \psi$).

Peano Arithmetic

The language of PA contains the (individual) constant 0, unary function symbol $succ$ (where $succ(x) = x + 1$) and binary function symbols $+$ and $\times$ in infix notation. Syntax as usual. Formulas such as $\forall v_0\ v_0 + v_0 = succ(succ(0)) \times v_0$.

The axioms of PA are those of FOL with the additions:

1. $\forall v_0\ \neg succ(v_0) = 0$
2. $\forall v_0 \forall v_1 (succ(v_0) = succ(v_1) \rightarrow v_0 = v_1)$
3. $\forall v_0\ v_0 + 0 = v_0$
4. $\forall v_0 \forall v_1\ v_0 + succ(v_1) = succ(v_0 + v_1)$
5. $\forall v_0\ v_0 \times 0 = 0$
6. $\forall v_0 \forall v_1\ v_0 \times succ(v_1) = (v_0 \times v_1) + v_0$
7. $\varphi(0) \rightarrow (\forall v_0(\varphi(v_0) \rightarrow \varphi(succ(v_0))) \rightarrow \forall v_0\, \varphi(v_0))$ (induction scheme)

Encoding

Basic coding:(Gödel uses prime numbers, Smoryński coded ordered pairs, we simplify)

$$\begin{array}{rcccccccccc} & 0 & succ & + & \times & < & = & \neg & \to & \forall & v_i \\ g: & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & i+9 \end{array}$$

The general idea of coding a string of symbols is multiplying powers of consecutive primes:

$$\begin{array}{cccccccccccccc} 2 & 3 & 5 & 7 & 11 & 13 & 17 & 19 & 23 & 29 & 31 & 37 & 41 \ldots \\ p_0 & p_1 & p_2 & p_3 & p_4 & p_5 & p_6 & p_7 & p_8 & p_9 & p_{10} & p_{11} & p_{12} \cdots \end{array}$$

So, e.g. $\ulcorner aba \urcorner = 2^{\ulcorner a \urcorner} \times 3^{\ulcorner b \urcorner} \times 5^{\ulcorner a \urcorner}$. In general $\ulcorner s_0 \cdots s_n \urcorner = p_0^{\ulcorner s_0 \urcorner} \times p_1^{\ulcorner s_1 \urcorner} \times \cdots \times p_n^{\ulcorner s_n \urcorner}$. This is abbreviated as $(\ulcorner s_0 \urcorner, \ulcorner s_1 \urcorner, \ldots, \ulcorner s_n \urcorner)$. By the Fundamental Theorem of Arithmetic the encoding is correct. This operation is used recursively to encode terms, formulas and deductions.

$$\begin{array}{ll} \ulcorner 0 \urcorner = (g(0)) = 2^0 = 1 & \ulcorner (t < t') \urcorner = (4, \ulcorner t \urcorner, \ulcorner t' \urcorner) \\ \ulcorner v_i \urcorner = (g(v_i)) = 2^{i+9} & \ulcorner (t = t') \urcorner = (5, \ulcorner t \urcorner, \ulcorner t' \urcorner) \\ \ulcorner succ(t) \urcorner = (1, \ulcorner t \urcorner) & \ulcorner \neg \varphi \urcorner = (6, \ulcorner \varphi \urcorner) \\ \ulcorner (t + t') \urcorner = (2, \ulcorner t \urcorner, \ulcorner t' \urcorner) & \ulcorner (\varphi \to \psi) \urcorner = (7, \ulcorner \varphi \urcorner, \ulcorner \psi \urcorner) \\ \ulcorner (t \times t') \urcorner = (3, \ulcorner t \urcorner, \ulcorner t' \urcorner) & \ulcorner \forall v_i \varphi \urcorner = (8, 2^{i+9}, \ulcorner \varphi \urcorner) \end{array}$$

Recursion Theory

Goal: what can be described by (simple) arithmetical functions, which can then be expressed in PA

Like Gödel we use so-called primitive recursive functions, rather than the more general *recursive* functions. A function is *primitive recursive* if it is defined through a finite number of steps using only:

- The basic functions $zero$, $succ$ and $proj_i^n$ where $zero(x) = 0$, $succ(x) = x + 1$ and $proj_i^n(\vec{x}) = x_i$, where $\vec{x} = \langle x_1, \ldots, x_n \rangle$
- [composition] If $f, g_1, \ldots, g_m$ are primitive recursive, so is the function h defined by $h(\vec{x}) = f(g_1(\vec{x}), \ldots, g_m(\vec{x}))$. The notations $f \circ g_1 \cdots g_m$ and $f(g_1, \ldots, g_m)$ are both used for h.
- [recursion] If f and g are primitive recursive, so is $h = rec\ f\ g$ defined by:
  - $h(\vec{x}, 0) = f(\vec{x}, 0)$
  - $h(\vec{x}, y + 1) = g(h(\vec{x}, y), \vec{x}, y)$

Examples:

1. *constant* functions are defined by iterated succession, e.g. $f : x \mapsto 2$ is defined by $f = succ \circ (succ \circ zero)$.

2. *addition* needs recursion: $\begin{cases} x+0=x \\ x+(y+1)=(x+y)+1 \end{cases}$
so $+ = rec\ proj_1^2\ succ \circ proj_1^3$.

3. *multiplication* needs recursion and addition:
$\begin{cases} x \times 0 = 0 \\ x \times (y+1) = (x \times y) + x \end{cases}$
so $\times = rec\ zero \circ proj_1^2\ proj_1^3 + proj_2^3$.

4. also, *exponentiation* as iterated multiplication:
$\begin{cases} x \uparrow 0 = x^0 = 1 \\ x \uparrow (y+1) = x^{y+1} = (x^y) \times x \end{cases}$
so $\uparrow = rec\ succ \circ zero \circ proj_1^2\ proj_1^3 \times proj_2^3$.

5. *faculty* $\begin{cases} fac(0) = 1 \\ fac(x+1) = fac(x) \times (x+1) \end{cases}$
is defined by:
$fac = rec\ succ \circ zero\ proj_1^2 \times succ \circ proj_2^2$.
Instead of $fac(x)$ we write $x!$.

6. the *sign* function $sg(x) = \begin{cases} 0 & \text{if } x = 0 \\ 1 & \text{if } x \neq 0 \end{cases}$ is defined by $sg = rec\ zero\ succ \circ zero \circ proj_1^2$; its complement $\overline{sg}(x) = 1 - sg(x)$ is defined by: $\overline{sg} = rec\ succ \circ zero\ zero \circ proj_1^2$.

7. $predec(x) = \begin{cases} 0 & \text{if } x = 0 \\ x-1 & \text{if } x \neq 0 \end{cases}$ is defined by $predec = rec\ zero\ proj_2^2$.

8. *cut-off subtraction* or *monus*: $x \dot{-} y = \begin{cases} x - y & \text{if } x \geq y \\ 0 & \text{if } x < y \end{cases}$ is defined by $\dot{-} = rec\ proj_1^2\ predec \circ proj_1^3$.

9. *definition by cases*. If f_1, f_2, g are primitive recursive, so is $h(\vec{x}) = \begin{cases} f_1(\vec{x}) & \text{if } g(\vec{x}) = 0 \\ f_2(\vec{x}) & \text{if } g(\vec{x}) \neq 0 \end{cases}$
   Proof: $h = f_1 \times \overline{sg} \circ g + f_2 \times sg \circ g$. (This can be generalized to many cases)

10. *closure under restricted sum and product*. If $f(\vec{x}, y)$ is primitive recursive, so are $g(\vec{x}, y) = \sum_{z \leq y} f(\vec{x}, z)$ and $h(\vec{x}, y) = \prod_{z \leq y} f(\vec{x}, z)$.
    Proof for h: $h = rec\ f\ proj_1^{n+2} \times$
    $f(proj_2^{n+2}, \ldots, proj_{n+1}^{n+2}, succ \circ proj_{n+2}^{n+2})$.

A *predicate* (*relation*) is primitive recursive if its characteristic function is (where χ_P is complementary to usual, i.e. $\chi_P(\vec{x}) = 0$ if $P(\vec{x})$ and $\chi_P(\vec{x}) = 1$ if $\neg P(\vec{x})$).

Examples:

11. $=$ is prim. recursive, for defined by $sg((proj_1^2 \dot{-} proj_2^2) + (proj_2^2 \dot{-} proj_1^2))$.

12. $<$ is prim. recursive, for defined by $sg(succ \circ proj_1^2 \dot{-} proj_2^2)$.

13. $\leq$ is prim.rec., defined by $sg(proj_1^2 \dot{-} proj_2^2)$.

14. *some logical closures*. If R and S are prim. recursive, so are $\neg R$, $R \vee S$ and $\exists y \leq x R$. Proofs: $\chi_{\neg R} = \overline{sg} \circ \chi_R$, $\chi_{R \vee S} = \chi_R \times \chi_S$ and $\chi_{\exists z \leq y R}(\vec{x}, y) = \prod_{z \leq y} \chi_R(\vec{x}, z)$.

15. *divisibility*. $x \mid y$ is prim.rec., since $x \mid y \Leftrightarrow \exists z \leq y \; y = x \times z$ [Gödel, concept 1]

16. *prime number*. $Prime(x) \Leftrightarrow 1 < x \wedge \neg \exists z \leq x [z \mid x \wedge z \neq 1 \wedge z \neq x]$ [Gödel, concept 2]

17. *bounded minimization*. If $P(\vec{x}, y)$ is prim.rec., so is $f(\vec{x}, y) = \mu z < y \; P(\vec{x}, z)$ [the *minimal* $z < y$ with $P(\vec{x}, z)$; is there is no such z, let $f(\vec{x}, y) = y$].
Proof: $\begin{cases} f(\vec{x}, 0) = 0 \\ f(\vec{x}, y+1) = \begin{cases} y+1 & \text{if } \neg \exists z \leq y P(\vec{x}, y) \\ f(\vec{x}, y) & \text{if } \exists z \leq y P(\vec{x}, y) \end{cases} \end{cases}$

18. nth *prime*. $prime(x)$ is recursively defined by: $prime(0) = 2$, $prime(x+1) = \mu y < prime(x)! + 1 [prime(x) < y \wedge Prime(y)]$. Instead of $prime(x)$ we usually write p_x. [Gödel, concept 5]

19. *sequence code*. $Seq(a) \Leftrightarrow a = 1 \vee (a > 1 \wedge \forall x \leq a [p_{x+1} \mid a \wedge p_x \nmid a])$. The *length* of sequence code is:

$$lh(a) = \begin{cases} \mu x \leq a[p_x \mid a \wedge p_{x+1} \nmid a] & \text{if } Seq(a) \wedge a \neq 1 \\ 0 & \text{if } \neg Seq(a) \vee a = 1 \end{cases}$$

The *number coded at place* x is $(a)_x = \mu y \leq x+1[p_x^y \mid a \wedge p_x^{y+1} \nmid a]$. E.g., if $a = 567000 = 2^3 \times 3^4 \times 5^3 \times 7$, then a codes $\langle 3,4,3,1\rangle$, $lh(a) = 3$ (!) and $(a)_0 = 3, (a)_1 = 4$ etc.

20. *coded concatenation*

$$a*b = \begin{cases} a \times \prod_{x \leq lh(b)} p_{lh(a)+x+1}^{(b)_x} & \text{if } a \neq 1 \wedge b \neq 1 \\ a & \text{if } a \neq 1 \wedge b = 1 \\ b & \text{if } a = 1 \end{cases}$$

So, $\ulcorner \langle k_1, \ldots, k_m \rangle \urcorner * \ulcorner \langle k_{m+1}, \ldots, k_{m+n} \rangle \urcorner = \ulcorner \langle k_1, \ldots, k_{m+n} \rangle \urcorner$.

21. *course–of–values recursion*. If f and g are prim.rec., so is h defined by: $\begin{cases} h(\vec{x}, 0) = f(\vec{x}, 0) \\ h(\vec{x}, y+1) = g(h^*(\vec{x}, y), \vec{x}, y) \end{cases}$
where $h^*(\vec{x}, y) = (\ h(\vec{x}, 0), \ldots, h(\vec{x}, y)\ )$
Proof: $h^* = rec\ 2^{f(\vec{x},0)}\ proj_1^3 * g$ and $h = (h^*)_y$ are both prim.rec.

22. *terms*. $Var(x) \Leftrightarrow \exists i < x[x = 2^i \wedge i > 8]$. Then

$$Term(x) \Leftrightarrow \begin{cases} 0 = 0 & \text{if } x = 1 \\ Term(y) & \text{if } x = (1, y) \\ Term(y) \wedge Term(z) & \text{if } x = (\frac{2}{3}, y, z) \\ Var(x) & \text{else} \end{cases}$$

(Here $\frac{2}{3}$ abbreviates "either 2 or 3")

23. *standard names*. the function $num : x \mapsto \ulcorner \underline{x} \urcorner$ assigns standard names (logical numerals) to numbers by simple recursion.

24. *formulas*: similar to terms.

25. *substitution*: $sub(x, i, \ulcorner t \urcorner) = \ulcorner [t/v_i]X \urcorner$ where $\ulcorner X \urcorner = x$, $Term(x)$ or $Form(x)$, and $Term(\ulcorner t \urcorner)$.
By recursion and cases define: $sub(x, i, y) =$
$$\begin{cases} y & \text{if } x = 2^{i+9} \\ (m, sub(n, i, y)) & \text{if } x = (\frac{1}{6}, n) \\ (m, sub(n, i, y), sub(r, i, y)) & \text{if } x = (m, n, r) \wedge m \neq 8 \\ (8, n, sub(r, i, y)) & \text{if } x = (8, n, r) \wedge n \neq 2^{i+9} \\ x & \text{else} \end{cases}$$
Then we can define
$$sub(x, y) = \begin{cases} sub(x, i, y) & \text{if } i = \mu j{<}x\ sub(x, j, j) \neq x \\ x & \text{else} \end{cases}$$

26. *axioms*: spelling out their forms.

27. *deductions*: strings of axioms or derived formulas. Define the prim. rec. Bew_T by: $Bew_T(x, y) \Leftrightarrow Seq(x) \wedge y = (x)_{lh(x)} \wedge \forall i \leq lh(x)[Axiom_T((x)_i) \vee (Form((x)_i) \wedge \exists j, k < i : (x)_k = (7, (x)_j, (x)_i))]$.
Then define provability: $Bew_T(y) \Leftrightarrow \exists x Bew_T(x, y)$.
Notice that the latter property is *not* prim.rec., but as a (non-rec.) final step this does not hinder the proof!

Representation

The primitive recursive functions can be represented in PA, i.e. for every prim. rec. f there is a formula φ_f such that for any $\vec{x}$:

$$\text{PA} \vdash \forall v_0(\varphi_f(\underline{\vec{x}}, v_0) \leftrightarrow v_0 = \underline{f(\vec{x})}).$$

Proof Still a lot of tedious details. The basic functions are easy, though:
PA $\vdash \forall v_0\ v_0 = 0 \leftrightarrow v_0 = \underline{zero(x)}$
PA $\vdash \forall v_0\ v_0 = succ(\underline{x}) \leftrightarrow v_0 = \underline{succ(x)}$
PA $\vdash \forall v_0\ v_0 = \underline{x_i} \leftrightarrow v_0 = \underline{proj_i^n(\vec{x})}$

NB Here x is a *meta-logical* variable, cf. a parameter. These formulas are actually sentences.

Diagonalization

Fixpoint or Diagonalization Lemma
If $\varphi(x)$ is in the language of theory T then there is a sentence ψ such that $T \vdash \psi \leftrightarrow \varphi(\ulcorner\psi\urcorner)$.
Proof: Cf. Cantor's diagonalization proof that the set of real numbers is uncountable. Let $\psi = \varphi(sub(\ulcorner\varphi(sub(x,x))\urcorner, \ulcorner\varphi(sub(x,x))\urcorner))$. Then $\psi \leftrightarrow \varphi(\ulcorner\varphi(sub(\ulcorner\varphi(sub(x,x))\urcorner, \ulcorner\varphi(sub(x,x))\urcorner))\urcorner) = \varphi(\ulcorner\psi\urcorner)$.

First Incompleteness Theorem
Let T be a (ω-)consistent extension of PA. Then there is a sentence ψ in the language of T such that $T \nvdash \psi$ and $T \nvdash \neg\psi$.
Proof: Let $\varphi(x) = \neg Bew_T(x)$. By the Diagonalization Lemma, there is a ψ such that (†) $T \vdash \psi \leftrightarrow \neg Bew_T(\ulcorner\psi\urcorner)$. The rest is a formal repetition of the earlier proof sketch, using the fact that under the conditions for T: (*) $T \vdash \alpha \Leftrightarrow T \vdash Bew_T(\ulcorner\alpha\urcorner)$ for any α in the language of T. Then (i) suppose $T \vdash \psi \overset{*}{\Rightarrow} T \vdash Bew_T(\ulcorner\psi\urcorner) \overset{\dagger}{\Rightarrow} T \vdash \neg\psi$, $\lightning$ consistency of T. Therefore, $T \nvdash \psi$. (ii) suppose $T \vdash \neg\psi \overset{\dagger}{\Rightarrow} T \vdash Bew_T(\ulcorner\psi\urcorner) \Rightarrow$ (by *) $T \vdash \psi$, $\lightning$ (i) + consistency of T. Therefore, $T \nvdash \neg\psi$. QED

Consequences and Further Work

- Gödel's 2nd incompleteness theorem
- Tarski's undefinability of truth
- 'concrete' unprovability results in combinatorics and number theory, e.g. Paris&Harington(1978): 'Infinite Ramsey Theorem is unprovable' and Matijasevic(1970):'No decision procedure for diophantine equations'
- non-standard models
- similar incompleteness results for other systems with sufficient locomotive power:
  - self-reference/reflection
  - enough calculus
  - truth (?)

  Examples of such systems:
  - ZF or any other formal system of set theory
  - [Tarski,1968] Group Theory (!)

- (a sufficiently rich, consistent chunk of) NL (as an FL) is incomplete. Proof ideas:
  - self-reference/encoding by direct speech/quotation and (reflexive) pronouns
  - axioms: S (or IP or different sentence types)
  - inference rules = rewriting rules
  - calculus?

(Mis)interpretations

- "Logic is useless" (+ variations)
- "Manfred Eigen's *universal syntax* approach to the genetic foundations of biology is viable to the same incompletenss problem" see http://www.mitdenker.at/life/life04.htm
- [Neil Tennant, 2000] "Incompleteness theorems should have been discovered much earlier"
- [Hilary Putnam, 2000-2002] (Reinhard Muskens provided this reference one day after the talk.) "[Kripke's 1984 proof] establishes independence by means that could, in principle, have been understood by nineteenth-century mathematicians." This is hard to believe since the proof uses some game theory, a considerable lot of model theory (*upward persistence, non-standard models, ultra-powers*), Skolem functions, Gödel numbers and the quantificational hierarchy.
- [Bhupinder Singh Anand, 2002] "PA is ω-inconsistent", see http://alixcomsi.com/CTG_02.htm and updates

References

[1] Gödel, K. - 'Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I', *Monatshefte für Mathematik und Physik 38*, 173–198, 1931. English translation in J. van Heijenoort (ed.) *From Frege to Gödel*, Harvard UP, 1967

[2] Hofstadter, D. - *Gödel, Escher, Bach: an eternal golden braid*, Basic Books, 1979

[3] Putnam, K. - 'Nonstandard models and Kripke's proof of the Gödel Theorem', *Notre Dame Journal of Formal Logic 41:1*, 53 – 58, 2000 (appeared in 2002)

[4] Smoryński, C. - 'The Incompleteness Theorems', in J.Barwise (ed.) *Handbook of Mathematical Logic*, North-Holland, 1978

[5] Tarski, A., A. Mostowski, R. Robinson - *Undecidable Theories*, North-Holland, 1968

[6] Tennant, N. - 'Deductive vs expressive power: a pre-Gödelian predicament', *The Journal of Philosphy 97*, 257–277, 2000