Thm: There is a set $A \subseteq \mathbb{N}$, recursively enumerable, but not recursive

Proof: (1) $\exists f: \mathbb{N}\times\mathbb{N} \to \mathbb{N}$ recursive $(\forall g)_{\text{recursive}}\ \mathbb{N} \to \mathbb{N}\ (\exists n)\ f(n,\cdot)=g(\cdot)$

f - a universal recursive function - for every partial recursive function there is a (effectively) section equal to g

- We enumerate the recipes for recursive functions: $\alpha_0, \alpha_1, \dots$ (by algorithm)

$f(n,m) = [\alpha_n \text{ applied to } m]$

(2) ~~Let~~ $A = \{x \in \mathbb{N} : f(x,x) = 0\}$: rec. enumerable recursive

Proof: An algorithm, acts in steps, generates a list of nat numbers

[step i] we render i steps in calc. of $f(x,x)$ for all $x \leq i$

- whenever $f(x,x)$ is calculated, we add it to the list

We have enumerated A.

(3) A is not recursive:

Suppose $\chi_A \in Rec$, then $\chi_A(\cdot) = f(n,\cdot)$ for some n

$f(n,\cdot)$ is total

$f(n,n) = 0 \iff n \in A \iff \chi_A(n) \neq 0 \iff f(n,n) \neq 0$, a contradiction

Languages, coding, decidability

Let $L = \{P_i, f_j, c_t\}_{i,j,t}$ - a finite language

Formulas of L are computable objects - having a string of symbols we can decide whether it is indeed a formula

$$\mathcal{F}_L \subseteq \left(L \cup \{(,),x,\text{I},\text{\textbf{,}},\wedge,\neg,\exists\}\right)^*$$

$\uparrow$

$x_2 = x_{11}$

We will think of formulas as of numbers (Gödel coding)

$\longleftarrow p_2 = p_{11}$

$($, $)$, x, I, $\textbf{,}$, $\wedge$, $\neg$, $\exists$, $\forall$, p, f, c
1 2 3 4 5 6 7 8 9 10 11 12

Ex: $\forall x_3 \, P_1(x_3) \longleftarrow \varphi$

$\forall x_{III} \, P_1(x_{III}) \longleftarrow \varphi$ in our language

$\downarrow$

$\langle 9,3,4,4,4,10,4,1,3,4,4,4,2 \rangle \rightsquigarrow$ the Gödel code

$$2^9 \cdot 3^3 \cdot 5^4 \cdot 7^4 \cdot 11^4 \cdot 13^{10} \cdot 17^4 \cdot 23^1 \cdot 29^3 \cdot 31^4 \cdot 37^4 \cdot 41^4 \cdot 43^2 = \ulcorner \varphi \urcorner$$

Def. (1) $A \subseteq \mathcal{F}_L$ is recursive (computable) $\Leftrightarrow \{\ulcorner\varphi\urcorner : \varphi \in A\}$ recursive $\Leftrightarrow A$ is TM-computable

(2) $A \subseteq \mathcal{F}_L$ is recursively enumerable $\Leftrightarrow \{\ulcorner\varphi\urcorner : \varphi \in A\}$ is r.e $\Leftrightarrow A$ is TM-computable enum.

$\Leftrightarrow A = \emptyset$ or there is a total TM-computable $f : \mathbb{N} \to \mathcal{F}_L$ with

$A = f[\mathbb{N}]$

Example: $\{\varphi \in \mathcal{F}_L : \vdash \varphi\}$ is r.e.

We have a semi-algorithm
Given φ:
we write down all formal proofs (in KRL, L)
we look at their conclusions
if φ is a conclusion, we stop and answer yes

Def. $T \subseteq \mathcal{F}_L$ is decidable if T is recursive

Thm. If T is r.e. and complete, then T is decidable.

Peano arithmetic

Language $L_{PA} = \{+, \cdot, 0, S, <\}$ $PA \subseteq \underbrace{Th(\mathbb{N}, +, \cdot, 0, S, <)}_{TA\text{ - true arithmetic}}$

Classically: primitive notions: 0 (constant), S (successor)

Axioms: 1) $0 \neq Sx$
2) $Sx = Sy \Rightarrow x = y$
induction scheme 3) $\varphi(x, \dots)$ a formula in our lang. $[\varphi(0, \dots) \wedge \forall x (\varphi(x, \dots) \to \varphi(Sx, \dots)) \to \forall x\, \varphi(x, \dots)]$

+ a rule for introducing new functions symbols:

Suppose f, g - function symbols of suitable arities
Then we introduce a function symbol h and new axioms for it

$$\begin{cases} h(0, \bar{x}) = f(\bar{x}) \\ h(Sn, \bar{x}) = g(\bar{x}, n, h(n, \bar{x})) \end{cases}$$ (simple) recursion

Example: $+ : \begin{cases} 0 + x = x \\ S_n + x = S(n+x) \end{cases}$ $\quad \cdot : \begin{cases} 0 \cdot x = 0 \\ S_n \cdot x = n \cdot x + x \end{cases}$

PA1: $0 \neq S_x$
PA2: $S_x = S_y \to x = y$
PA3: the induction scheme
PA4 – PA7

PA4, PA5   PA6, PA7

· PA is undecidable (Rosser)
· PA is incomplete (Gödel)

Example: PA $\vdash$ $x + 0 = x$ $\varphi(x)$

- $\varphi(0)$ (by PA(4) for $x = 0$)
- Suppose $\varphi(x)$. We will show $\varphi(S_x)$

$x + 0 = x$ $S_x + 0 = S_x$

From PA5 we have $S_x + 0 = S(x+0) = S_x$
Now we apply PA3(φ).

Similarly: PA $\vdash$ $+, \cdot$ are associative, commutative, distributive and "$<$" $\underbrace{x < y \overset{df}{\iff} (\exists z)\, x + z = y}$ is correct

For $n \in \mathbb{N}$ let $\underline{n} = \underbrace{S \dots S}_{n} 0 \in J_{L_{PA}}$. It is called a numeral.

Lemma (representability of recursive sets and functions in PA)

(1) Assume $A \subseteq \mathbb{N}$ is recursive. Then there is a formula $\varphi_A(x) \in \mathcal{F}_{L_{PA}}$ s.t. $(\forall n \in \mathbb{N})$ $n \in A \Rightarrow PA \vdash \varphi_A(\underline{n})$ and $n \notin A \Rightarrow PA \vdash \neg \varphi_A(\underline{n})$

(2) Assume $f : \mathbb{N}^k \to \mathbb{N}$ rec. Then there is a formula $\varphi_f(\bar{x}, y) \in \mathcal{F}_{L_{PA}}$ s.t. $PA \vdash (\forall \bar{x})(\exists^{\leq 1} y)\, \varphi_f(\bar{x}, y)$ and $\forall \bar{n} \in \mathbb{N}^k$ (if $f(\bar{n})\downarrow$ then $PA \vdash \varphi_f(\underline{\bar{n}}, f(\bar{n}))$)

consider χ_A

Proof: Enough to prove (2).

Induction on the length of def of f

1° the basic functions: obvious

2° composition scheme: obvious

minimum operator: $f(\bar{n}) = \min\{y : g(\bar{n}, y) = 0\}$

ind. hyp.

$\varphi_f(\bar{x}, y)$ represented by $\varphi_g(\bar{x}, y, z)$

$\varphi_g(\bar{x}, y, 0) \wedge (\forall y' < y)((\exists z)\, \varphi_g(\bar{x}, y', z) \wedge \neg \varphi_g(\bar{x}, y', 0))$

$\varphi_f(\bar{x}, y)$ represents f.

(a) Assume $f(\bar{n})\downarrow = k$ then $g(\bar{n}, k) = 0$ and $(\forall k' < k)\, g(\bar{n}, k')\downarrow \neq 0$

$PA \vdash \varphi_g(\underline{\bar{n}}, \underline{k}, \underline{0}) \wedge (\forall y' < \underline{k})((\exists z)\, \varphi_g(\underline{\bar{n}}, y', z) \wedge \neg\varphi_g(\underline{\bar{n}}, y', 0))$

Fact: $P \vdash x < \underline{k} \Leftrightarrow (x = \underline{0} \vee \ldots \vee x = \underline{k-1})$

(b) $PA \vdash (\exists^{\leq 1} y)\, \varphi_f(\bar{x}, y) \leftarrow$ ex.

(d) recursion scheme:

$\begin{cases} h(0, \bar{x}) = f(\bar{x}) \\ h(Sn, \bar{x}) = g(\bar{x}, n, h(n, \bar{x})) \end{cases}$

We have φ_f, φ_g rep. f, g

We want φ_h rep. h.

Trick: coding sequences

Idea: $h(n, \bar{x}) = m \iff (\exists \langle a_0, \ldots, a_n \rangle) \begin{cases} a_0 = f(\bar{x}) \\ (\forall i < n)[a_{S i} = g(\bar{x}, i, a_i)] \\ a_n = m \end{cases}$

We use the Chinese remainder thm to make it quantify over 1 thing