Computability

Mike Antonenko

Based on lectures by Svetlana Puzynina

Typeset on March 6, 2021

Contents

Here starts the lecture #1, from February 11, Thursday.

Note

For the rest of this course, $\mathbb{N}$ contains zero. For any set S, we think that $S^0 = \{0\} = \{\varnothing\}$.

Partial recursive functions

Definition. Let $f\colon \subseteq\mathbb{N}^k \to \mathbb{N}$ be a partial function. f is *simplistic*, iff

1. $f(x) = 0$ (*zero*, $f =: 0$).
2. $f(x) = x + 1$ (*successor*, $f =: s$).
3. $f(x_1, \ldots, x_n) = x_m$ (*projection*, $f =: I^n_m$)

Definition. There are several operations with functions $\mathbb{N}^k \to \mathbb{N}$, each of which we assign a letter.

The *composition operator* S. If we have $h(y_1, \ldots, y_m)$ and $g_i(x_1, \ldots, x_n)$, $i = 1, \ldots, n$, we define their *composition* f as

$$f = h\big(g_1(x_1, \ldots, x_n), \ldots, g_m(x_1, \ldots, x_m)\big).$$

The *primitive recursion* operator R. f of arity $n+1$ is defined with g and h of arities n and $n+2$ as

$$\begin{aligned} f(x_1, \ldots, x_n, 0) &= g(x_1, \ldots, x_n) \\ f(x_1, \ldots, x_n, y+1) &= h\big(x_1, \ldots, x_n, y, f(x_1, \ldots, x_n, y)\big). \end{aligned}$$

f is said to be a *primitive recursive* function, iff there exists $f_1, \ldots, f_k$ — a sequence of functions such that f_i is either simplistic, or gotten from $f_1, \ldots, f_{i-1}$ with the help of S and R; and $f_k = f$.

Example. $f(x, y) = x + 1$ is primitive recursive:

$$\begin{cases} f(x, 0) = x = I_1^1(x), \\ f(x, y+1) = (x+y) + 1 = s\Big(f(x, y)\Big) = s\Big(I_3^3(x, y, f(x, y))\Big), \end{cases}$$

so we can put $g = I_1^1$ and $h = s \circ I_3^3$ in the definition above.

Lemma. The following are primitive recursive:

1. Constants.
2. Binary sums, products, powers.
3. $[x \neq 0]$.
4. $[x = 0]$.
5. $(x - 1)[x > 0]$.
6. $(x - y)[x \geq y]$.
7. $|x - y|$.

Proof.

1. Suppose $f\colon A \to \mathbb{N}$, where $A \subseteq \mathbb{N}^k$, is $c \in \mathbb{N}$ everywhere. If $c = 0$, then f is simplistic, and is primitive as such. Suppose $c > 0$. By induction, the function $g\colon A \to \mathbb{N}^k$ that maps $x \mapsto c - 1$ is primitive. We then have

$$f(x) = s(g(x))$$

for any $x \in A$, so f is primitive by the composition rule.

2. For sums this has been shown in the preceding example. Let $f(x, y) = xy$.

$$f(x, 0) = 0,$$
$$f(x, y + 1) = x(y + 1) = xy + x = f(x, y) + x.$$

Since sums are primitive, f is primitive by the recursion rule.

3. Let $f(x, y) = x^y$.

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

Since products are primitive, f is primitive by the recursion rule.

4. Let $f(x) = [x \neq 0]$. Let $h\colon A \to \mathbb{N}$ be the constant 1. Then

$$f(0) = 0,$$
$$f(y+1) = 1 = h(y).$$

Recursion.

5. Let $f(x) = [x = 0]$. Then

$$f(x) = 1 - [x \neq 0].$$

The function $g(x) = 1 - x$, defined on $\{0, 1\}$, is primitive by a trivial application of the recursion rule. Hence f is, by the composition rule.

6. Let $f(x) = (x-1)[x > 0]$. We denote $f(x) = x \dot{-} 1$.

$$f(0) = 0,$$
$$f(x+1) = x.$$

f is primitive by the recursion rule (the identity function is the projection I_1^1).

7. Let $f(x, y) = (x-y)[x \geq y] = x \dot{-} y$. Observe that

$$x \dot{-} (y+1) = (x \dot{-} y) \dot{-} 1,$$

so

$$f(x, 1) = (x-1)[x \geq 1] = (x-1)[x > 0],$$
$$f(x, y+1) = \left(f(x, y) - 1\right)\left[f(x, y) > 0\right].$$

8. Let $f(x, y) = \left|x - y\right|$. Then

$$\begin{aligned} f(x, y) &= \max(0, x-y) - \min(0, x-y) \\ &= \max(0, x-y) + \max(0, y-x) \\ &= \left(x \dot{-} y\right) + \left(y \dot{-} x\right). \end{aligned}$$

■

Minimisation and partial recursive functions

Definition (minimisation operator μ)**.** If g is a function of arity $n+1$, we may construct f of arity n as

$$f(x_1, \ldots, x_n) = \min\big\{y \mid g(x_1, \ldots, x_n, y) = 0\big\}.$$

Example.

$$x \dot{-} y = \min\Big\{z \mid \big|(x-y) - z\big| = 0\Big\}.$$

Bounded minimisation

Notation. $\overline{x} = (x_1, \ldots, x_n)$.

Definition (bounded minimisation operator $\mu_{\le}$)**.** If g and h are functions of arity $n+1$ and n, respectively, we may construct partial f as

$$f(\overline{x}) = \min\big\{y \mid g(\overline{x}, y) = 0,\ y \le h(\overline{x})\big\}.$$

Lemma. If $f \in \mathrm{PR}$, binary operation $\odot \in \mathrm{PR}_{n+1}$ is associative, and

$$g(\overline{x}, y) = \bigodot_{i=0}^{y} f(\overline{x}, i),$$

then $g \in \mathrm{PR}$.

Proof.

$$g(\overline{x}, 0) = f(\overline{x}, 0),$$
$$g(\overline{x}, y+1) = g(\overline{x}, y) \odot f(\overline{x}, y).$$

■

Lemma. If g and h are total and primitive recursive, and f is as in the previous definition, then f is primitive recursive.

Proof.

$$f(\overline{x}) = \sum_{i=0}^{h(\overline{x})} \prod_{j=0}^{i} \left[g(\overline{x}, j) \neq 0 \right].$$

■

Primitive recursive predicates

Definition. The predicate T is called *primitive recursive*, iff its characteristic function $x \mapsto [T(x)]$ is primitive recursive.

Lemma. If P and Q are primitive recursive predicates, then $\neg P$, $P \vee Q$, $P \wedge Q$, $P \Rightarrow Q$ are primitive recursive.

Proof. The last statement is superfluous, but we write the formula, nevertheless:

$$\begin{aligned} [\neg P] &= 1 - [P], \\ [P \wedge Q] &= [P][Q], \\ [P \vee Q] &= [P] + [Q] - [P][Q], \\ [P \Rightarrow Q] &= [\neg P \vee Q] \\ &= 1 - [P][\neg Q]. \end{aligned}$$

■

Lemma. $=, \leq, \geq, <, >$ are primitive recursive predicates.

Proof.

1. $[x = y] = \left[|x - y| = 0 \right]$.

2. Let $f(x, y) = [x \leq y]$. Then

$$\begin{aligned} f(x, 0) &= 0, \\ f(x, y+1) &= [x \leq y+1] \\ &= [x \leq y] + [x = y+1] \\ &= f(x, y) + [x = y+1]. \end{aligned}$$

   Now recall the point 1.

3. Composing with projections, we swap arguments of $\leq$ to get $\geq$.

4. $[x < y] = [x \leq y] \cdot [x \neq y]$.

5. $[x > y] \in \mathrm{PR}$ by the same token as with $\geq$.

■

Lemma. Let $R \subseteq \mathbb{N}^{n+1}$ be a primitive recursive predicate. Then the predicates

$$\begin{gathered} \exists\, i \leq y \colon R\left(\overline{x}, i\right), \\ \forall\, i \leq y \colon R\left(\overline{x}, i\right), \\ \exists\, i < y \colon R\left(\overline{x}, i\right), \\ \forall\, i < y \colon R\left(\overline{x}, i\right) \end{gathered}$$

are primitive recursive.

Proof. For the first one, observe that

$$\left[\exists\, i \leq y \colon R\left(\overline{x}, i\right)\right] = \bigvee_{i=0}^{y} \left[R\left(\overline{x}, i\right)\right].$$

$\vee$ is an associative operation.

Likewise,

$$\left[\forall\, i \leq y \colon R\left(\overline{x}, i\right)\right] = \bigwedge_{i=0}^{y} \left[R\left(\overline{x}, i\right)\right].$$

The last two predicates are gotten by composing the first two ones with $y \dot{-} 1$. ■

Applications of minimisation

Lemma. The functions

1. $\left\lfloor \frac{x}{y} \right\rfloor$,
2. $[x \mid y]$,
3. $[x \in \mathbb{P}]$,
4. $p_x = \big(\text{the prime } x \text{ in order}\big)$

are primitive recursive.

Proof.

1. $\lfloor x/y \rfloor = \min\{q \mid x < (q+1)y,\ q \le xy\}$ (we need the second condition for the minimisation to be bounded). Since multiplication and comparisons are primitive, the predicate is primitive.
2. $[x \mid y] = \Big[x = y \cdot \lfloor x/y \rfloor\Big]$.
3. Let $f(x)$ be the minimal divisor of x that differs from 1. Then $[x \in \mathbb{P}] = \big[f(x) = x\big]$, and

$$f(x) = \min\{d \mid d \mid x,\ d \neq 1,\ d \le x\}.$$

4. The equations

$$p_0 = 2,$$
$$p_{x+1} = \sum_{i=0}^{p_x!+1} \prod_{j=0}^{i} \big[j \notin \mathbb{P} \vee j \le p_x\big]$$

define $p_\square$, since there is at least one prime in the sum, $p_x! + 1 \in \mathbb{P}$.

■

Lemma. The function

$$x \mapsto \text{undefined}$$

is primitive.

Mutual and complete recursion

Lemma. The function $\binom{x}{2}$ is primitive.

Proof. Indeed,

$$\binom{0}{2} = 0,$$
$$\binom{x+1}{2} = \binom{x}{2} + x.$$

■

Definition. Call $f\colon \mathbb{N}^n \to \mathbb{N}$ a *Cantor enumeration*, iff it is bijective, primitive recursive, and has all coordinate functions of the inverse primitive recursive.

Lemma. Define the $f\colon \mathbb{N}^2 \to \mathbb{N}$ as

$$f(x, y) = \binom{x+y+1}{2} + y.$$

Then f is a Cantor enumeration.

An irrelevant note: $\binom{x+y+1}{2}$ is the number of cells before the diagonal number $x + y$ from the origin. y is the height of the cell (x, y) on this diagonal.

Proof. $\binom{x+y+1}{2}$ is the greatest triangular number not surpassing $f(x, y)$: if the next one, $\binom{x+y+2}{2}$, is $\leq f(x, y)$ (they are monotonous, since there is an injection of pairs), then

$$\sum_{i=0}^{x+y+1} i \leq y + \sum_{i=0}^{x+y} i \iff x + y + 1 \leq y \iff x + 1 \leq 0,$$

which is hardly true for natural x. Hence $x + y$ is uniquely determined, as is y. This we use to write down the inverses g_x, g_y. Put

$$g_s(z) = \min\left\{ t \,\middle|\, \binom{t+2}{2} > z,\ t \le z \right\},$$
$$g_y(z) = z - \binom{g_s(z)}{2},$$
$$g_x(z) = g_s(z) - g_y(z).$$

$\binom{z+2}{2} > z$, since each of the z initial elements gives rise to a pair with the element number $z+1$, and there is a pair which consists of the last two elements. Therefore, g_s is defined everywhere. ■

Theorem. For each $n \in \mathbb{N}_{\ge 1}$ there exists a Cantor enumeration of $\mathbb{N}^n$.

(For $n = 0$, $\mathbb{N}^n$ is finite.)

Proof. By induction over n. In case $n = 1$, we have $\mathrm{id}_{\mathbb{N}}$. Let f and g be Cantor enumerations of $\mathbb{N}^n$ and $\mathbb{N}^2$. Define an enumeration h of $\mathbb{N}^{n+1}$ as

$$h(x_1, \ldots, x_{n+1}) = g\big(f(x_1, \ldots, x_n), x_{n+1}\big).$$

Since f_i^{-1} are functional and primitive by induction, we see that

$$x_i = f_i^{-1}\Big(g_1^{-1}(h)\Big) \text{ for all } i \in \{1, \ldots, n\},$$
$$x_{n+1} = g_2^{-1}(h).$$

■

Definition. Denote

$$\mathrm{ex}(i, x) = \max\left\{ k \,\middle|\, p_i^k \mid x \right\}.$$

Lemma. $\mathrm{ex} \in \mathrm{PR}_2$.

Proof. Since

$$\mathrm{ex}(i, x) = \min\left\{ k \,\middle|\, p_i^{k+1} \nmid x,\ k \le x \right\}.$$

$p_i^{x+1} \nmid x$, since $b^x > x$ for $b \ge 2$. ■

Theorem (complete recursion). Let $s \in \mathbb{N}_{\geq 1}$, $g \in \mathrm{PR}_n$, $h \in \mathrm{PR}_{n+2}$, $t_1, \ldots, t_s \in \mathrm{PR}_1$, $t_i(y) \leq y$ for all $i \in \{1, \ldots, s\}$. Define f as

$$\begin{aligned} f(\overline{x}, 0) &= g(\overline{x}), \\ f(\overline{x}, y+1) &= h\Big(\overline{x}, y, f(\overline{x}, t_1(y)), \ldots, f(\overline{x}, t_s(y))\Big). \end{aligned}$$

Then $f \in \mathrm{PR}_{n+1}$.

Proof. To simplify notation, we put $n = 0$ (the proof would be the same anyway). Using primitive recursion, define

$$\begin{aligned} q(0) &= 2^g, \\ q(x+1) &= q(x) \cdot p_{x+1}^{h\left(x, \mathrm{ex}(t_1(x), q(x)), \ldots, \mathrm{ex}(t_s(x), q(x))\right)}. \end{aligned}$$

Obviously,

$$f(x) = \mathrm{ex}(x, q(x))$$

for all $x \in \mathbb{N}$ — a primitive function. ∎

Theorem (mutual recursion). For $i \in \{1, \ldots, k\}$ and some $g_1, \ldots, g_k, h_1, \ldots, h_k \colon \mathbb{N}^n \to \mathbb{N}$, define

$$\begin{aligned} f_i(\overline{x}, 0) &= g_i(\overline{x}), \\ f_i(\overline{x}, y+1) &= h_i\Big(\overline{x}, y, f_1(\overline{x}, y), \ldots, f_k(\overline{x}, y)\Big). \end{aligned}$$

Suppose g_i, h_i for $i \in [1, s]$ are primitive recursive. Then f is primitive recursive.

Proof. Let $c \colon \mathbb{N}^n \to \mathbb{N}$ be a Cantor enumeration, and, for every $i \in \{1, \ldots, n\}$, $p_i \colon \mathbb{N} \to \mathbb{N}$ its ith inverse. Define

$$f(\overline{x}, y) = c\Big(f_1(\overline{x}, y), \ldots, f_n(\overline{x}, y)\Big).$$

We assert $f \in \mathrm{PR}_{n+1}$: if this is settled, $f_i = p_i \circ f$ are primitive as well. First, define

$$\begin{aligned} \widehat{h}_i(\overline{x}, y, z) &:= h_i(\overline{x}, y, p_1(z), \ldots, p_n(z)) \text{ for any } i \in \{1, \ldots, n\}, \\ h(\overline{x}, y, z) &:= c\Big(\widehat{h}_1(\overline{x}, y, z), \ldots, \widehat{h}_n(\overline{x}, y, z)\Big). \end{aligned}$$

This h is a primitive function. And now we have made our way to applying the recursion rule:

$$\begin{aligned} f(\overline{x}, 0) &= c\Big(g_1(\overline{x}), \ldots, g_n(\overline{x})\Big), \\ f(\overline{x}, y+1) &= h\Big(\overline{x}, y, f(\overline{x}, y)\Big). \end{aligned}$$

■

Theorem. Let $R_0, \ldots, R_k$ be n-ary relations such that

$$R_0 \sqcup \cdots \sqcup R_k = \mathbb{N}^n.$$

For some $f_1, \ldots, f_k \colon \mathbb{N}^n \to \mathbb{N}$, define

$$f(\overline{x}) = \begin{cases} f_0(\overline{x}), & R_0(\overline{x}), \\ \vdots \\ f_k(\overline{x}), & R_k(\overline{x}). \end{cases}$$

Suppose f_i and R_i are primitive recursive. Then f is primitive recursive.

Proof. Indeed,

$$f(\overline{x}) = \sum_{i=0}^{k} f_i(\overline{x}) \left[R_k(\overline{x})\right].$$

■

Computable functions

Definition. Let $D \subseteq \mathbb{N}^m$. A function $f \colon D \to \mathbb{N}^n$ is *computable*, iff there exists a TM that, starting with any $x \in D$ written on its input tape, stops with only $f(x)$ written on the output tape. We denote by R_m the set of all computable partial functions $\subseteq \mathbb{N}^m \to \mathbb{N}$, and by $\mathrm{R}^*_m \subseteq \mathrm{R}_m$ the set of computable functions $\mathbb{N}^m \to \mathbb{N}$.

Definition. A set $X \subseteq \mathbb{N}^k$ is *decidable*, iff its characteristic function is computable.

Lemma. Let $f\colon \mathbb{N} \to \mathbb{N}$ be a constant almost everywhere function. Then f is computable.

Proof. Indeed, it is a primitive recursive function: if $f|_{[t,+\infty)}$ is constant, then

$$f(x) = f(x)[x \geq t] + \sum_{i=0}^{t-1} f(i)[x = i].$$

■

Example. Let $S \subseteq \mathbb{N}$ be the set of such n that the decimal expansion of e contains n consecutive nines. Then S is decidable, since its characteristic function is nondecreasing.

Lemma. An infinite set $A \subseteq \mathbb{N}$ is decidable iff there exists a computable increasing function $f\colon \mathbb{N} \to \mathbb{N}$ such that $A = \operatorname{im} f$.

Proof. Suppose A is decidable. Define f as

$$f(x) = \begin{cases} \min\{a \mid a \in A,\ a > f(x-1)\}, & x > 0, \\ \min\{a \mid a \in A\}, & x = 0. \end{cases}$$

By what we know about minimisation, this function is indeed computable. By its definition, it is increasing. Its image is the complete A: otherwise take the smallest natural $n \in A \setminus \operatorname{im} f$; all the lesser elements of A are in the image; then there exists x such that $f(x-1)$ is the largest number in the image; but then $f(x) = n$. Finally, f is defined everywhere, since otherwise A would be finite.

Conversely, suppose that there exists such a function. To compute $[x \in A]$ for any $x \in \mathbb{N}$, check all values of f until we reach one that is at least this x. The definition of f allows us to do that for all x. ■

Equivalence of Kleene and Turing computability

Theorem. f is computable iff it is partial recursive.

Proof of $\supseteq$. Suppose the function f is partial recursive. We agree to represent a tuple of arguments $\overline{x}$ to f as

$$0\prod_{i=1}^{n} 1^{x_i}0.$$

The proof is as follows:

1. *There is a machine for each of the simplistic functions.* Easy to see.

2. *There is a machine for composition of functions.* In terms of the composition operator, copy the input n times; run TMs for the functions $g_1, \dots, g_n$; run the TM for h on the result.

3. *There is a machine for functions which are constructed by primitive recursion.*

$$\begin{aligned}
M_1 &\colon (\overline{x}, y) \mapsto \Big(\overline{x}, g(\overline{x})\Big), \\
M_2 &\colon (y, \overline{x}, u, z) \mapsto \Big(y, \overline{x}, u+1, h(\overline{x}, u, z)\Big), \\
M_3 &\colon (y, \overline{x}, u, z) \mapsto (z), \\
M_4 &\colon (y, \overline{x}, u, z) \mapsto \Big([u \neq y]\Big).
\end{aligned}$$

   Now the wanted machine can be built as

$$M_1;\text{ while } M_4 \text{ do } M_2; M_3.$$

4. *There is machine for functions constructed by bounded minimisation.* Let

$$\begin{aligned}
N_1 &\colon (\overline{x}) \mapsto (\overline{x}, 0), \\
N_2 &\colon w \mapsto w\#w, \\
N_3 &\colon (\overline{x}, y)\#(\overline{x}, y) \mapsto (\overline{x}, y)\#\Big(g(\overline{x}, y)\Big), \\
N_4 &\colon w\#v \mapsto [v \neq \epsilon], \\
N_5 &\colon (\overline{x}, y)\#w \mapsto (y), \\
N_6 &\colon (\overline{x}, y)\#w \mapsto (\overline{x}, y+1).
\end{aligned}$$

   The sought for machine is

$$N_1, N_2, N_3;\text{ while } N_4 \text{ do } N_6, N_2, N_3; N_5.$$

■

Proof of $\subseteq$. Suppose we have m symbols in the alphabet $\Gamma = \{a_0, \ldots, a_{m-1}\}$. We code configurations as

$$\alpha q a \beta \mapsto \left(\widehat{\alpha}, q, \widehat{a}, \widetilde{\beta}\right),$$

where $\widehat{\square}$ is the number in base m which is written as $\square$; $\widetilde{\square} = \widehat{\square^R}$.

By a pair $(q, a) \in Q \times \Gamma$ we can determine the action of the machine, and this will be a PR function (since it takes meaningful values on only a finite set).

We can transform a configuration by a PR function. For example, if the head moved right, the number $\widehat{\alpha}$ becomes $m \cdot \widehat{\alpha} + \widehat{a}$. The new symbol $\widehat{a}$ is found, in this case, by computing $\widehat{\beta} \,\%\, m$, and the new string $\widehat{\beta}$ as $\left\lfloor \widehat{\beta}/m \right\rfloor$.

We can encode the work of the complete machine using mutual recursion. Define the functions $K, K_\alpha, K_\beta, K_a, K_q$ that compute the elements of the next configuration, based on the previous one. The last parameter of each is some t, so we compute their values on $t+1$, referring to the ones on t.

We can now find the first moment t_f, on which a final state is reached, by using minimisation on K_q. Afterwards we compute $K_a(t_f)$ and $K_b(t_f)$ to find the computation result (wlog, the machine stops with $\alpha = \epsilon$). ■

Corollary. Any partial recursive function can be computed using at most one minimisation.

Corollary. A function, which is computable on a Turing machine in time $O(f)$ where f is primitive recursive, is primitive recursive.

The Ackermann function

In this section, all powers are functional powers.

Definition. Define

$$\alpha_0(x) = x + 1,$$
$$\alpha_i(x) = \alpha_{i-1}^{n+2}(x).$$

The *Ackermann function* $\beta : \mathbb{N} \to \mathbb{N}$ is then defined as

$$\beta(x) = \alpha_x(x).$$

We assert that the function β grows faster than any primitive recursive functions. Yet it is computable (so partial recursive).

Lemma. $\alpha_i(x) > x$ for all $i, x \in \mathbb{N}$.

Proof. For $i = 0$, $x + 1 > x$. For $i > 0$,

$$\begin{aligned}\alpha_i(x) &= \alpha_{i-1}\left(\alpha_{i-1}^{x+1}(x)\right) \\ &> \alpha_{i-1}^{x+1}(x) \\ &\vdots \\ &> x.\end{aligned}$$

∎

Lemma. If $x > y$, then $\alpha_i(x) > \alpha_i(y)$.

Proof. By induction on i, then by induction on x. If $i = 0$,

$$x > y \implies x + 1 > y + 1.$$

If $i > 0$, then

$$\begin{aligned}\alpha_i(y) &= \alpha_{i-1}\left(\alpha_{i-1}^{n+1}(y)\right) \\ &> \alpha_{i-1}\left(\alpha_{i-1}^{n+1}(x)\right) \\ &= \alpha_i(x).\end{aligned}$$

∎

Lemma. For every $x \in \mathbb{N}$, if $i > j$, then $\alpha_i(x) > \alpha_j(x)$.

Proof. If $i = j + 1$, then

$$\begin{aligned} \alpha_{j+1}(x) &= \alpha_j^{x+1}\big(\alpha_j(x)\big) \\ &> \alpha_j(x), \end{aligned}$$

since $\alpha_j(\square) > \square$. ■

Lemma. $\alpha_i(x) > \alpha_{i-1}\big(\alpha_{i-1}(x)\big)$.

Proof.

$$\begin{aligned} \alpha_i(y) &= \alpha_{i-1}^{n+1}\left(\alpha_{i-1}(y)\right) \\ &> \alpha_{i-1}\big(\alpha_{i-1}(x)\big). \end{aligned}$$

■

Lemma. Let $f \in \mathrm{PR}_n$. Then exists k such that, for all $x_1, \ldots, x_n \in \mathbb{N}$,

$$f(x_1, \ldots, x_n) \le \alpha_k\big(\max(x_1, \ldots, x_n)\big).$$

Proof. By induction on the structure of primitive functions.

Consider the simplistic functions.

1. If $f(x) = 0$, then $k = 0$ goes, since $x + 1 > 0$.
2. If $f(x) = x + 1$, then $k = 0$ goes, since $\alpha_1(x) \ge \alpha_0(x) = f(x)$.
3. If $f(\overline{x}) = x_i$, then $k = 0$ goes.

Consider the composition operator. Suppose

$$f(\overline{x}) = h\big(g_1(\overline{x}), \ldots, g_m(\overline{x})\big).$$

By induction there exist k and l such that

$$\begin{aligned}
h\big(g_1(\overline{x}), \ldots, g_m(\overline{x})\big) &\geq \alpha_k\big(g_i(\overline{x})\big) \\
&\geq \alpha_k\Big(\alpha_l\big(g_i(\max \overline{x})\big)\Big) \\
&> \alpha_k\Big(\alpha_l\big(g_i(\max \overline{x})\big)\Big) \\
&> \alpha_k(\max \overline{x}).
\end{aligned}$$

Consider the primitive recursion operator. Suppose

$$\begin{aligned}
f(x_1, \ldots, x_n, 0) &= g(x_1, \ldots, x_n) \\
f(x_1, \ldots, x_n, y+1) &= h\big(x_1, \ldots, x_n, y, f(x_1, \ldots, x_n, y)\big).
\end{aligned}$$

There exists k such that

$$\begin{aligned}
f(x_1, \ldots, x_n, 0) &= g(x_1, \ldots, x_n) \\
&\leq \alpha_k\big(\max\{x_1, \ldots, x_n\}\big),
\end{aligned}$$

$$\begin{aligned}
f(x_1, \ldots, x_n, y+1) &= h\big(x_1, \ldots, x_n, y, f(x_1, \ldots, x_n, y)\big) \\
&\leq \alpha_k\Big(\max\big\{x_1, \ldots, x_n, y, f(x_1, \ldots, x_n, y)\big\}\Big) \\
&\leq \alpha_k\Big(\max\big\{x_1, \ldots, x_n, y+1\big\}\Big)
\end{aligned}$$

■

Theorem. Let $\beta(x) = \alpha_x(x)$, $f \in \mathrm{PR}_n$. There exists $k \in \mathbb{N}$ such that $\beta(x) > f(x)$ for all $x > k$.

Proof. By the previous lemma, there is k such that

$$f(x) \leq \alpha_k(x).$$

If $x > k$, this inequality can be continued to yield

$$f(x) < \alpha_x(x).$$

■

Some partial recursive functions

Lemma. The function

$$f(x,y) = \begin{cases} x/y, & [y \mid x], \\ \text{undefined}, & \text{otherwise} \end{cases}$$

is partial recursive.

Proof. Indeed,

$$f(x,y) = \min\{q \mid qy = x\}.$$

■

Enumerability

Definition. A set $S \subseteq \mathbb{N}$ is *enumerable*, iff there exists a TM that outputs all the elements of S and only them, separated by commas.

Example. $\varnothing$ and $\mathbb{N}$ are enumerable. In general, all decidable sets are enumerable.

It is not long before until we give an example of an enumerable, non-decidable set.

Definition. Let $S \subseteq \mathbb{N}$. Its *semicharacteristic* function is the partial function

$$\urcorner_S(x) := \begin{cases} 1, & x \in S \\ \text{undefined}, & \text{otherwise.} \end{cases}$$

Lemma. Let $S \subseteq \mathbb{N}$ be nonempty. The following are equivalent:

1. S is enumerable.
2. Its semicharacteristic function is computable.
3. There exists a computable $f\colon S \to \mathbb{N}$.
4. There exists an initial segment $R \subseteq \mathbb{N}$ and a computable bijective $g\colon R \to S$.
5. There exists a computable surjective $h\colon \mathbb{N} \to S$.

$1 \Rightarrow 2$. Run the TM until it outputs the element. We are comfortable with the prospect of it never doing that. ■

$2 \Rightarrow 3$. Obvious. ■

$3 \Rightarrow 4$. Via minimisation: let $g(0)$ be the smallest member of S, and let $g(x)$ for $x > 0$ be the smallest member of S that is greater than $g(x-1)$. This function is not defined everywhere in case of a finite S. ■

$4 \Rightarrow 5$. To deal with the case of a finite S, we put $h(x) := g(x)$ where g is defined, and $h(x) = \max S$, if $x \geq |R|$. ■

$5 \Rightarrow 1$. Build a TM that outputs $f(0), f(1), \ldots$ ■

Lemma (Post's criterion). Let $S \subseteq \mathbb{N}$. S is decidable iff both S and $\overline{S}$ are enumerable.

$\Rightarrow$. Iterate over naturals and check inclusion for each. Output accordingly. ■

$\Leftarrow$. To compute the characteristic function on $x \in \mathbb{N}$, we run machines for S and $\overline{S}$ in parallel. The first to output x corresponds to the correct answer. ■

Projections

Definition. Let $S \subseteq \mathbb{N}^n$. We call the set

$$\operatorname*{Proj}_i S = \left\{x_i \mid \exists\, x_-, x_+ \colon (x_-, x_i, x_+) \in S\right\}$$

a *projection* of S onto the coordinate i.

Lemma (on projections)**.** Let $P \subseteq \mathbb{N}$. The following are equivalent:

1. P is enumerable.
2. There exists a decidable $Q \subseteq \mathbb{N}^2$ such that P is a projection of Q.

$1 \Rightarrow 2$. Define Q to be the set of pairs (n, t) such that the number n appears in the output of the enumerating machine of P within t steps. ■

$2 \Rightarrow 1$. Using a Cantor enumeration, iterate over all members of Q, outputting their projections (onto the known coordinate). ■

Definition. A set $S \subseteq \mathbb{N}^n$ is *enumerable*, iff its image under a Cantor enumeration of $\mathbb{N}^n$ is enumerable.

Lemma (on graphs)**.** Let $f \colon \mathbb{N} \to \mathbb{N}$. The following are equivalent:

1. f is computable.
2. Its graph is enumerable.

$1 \Rightarrow 2$. Since f is defined everywhere, we may iterate over the naturals and output for each $x \in \mathbb{N}$ the pair $(x, f(x))$ (in the guise of its image under a Cantor enumeration). ■

$2 \Rightarrow 1$. To compute $f(x)$, run the Turing machine for the graph until arriving at $(x, f(x))$. This will happen. ∎

Universal functions

Definition. Let $D \subseteq \mathbb{N}^{m+1}$, and let $U : D \to \mathbb{N}^n$ be a computable function. The function $U_k : \mathrm{Proj}_{1,\dots,m} D \to \mathbb{N}^n$, defined, for $k \in \mathrm{Proj}_0\, d$ and $x \in \mathbb{N}^m$ as

$$U_k(x) := U(k, x),$$

we will call a *section* of U. U itself is said to be *universal* for the class of functions

$$C = \left\{ U_k \mid k \in \operatorname*{Proj}_0 D \right\}.$$

Lemma. There exists a universal function for the class R_m.

Proof. Let $U_k(x)$ be the output of the TM number k on x. ∎

Lemma. There does not exist an everywhere defined universal function $U : \mathbb{N}^2 \to \mathbb{N}$ for the class R_1^*.

Proof. By diagonal argument. Consider the function $n \mapsto U(n, n) + 1$. It is computable, but differs from any section U_k at k:

$$U_k(k) + 1 \neq U_k(k).$$

A contradiction. ∎

Remark. This proof fails for R_1, since $U(k, k)$ may not exist.

Lemma. There exists a computable $f\colon \mathbb{N} \to \mathbb{N}$ such that there does not exist an $F\colon \mathbb{N} \to \mathbb{N}$ with $F|_{\operatorname{dom} f} = f$.

Proof. Take $f\colon n \mapsto U(n,n)+1$, where U is an (existent) universal function for R_1. Suppose F exists.

- If $n \in \operatorname{dom} f$, then $F(n) \neq U(n,n)$.
- If $n \notin \operatorname{dom} f$, then $n \notin \operatorname{dom} U$ and $n \in \operatorname{dom} F$.

Therefore, $F \neq U_n$ for any $n \in \mathbb{N}$. But $F \in \mathrm{R}_1^* \subseteq \mathrm{R}_1$, and U is universal for R_1. A contradiction. ■

An enumerable non-decidable set

Theorem. There exists an enumerable non-decidable set.

Proof. Let $f\colon n \mapsto U(n,n)+1$ be as in the previous lemma. $S := \operatorname{dom} f$ is enumerable, as it is the domain of a computable function. We assert that S is undecidable. Suppose otherwise. Put

$$F(x) = \left[x \in S\right] f(x).$$

Since the characteristic function $\left[x \in S\right]$ is computable, F is computable and defined everywhere. This contradicts the previous lemma. ■

Some remarks on this proof

- In fact, the set
$$S = \left\{n \mid U(n,n) \text{ is defined}\right\}$$
is a guise of the classic diagonal argument on machines that do not accept themselves.
- $\overline{S}$ is not enumerable. If both S and $\overline{S}$ were enumerable, they would be decidable (obvious, but we have met that on page 22).

- The domain of this $U\colon\subseteq \mathbb{N}^2 \to \mathbb{N}$ is an enumerable, undecidable set itself. It is enumerable, as it is a domain of a computable function, but the diagonal function $n \mapsto U(n,n)$ serves as a counterexample to decidability.

Lemma. There exists a computable $f\colon\subseteq \mathbb{N} \to 2$ that does not a have an everywhere defined computable continuation $F\colon \mathbb{N} \to \mathbb{N}$ (so that $F|_{\operatorname{dom} f} = f$).

Proof. Put

$$f(x) = \begin{cases} [U(x,x) = 0], & \text{if } U(x,x) \text{ is defined,} \\ \text{undefined} & \text{otherwise.} \end{cases}$$

F differs from any section of U in the diagonal. ∎

Unseparable enumerable sets

Lemma. There exist enumerable disjoint X and Y such that, if Z is decidable and $Z \supseteq X$, then

$$Y \cap Z \neq 0.$$

Thus, X and Y cannot be 'separated' by decidable sets.

Proof. Let f be as in the previous lemma, and put $X = f^{-1}(1)$, $Y = f^{-1}(0)$. Suppose there exists a decidable Z such that $Z \supseteq X$ and $Z \cap Y = \varnothing$. Now let

$$F(x) = \begin{cases} [x \in Z], & x \in \operatorname{dom} f, \\ 0 & x \notin \operatorname{dom} f. \end{cases}$$

∎