1 Computability

1.1 Recursive Functions

Definition 1.1. The class of **primitive recursive** functions is the smallest class of functions $f : \mathbb{N}^n \to \mathbb{N}$ s.t. they contain

a. $s(x) = x + 1$

b. $P^n_m(x_1, \dots, x_n) = x_m$

c. $C^n_m(x_1, \dots, x_n) = m$

and is closed under

a'. Composition: $f(x_1, \dots, x_n) = h(g_1(x_1, \dots, x_m), \dots, g_m(x_1, \dots, x_m))$

b'. Primitive recursive rule:

$$f(0, x_1, \dots, x_n) = g(x_1, \dots, x_n)$$
$$f(n + 1, x_1, \dots, x_n) = h(f(n, x_1, \dots, x_m), n, x_1, \dots, x_n)$$

Definition 1.2. A **primitive recursive derivation** is a list $f_0, \dots, f_n$ s.t. each f_i is either a.-c. above or defined from previous f_j using a'., b'.

Example.

1. Addition

$$f(0, y) = P^1_1(y)$$
$$f(n + 1, y) = S(f(n, y))$$

2. Multiplication

$$f(0, y) = 0 = C^1_0(y)$$
$$f(n + 1, y) = f(n, y) + y$$

3. Exponentiation

$$f(0, y) = 1$$
$$f(n + 1, y) = f(n, y) \cdot y$$

4. Factorial

$$f(0, y) = 1$$
$$f(n + 1, y) = f(n) \cdot (n + 1)$$

5. Predcessor

$$f(0) = 0$$
$$f(n + 1) = n$$

6. Subtraction

$$f(0, y) = y$$
$$f(n+1, y) = \text{Pred}(f(h, y))$$

7. SG

$$\text{Sg}(0) = 0$$
$$\text{Sg}(n+1) = 1$$

8. Equality

$$\text{Eq}(x, y) = \bar{\text{Sg}}((x-y) + (y-x))$$
$$\text{Ineq}(x, y) = \text{Sg}(\text{Eq}(x, y))$$

9. Minimum

$$\min(x, y) = \text{Eq}(x, y) \cdot x + \text{Ineq}(x, y) \cdot [x \cdot \bar{\text{Sg}}(x-y) + y \cdot \bar{\text{Sg}}(y-x)]$$
$$\min(x_1, \dots x_n) = \min(\min_{n-1}(x_1, \dots, x_n), x_n)$$

10. Maximum: same as min but replace each $\bar{\text{Sg}}$ with just Sg.

11. Summation of some $f(x, i)$.

$$g(x, 0) = f(x, 0)$$
$$g(x, n+1) = g(x, n) + f(x, n+1)$$

12. Product of some $f(x, i)$.

$$g(x, 0) = f(x, 0)$$
$$g(x, n+1) = g(x, n) \cdot f(x, n+1)$$

Denote by $\bar{x} = x_1, \dots, x_n$.

Definition 1.3. A **relation** is primitive recursive if its characteristic function f is primitive recursive.

$$f(\bar{x}) = \begin{cases} 1 \text{ if } R(\bar{x}) \\ 0 \text{ if } \neg R(\bar{x}) \end{cases}$$

Let pr stand for primtive recursive.

Theorem 1.1. Let P_1, P_2 be pr relations and $f_1, \dots, f_n$ be pr functions. Then the following are pr:

i. $R(\bar{x}) = \neg P(\bar{x})$

ii. $R(\bar{x}) = P_1(\bar{x}) \wedge P_2(\bar{x})$, similarly for all other logical connectives.

iii. $R(\bar{x}) = P_1(f_1(\bar{x}), \dots, f_n(\bar{x}))$

iv. $R(t, \bar{x}) = \exists i \leq t(P_1(i, \bar{x}))$

v. $R(t, \bar{x}) = \forall i \leq t(P_1(i, \bar{x}))$

Proof. A sketch.

i. Use $\bar{\mathrm{Sg}}(f_p(\bar{x}))$.

ii. Use $f_{P_1}(\bar{x}) \cdot f_{P_2}(\bar{x})$.

iii. Use $f_{P_1}(f_1(\bar{x}), \dots, f_n(\bar{x}))$.

iv. Use $\mathrm{Sg}(\sum f_{P_1}(i, \bar{x}))$.

v. Use $\prod f_{P_1}(i, \bar{x})$.

$\square$

Corollary 1.2. Two items.

a. x divides y evenly: $\exists i \leq y(x \cdot i = y)$

b. x is prime: $\forall i \leq x(i \mid x \to (i = 1) \vee (i = x))$

Theorem 1.3. Let $f_1, \dots, f_n$ and $P_1, \dots, P_n$ be pr. Suppose that

1. For all $\bar{x}$, $P_1(\bar{x}) \vee \cdots \vee P_n(\bar{x})$ is true.
2. For no $\bar{x}$, $i \neq j < n$, $P_i(\bar{x}) \wedge P_j(\bar{x})$ is true.

Then,

$$g(\bar{x}) = \begin{cases} f_1(\bar{x}) \text{ if } P_1(\bar{x}) \\ \vdots \\ f_n(\bar{x}) \text{ if } P_n(\bar{x}) \end{cases}$$

is pr. Essentially, primitive recursion is closed under comprehensive cases.

Proof.

$$g(\bar{x}) = f_{P_1}(\bar{x}) f_1(\bar{x}) + \cdots + f_{P_n}(\bar{x}) f_n(\bar{x})$$

$\square$

Definition 1.4. $\mu_i P(\bar{x})$ is an operator to mean the least i such that $P(\bar{x})$. Use $\mu_i \leq t$ to denote the least i such that $i \leq t$.

Theorem 1.4. Any function $f(\bar{x}, n) = (\mu_i \leq n)(P(\bar{x}, i))$ is pr.

Proof. Define f in the following way:

$$f(\bar{x}, 0) = 0$$
$$f(\bar{x}, n+1) = \begin{cases} f(\bar{x}, n) & \text{if } \exists y \leq n(P(\bar{x}, i)) \\ n+1 & \text{if } \neg\exists i \leq n(P(\bar{x}, i)) \wedge P(\bar{x}, n+1) \\ 0 & \text{else} \end{cases}$$

$\square$

Proposition 1.5. The following are pr:

1. Let $a = nb + r$ such that $r < b$.
   a) $\mathrm{div}(a, b) = a$
   b) $\mathrm{rem}(a, b) = a - \mathrm{div}(a, b) \cdot b$

2. $p(i) = i$th prime number

Proof. Define div and p by the following.

$$\text{div}(a, b) = \mu_n \leq a(a - nb < 1)$$

$$\begin{aligned} p(0) &= 2 \\ p(n+1) &= (\mu_i \leq (p(n)! + 1))(\text{prime}(i) \wedge p(n) < i) \end{aligned}$$

□

Remark. $p_i = i$th prime is 1-indexed by

$$p_i = \begin{cases} 0 \text{ for } i = 0 \\ p(i-1) \text{ for } i > 0 \end{cases}$$

Definition 1.5. We may define a function using *course of values recursion* where we may use $f(n-k)$ in addition to $f(n)$. That is, we may define a recursive function f by

$$\begin{aligned} f(0, \bar{x}) &= g(\bar{x}) \\ f(n+1, \bar{x}) &= h(f(n, \bar{x}), f(n-1, \bar{x}), \ldots, f(0, \bar{x}), n, \bar{x}) \end{aligned}$$

To make the above definition more concrete, we may code the course of values of f into a new function G and then use G to define f.

$$\begin{aligned} G(0, \bar{x}) &= p_1^{g(\bar{x})} \\ G(n+1, \bar{x}) &= G(n, x) p_{n+2}^{h(\dagger)} \end{aligned}$$

where † denotes the next prime power that makes a legitimate Gödel encoding. So then f may be defined as

$$f(n, \bar{x}) = \mu_i \leq G(n, \bar{x})[p(n)^{i+1} \nmid G(n, \bar{x})]$$

For example,

$$\begin{aligned} G(0, x) &= 2^{g(x)} = 2^{f(0)} \\ G(1, x) &= 2^{f(0)} \cdot 3^{f(1)} \end{aligned}$$

So f may be defined by "pullilng down" those exponents of prime numbers that make up G.

Theorem 1.6. *(the coding theorem)* There is an injection $f \colon \mathbb{N}^{\prec\omega} \to \mathbb{N}$ such that

1. The image of f is pr, that is,

$$\text{seq}(z) \Leftrightarrow \exists n_1, \ldots, n_m [f(n_1, \ldots, n_m) = z]$$

2. For each n, $f_n(\bar{x}) = f(\bar{x})$ is pr.

3. There exists pr functions

   a) $\ln(x)$ such that $\ln(f(x_1, \ldots, x_m)) = m$ (length).
   b) $\text{proj}(x, i)$ such that $\text{proj}(f(x_1, \ldots, x_m), i) = x_i$ (projection).
   c) $h(x, y)$ such that $h(f(\bar{x}), f(\bar{y})) = f(\bar{x}, \bar{y})$, or simply $[\bar{x} * \bar{y}]$ (concatentation).

Proof. Define $f(n_1, \ldots, n_m) = p_1^{n_1+1} \cdot p_2^{n_2+1} \cdot \cdots \cdot p_m^{n_m+1}$. We add 1 to each exponent for the $n_i = 0$ case.

1. We redefine $\text{seq}(z)$ by

$$\exists m \leq z[(\forall i \leq m, p_i \mid z) \wedge (\forall j \leq z, m < j \to p_j \nmid z)]$$

   That is, there is a substring of z where the first m primes divide z and nothing else divides z.

2. Use the following:

$$f_1(n_1) = 2^{n_1+1}$$
$$f_{m+1}(n_1, \ldots, n_{m+1}) = f_m(n_1, \ldots, n_m) p_{m+1}^{n_{m+1}+1}$$

3. a)

$$\ln(z) = \begin{cases} 0 \text{ if } \neg\text{seq}(z) \\ \mu_i \leq z[p_i \mid z \wedge p_{i+1} \nmid z] \text{ else} \end{cases}$$

   b)

$$\text{proj}(z, i, =) \begin{cases} 0 \text{ if } \neg\text{seq}(z) \text{ or } (\text{seq}(z) \wedge \ln(z) < i) \\ (\mu_j \leq n[p_i^j \nmid n]) - 2 \text{ else} \end{cases}$$

   c)

$$h(x, y) = \begin{cases} 0 \text{ if } \neq \text{seq}(x) \vee \neg\text{seq}(y) \\ x \cdot \prod_{j \leq \ln(y)} p_{j+\ln(x)}^{\text{proj}(y,i,+)1} \end{cases}$$

To illustrate part c) of the proof,

$$x = f(3, 4, 4) = 2^4 \cdot 3^5 \cdot 5^5$$
$$y = f(1, 7) = 2^2 \cdot 3^8$$

so, concatenating them,

$$h(x, y) = 2^4 \cdot 3^5 \cdot 5^5 \cdot 7^2 \cdot 11^8$$

□

Definition 1.6. Let $P(\bar{x}, y)$ be a predicate. We say P is *regular* if $\forall \bar{x} \exists y P(\bar{x}, y)$.

Definition 1.7. A function is μ-recursive if it can be generated from primitives S (successor), C_m^n, P_m^n by composition, primitive recursion, and applying μ to regular predicates.

Theorem 1.7. Any primitive recursive function is also μ-recursive. Also, the same proofs for the primitive recursive functions prove that μ-recursion is closed under

1. $\neg, \wedge\vee, \to, \Leftrightarrow, \forall i \leq t, \exists i \leq t$,
2. taking cases using μ-recursive functions and predicates,
3. and the bounded μ operator.

Theorem 1.8. There are functions which are not μ-recursive.

Proof. By induction on the length of μ-recursive derivations, we show there are only countably many μ-recursive functions. But $\{f \mid f\colon \mathbb{N} \to \mathbb{N}\} \succ \mathbb{N}$. $\square$

Example. Here's the Ackerman function:

$$f_n(x, 0) = S(x)$$
$$f_n(x, y+1) = f_{n-1}(f_n(x, y), x)$$

The idea behind the Ackerman function is that it generalizes from S to $+$ to $\cdot$ to exponentiation etc. Its μ-recursive but not pr. A primitive recursive function can't keep track of how many times we've composed the preceding operators.

1.2 Turing Machines

Definition 1.8. An *alphabet* A is a finite list of symbols. Preserve b for blanks.

Definition 1.9. A *tape* on A is a function $\tau\colon \mathbb{N} \to A \cup \{b\}$ such that $\tau(n) = b$ for all but finitely many n.

Definition 1.10. A *Turing machine* is a $\langle A, S, \text{beg}, T$ where

- A is an alphabet
- S is a finite set of natural numbers (states)
- $\text{beg} \in S$ (start state)
- T is a set of $\langle s, x, y, m, s' \rangle$ where s, s' are states, x, y are in $A \cup \{b\}$, and $m \in \{-1, 0, 1\}$. Further, no T_1, T_2 start with the same s, x and every s, x is in some T. In other words, T is our transition function. Think of -1 as going left and 1 as going right.

Definition 1.11. A state is *halting* if for every $x \in A \cup \{b\}$, we have $\langle s, x, x, 0, s \rangle \in T$.

Definition 1.12. A *configuration* of a machine is a triple $\langle \tau, s, i \rangle$ where τ is a tape, s is a state, and i is a natural number.

Definition 1.13. A *computation* is a sequence of configurations $\langle \tau_0, s_0, i_0 \rangle, \langle \tau_1, s_1, i_1 \rangle, \ldots$ such that, given $\langle \tau_i, s_i, i_i \rangle$, any $j \in T$ is of the form $\langle s, x, y, m, s' \rangle$ such that $s = s_i$ and $x = \tau_i(i_i)$. Futher,

$$\tau_{i+1}(n) = \begin{cases} \tau_i(n) & \text{if } n \neq i_i \\ y & \text{else} \end{cases}$$
$$s_{i+1} = s'$$
$$i_{i+1} = i_i + m$$

1.3 Coding Turing Machines

2 Incompleteness

2.1 Incompleteness and the Undefinability of Truth

Some notation:

- For R μ-recursive, let $\tilde{R}$ be the formula used to arithmetically express it.
- For a number n, $\tilde{n} = \underbrace{0 + 1 + \cdots + 1}_{n \text{ times}}$

Theorem 2.1. *(first incompleteness theorem)* Let Γ be a recursive set of sentences. Suppose $\mathbb{N} \models \Gamma$. Then Γ is incomplete.

Proof. Suppose note. Let Γ be such that

1. Γ is recursive
2. $\mathbb{N} \models \Gamma$
3. Γ is complete

We claim that $\forall \varphi, \mathbb{N} \models \varphi \Leftrightarrow \Gamma \vdash \varphi$. For $\Leftarrow$, we have that $\Gamma \vdash \varphi$ since $\mathbb{N} \models \Gamma$, then $\mathbb{N} \models \varphi$ by soundness. For $\Rightarrow$, suppose that $\Gamma \nvdash \varphi$. Then, by 3. above, $\Gamma \vdash \neg\varphi$. So $\mathbb{N} \models \neg\varphi$ by soundness. So $\mathbb{N} \not\models \varphi$.

So $\exists y T(e,n,y) \Leftrightarrow \Gamma \vdash \exists y \tilde{T}(\tilde{e},\tilde{n},\tilde{y})$. Define $g(e,n)$ by

$$g(e,n) = \mu_j[\mathrm{Proof}(\Gamma, j) \wedge ((\mathrm{proj}(j, \ln(j)) = \ulcorner \exists y \tilde{T}(\tilde{e},\tilde{n},y) \urcorner \vee \mathrm{proj}(j,\ln(j)) = \ulcorner \neg\exists y \tilde{T}(\tilde{e},\tilde{n},x)\urcorner))]$$

So define $f(e,n)$ by

$$f(e,n) = \begin{cases} 1 & \text{if } \mathrm{proj}(g(e,n), \ln(g(e,n)) = \ulcorner \exists y \tilde{T}(\tilde{e},\tilde{n},y)\urcorner, \\ )0 & \text{else} \end{cases}$$

But this solves the halting problem recursively for a contradiction. $\square$

Lemma 2.2. There is a primitive recursive function $s(f,x)$ such that if $f = \ulcorner \varphi(v_0) \urcorner$ and x is a number,

$$s(f,x) = \ulcorner \exists v_0[(v_0 = \tilde{x}) \wedge \varphi(v_0)] \urcorner$$

Note that $\exists v_0[(v_0 = \tilde{x}) \wedge \varphi(v_0)] \equiv \varphi(\tilde{x})]$.

Proof. Define $\mathrm{num}(x)\colon \mathbb{N} \to \mathbb{N}$ such that $\mathrm{num}(0) = \ulcorner 0 \urcorner$ and $\mathrm{num}(n+1) = \mathrm{num}(n) * \ulcorner +1 \urcorner$. So $\mathrm{num}(n) = \ulcorner \tilde{n} \urcorner$. Then

$$s(f,x) = \begin{cases} 0 & \text{if } \neg\mathrm{Formula}(f) \vee \neg\mathrm{Free}(y, v_0) \vee \exists i < f[(i \neq \ulcorner v_0 \urcorner) \wedge \mathrm{Free}(f,i)] \\ \ulcorner \exists v_0(v_0 = \urcorner * \mathrm{num}(x) * \ulcorner \wedge \urcorner * f * \ulcorner) \urcorner & \text{else} \end{cases}$$

$\square$

Let

$$\mathrm{Truth}_{\mathbb{N}}(x) \Leftrightarrow x \text{ is the code of a sentence } \varphi \text{ and is true}$$

Theorem 2.3. *(Tarski)* $\mathrm{Truth}_{\mathbb{N}}(x)$ is not arithmetical.

Proof. Suppose not. Define $R(s) \Leftrightarrow \exists w[S(s,s) = w \wedge \neg\mathrm{Truth}_{\mathbb{N}}(w)]$. Let e be the code of $\tilde{R}$. Then

$$s(e,e) = \exists w(S(\tilde{e},\tilde{e}) = w \wedge \neg\mathrm{Truth}_{\mathbb{N}}(w))$$

So

$$\begin{aligned} \mathbb{N} \models \exists w[\tilde{S}(\tilde{e},\tilde{e}) = w \wedge \neg\widetilde{\mathrm{Truth}_{\mathbb{N}}}(w)] \\ \Leftrightarrow \mathbb{N} \models \neg\mathrm{Truth}_{\mathbb{N}}(\widetilde{S(e,e)}) \\ \Leftrightarrow \mathbb{N}\neg \models \exists w[(\tilde{S}(\tilde{e},\tilde{e}) = w) \wedge \neg\mathrm{Truth}_{\mathbb{N}}(w)] \end{aligned}$$

For a contradiction. $\square$

2.2 The Gödel-Rosser Incompleteness Theorem

Our first proof of the incompleteness theorem involved the requirement $\mathbb{N} \models \Gamma$, i.e. Gödel assumed ω-consistency. Some problems with this:

1. $\mathbb{N} \models \varphi$ is very complicated, and it is not arithmetical per Tarski, so it's not recursive.
2. What about theories when $\mathbb{N} \not\models \Gamma$? For example, suppose $\mathbb{N} \models \Gamma$. We apply the incompleteness theorem to find a φ such that $\Gamma \not\vdash \varphi$ and $\Gamma \not\vdash \neg\varphi$. This implies that $\Gamma \cup \{\varphi\}$ and $\Gamma \cup \{\neg\varphi\}$ are consistent. Can $\Gamma \cup \{\neg\varphi\}$ be complete? If so, that'd be weird.

Note that we needed $\mathbb{N} \models \Gamma$ for 2 things:

1. Γ's consistency,
2. Ensuring pr predicates get coded.

We're allowed 1. since it's what we're interested in assuming, but 2. is harder.

Definition 2.1. A formula $\varphi(x_1, \ldots, x_n)$ **numeralwise represents a relation** $R(x_1, \ldots, x_n)$ in a theory T if

$$R(n_1, \ldots, n_n) \implies T \vdash \varphi(\tilde{n_1}, \ldots, \tilde{n_n})$$
$$\neg R(n_1, \ldots, n_n) \implies T \vdash \neg\varphi(\tilde{n_1}, \ldots, \tilde{n_n})$$

A formula $\varphi(x_1, \ldots, x_n, y)$ **numeralwise represents a function** $f\colon \mathbb{N}^n \to \mathbb{N}$ if

$$f(n_1, \ldots, n_n) = m \implies (T \vdash \varphi(\tilde{n_1}, \ldots, \tilde{n_n}) = m) \wedge (T \vdash \forall x(\varphi(\tilde{n_1}, \ldots, \tilde{n_n}) \to x = \tilde{m}))$$

Usually T can't prove $\forall x_1, \ldots, x_n, y, y(\varphi(x_1, \ldots, x_n, y) \wedge \varphi(x_1, \ldots, x_n, y') \to y' = y)$. This is an advantage of numeralwise representability; we can prove this for specific not quantified values.

Definition 2.2. We say T *understands* $<$ If

1. If $i < j$, then $T \vdash \tilde{i} < \tilde{j}$
2. For any a, $T \vdash \forall x(x < \tilde{a} \to x = 0 \vee x = 1 \vee \cdots \vee x = a - 1)$
3. For any a, $T \vdash \forall x(x \leq \tilde{a}$ or $\tilde{a} \leq x)$

Definition 2.3. We say T is *adequate* if

1. T numeralwise represents all recursive functions
2. T understands $<$

Note that if T is adequate and $T \subseteq T'$, then T' is adequate.

Definition 2.4. We say a set A is Σ_1 if it is semirecursive. In other words, for some recursive P, we have $x \in A \Leftrightarrow \exists y P(x, y)$

Lemma 2.4. Let T be adequate. Let A, B be disjoint Σ_1 sets. Then there is a formula $\varphi(v)$ such that

1. $x \in A \Rightarrow T \vdash \varphi(\tilde{x})$
2. $x \in B \Rightarrow T \vdash \neg\varphi(\tilde{x})$

Proof. Let $x \in A \Leftrightarrow \exists y P(x,y)$ and $x \in B \Leftrightarrow \exists y Q(x,y)$ since A, B are semirecursive. Let $\widetilde{P}, \widetilde{Q}$ numeralwise represent P, Q in T. We define

$$\varphi(v) \equiv \exists y[\widetilde{P}(v,y) \wedge \forall i(i < y \rightarrow \neg\widetilde{P}(v,i) \wedge \neg\widetilde{Q}(v,i))]$$

1. If $x \in A$, then $\exists y P(x,y)$. Let y_0 be the least such y. Then we know

$$T \vdash \widetilde{P}(\tilde{x}, \tilde{y_0}), T \vdash \neg\widetilde{Q}(\tilde{x}, \tilde{y_0})$$

$$\forall i < y_0, T \vdash \neg\widetilde{P}(\tilde{x}, \tilde{i}) \wedge \neg\widetilde{Q}(\tilde{x}, \tilde{i})$$

   By T understands $<$, then

$$T \vdash \forall i(i < \tilde{y_0} \rightarrow \neg\widetilde{P}(\tilde{x}, i) \wedge \neg\widetilde{Q}(\tilde{x}, i))$$

   Hence, $T \vdash \varphi(\tilde{x})$ as desired.

2. If $x \in B$, then $\exists y Q(x,y)$. Let y_0 be the least such y. Then

$$\begin{aligned} \forall i < y_0[T \vdash \neg\widetilde{P}(\tilde{y}, \tilde{i})] & \qquad \text{by numrep} \\ \Rightarrow T \vdash \forall i(i \leq \tilde{y_0} \rightarrow \neg P(\tilde{x}, i)) & \qquad \text{by understands } < \end{aligned}$$

   Next,

$$\begin{aligned} T \vdash \widetilde{Q}(\tilde{x}, \tilde{y_0}) & \qquad \text{by numrep} \\ \Rightarrow T \vdash (\tilde{y_0} < i \rightarrow \neg\forall j < i(\neg P(\tilde{x}, j) \wedge \neg Q(\tilde{x}, i))) & \qquad \text{by logic} \end{aligned}$$

   Finally,

$$\begin{aligned} T \vdash \forall i(i \leq \tilde{y_0} \vee i \geq \tilde{y_0}) & \qquad \text{by understands} \\ \Rightarrow T \vdash \neg\varphi(\tilde{x}) & \qquad \text{by logic} \end{aligned}$$

□

Lemma 2.5. Let T be recursive and complete. Then $A = \{\ulcorner\varphi\urcorner \mid T \vdash \varphi\}$ is recursive.

Proof. **Case 1**: T is inconsistent. This is trivial since T proves everything, so A is just every sentence.
Case 2: T is consistent. Then define $g\colon \mathbb{N} \rightarrow \mathbb{N}$ by

$$g(x) = \begin{cases} 0 & \text{if } \neg Sentence(x) \\ \mu_y[Proof_T(x,y) \vee Proof_T(2^2 \cdot x, y)] & \text{else} \end{cases}$$

Then $x \in A \Leftrightarrow Sentence(x) \wedge x = \mathrm{Proj}(g(x), \mathrm{len}(g(x)))$. Essentially, we're checking whether the code of the last line of the proof is x. □

Lemma 2.6. Let T be recursive and adequate. Then A is not recursive.

Proof. Let A, B be recursively inseperable Σ_1 sets. Let $\varphi(v)$ be as in the proof of Lemma 2.4. Suppose $C = \{\varphi \mid T \vdash \varphi\}$ is recursive. Then $C' = \{x \mid T \vdash \varphi(\tilde{x})\}$ is also recursive, which contradicts the assumption that A, B are recursively inseperable. □

Theorem 2.7. (Gödel -Rosser 1) Let T be recursive and adequate. Then T is incomplete.

Proof. By the last 2 lemmas. □

2.3 Diagonalization

Lemma 2.8 (Diagonalization lemma)**.** Let T be adequate. Then for any formula $\varphi(v_0)$, there is a sentence G such that $T \vdash G \leftrightarrow \varphi(\widetilde{[G]})$

Remark. If we can find an adequate and true T, then we get $\mathbb{N} \models G \leftrightarrow \varphi(\widetilde{[G]})$

Proof. Recall that we have a pr function $S(f,x)$ such that if f is the code of $\varphi(v_0)$, then

$$S(f,x) = [\exists v_0(v_0 = \tilde{x} \wedge \varphi(v_0))]$$

Given this, we define

$$D(x) = S(x,x)$$

Which is to say substitution, D for diagonalization. So for any $\varphi(v_0)$,

$$D([\varphi(v_0)]) = [\exists v_0(v_0 = \widetilde{[\varphi(v_0)]} \wedge \varphi(v_0))]$$

Let $\varphi(v_0)$ be arbitrary. We let

$$\varphi^d = \exists y(\tilde{D}(v_0, y) \wedge \varphi(y))$$

Let $n_0 = [\varphi^d]$. Then let G be

$$\exists v_0(v_0 = \tilde{n_0} \wedge \exists y(\tilde{D}(v_0,y) \wedge \varphi(y)))$$

Notice that

1. G is a sentence.
2. Since $n_0 = [\varphi^d]$, $[G] = S(n_0, n_0) = D(n_0)$. I.e., G is the diagonalization of φ^d.
3. $G \leftrightarrow \exists y(\widetilde{D}(\tilde{n_0}, y) \wedge \varphi(y))$

Now, let $n_1 = [G]$. So $n_1 = D(n_0)$. Since T numeralwise represents all recursive functions,

$$T \vdash \forall y(\widetilde{D}(\tilde{n_0}, y) \leftrightarrow y = \tilde{n_1})$$

This means

$$T \vdash G \leftrightarrow \exists y(D(n_0, y) \wedge \varphi(y))$$

by 3. and logic, which implies

$$T \vdash G \leftrightarrow \varphi(\tilde{n_1})$$

$$\implies T \vdash G \leftrightarrow \varphi(\widetilde{[G]})$$

□

Theorem 2.9 (first incompleteness theorem)**.**

Proof. Suppose T is true and adequate. Then, let $Proof_T(x,y)$ hold if x is the code of a formula and y is the code of a proof of that formula. Note that this implies this is pr. So by diagonalization, we can find some G such that

$$T \vdash G \leftrightarrow \neg\exists y(\widetilde{Proof_T}([\tilde{G}], y))$$

So, suppose $T \vdash G$. Then, for m the code of the relevant proof, $T \vdash \widetilde{Proof_T}([\tilde{G}], n)$. So,

$$T \vdash \exists y \widetilde{Proof_T}([\tilde{G}], y)$$

But, by $T \vdash G$ and above,

$$T \vdash \neg\exists y(\widetilde{Proof_T}([\tilde{G}], y))$$

By contradiction, $T \nvdash G$. But, since T is sound, $T \nvdash \neg G$ because $\neg G$ is true. So T is incomplete. □

Theorem 2.10 (Tarski's undefinability of truth)**.**

Proof. Suppose there is a formula $\varphi(v_0)$ such that

$$\mathbb{N} \models \chi \leftrightarrow \mathbb{N} \models \varphi(\widetilde{[\chi]})$$

for any χ. Then, diagonalize on $\neg\varphi(v_0)$. We get

$$G \leftrightarrow \neg\varphi(\widetilde{[G]})$$

for a contradiction. So there is no such $\varphi(v_0)$. $\square$

Lemma 2.11. Let T be consistent and adquate. Then we have that $A = \{[\varphi] \mid T \vdash \varphi\}$ is not numeralwise representable in T.

Proof. Suppose $\tilde{A}$ defines A in T. Using diagonalization, we get a sentence G such that

$$T \vdash G \leftrightarrow \neg\tilde{A}([G])$$

Let $n_1 = [G]$. Then,

$$\begin{aligned} T \vdash G &\implies T \vdash \neg\tilde{A}([G]) \\ &\implies T \nvdash G \end{aligned}$$

by numeralwise representability. Similarly,

$$\begin{aligned} T \nvdash G &\implies n_1 \notin A \\ &\implies T \vdash \neg\tilde{A}(n_1) \\ &\implies T \vdash G \end{aligned}$$

For a contradiction in either case. $\square$

Corollary 2.12. Any T that is recursive, consistent, and adequate is incomplete.

Proof. T recursive and consistent means $\{[\varphi] \mid T \vdash \varphi\}$ by an old lemma, so T being recurisve, complete, and adequate implies

$$\{[\varphi] \mid T \vdash \varphi\}$$

is numeralwise representable in T by adequateness implying numeralwise representability. By the last lemma, T being recursive, complete, and adequate implies T is inconsistent. $\square$

2.4 Gödel's original paper

This was originally a "by hand" diagonalization of $\neg\exists y Proof_T(x, y)$.

1. The T in question was not a modern system but a variant of the system in Russell & Whitehead's *Principia.*

2. Gödel knew how to prove, in our terms, that $T \nvdash G$ although G is true. He did not know how to get $T \nvdash \neg G$ without using $\mathbb{N} \models T$. What he assumed was ω-consistency.

Definition 2.5. We say T is ω-consistent if there is no formula φ such that

$$T \vdash \neg\varphi(1), T \vdash \neg\varphi(2), \ldots$$

But $T \vdash \exists x\varphi(x)$ (T doesn't know that 1, 2, etc are the only natural numbers).

What Gödel proved is that if *Principa* (PM) is ω-consistent, then PM $\nvdash \neg G$.
Rosser's original proof was concerned with:

1. Can we get rid of all uses of $\mathbb{N} \models T$ (replace with consistent, adequate, etc.)?
2. Can we get a G such that $T \nvdash G$ and $T \nvdash \neg G$ without invoking ω-consistency?

This involved using a slightly different sentence from Gödel.

$\Psi(v_0) \equiv \forall z(\exists y(S(v_0, v_0, y) \wedge Proof(y, z) \to \exists z' < z(\exists y, y'(S(v_0, v_0, y) \wedge (y' = [\neg]*y) \wedge Proof(y', z')))))$

Informally, "if you can prove me, then there is a shorter proof of my negation".

Example. Some examples of things that are adequate, consistent, etc.
The basic system (B).

1. $$\forall x(x + 1 \neq 0)$$ $$\forall x \forall y(x + 1 = y + 1 \to x = y)$$
2. $$\forall x(x + 0 = x)$$ $$\forall x \forall y(x + (y + 1) = (x + y) + 1)$$
3. $$\forall x(x \cdot 0 = 0)$$ $$\forall x \forall y(x \cdot (y + 1) = x \cdot +1)$$

Robinson's system (R_0). If B is the set of axioms in the basic system,

$$R_0 = B \cup \forall x(x = 0 \vee \exists y(y + 1 = x))$$

Basically, natural numbers don't start at different places.

Peano arithmetic (PA).

$$PA = R_0 \cup \{\varphi(0, \bar{y}) \wedge \forall x(\varphi(x, \bar{y}) \to \varphi(x + 1, \bar{y})) \to \forall x \varphi(x, \bar{x}) \mid y \in \mathcal{L}\}$$

3 The Arithmetic Hierarchy