Chapter 3

Recursive Functions

The whole chapter is highly inspired by René Cori and Daniel Lascar book's book: *"Mathematical Logic, Part 2, Recursion Theory, Gödel Theorems, Set Theory, Model Theory"* [9].

Recursive functions are functions from $\mathbb{N}^p$ to $\mathbb{N}$. We will show that they have a strong relation with the Turing computable ones.
We define the set of recursive functions by induction. For this purpose, for any integer p, we denote by $\mathbb{N}^{(\mathbb{N}^p)}$ the set of all mappings of the form $\mathbb{N}^p \longrightarrow \mathbb{N}$. Notice that $\mathbb{N}^p$ is a notation for the set of all mappings $\{i \in \mathbb{N} \mid i < p\} \longrightarrow \mathbb{N}$. When $p = 0$, the set $\{i \in \mathbb{N} \mid i < p\}$ becomes $\{i \in \mathbb{N} \mid i < 0\} = \varnothing$. Thus the set $\mathbb{N}^0$ only contains one element: the empty function whose graph is $\varnothing$. Therefore the set of all mappings of the form $\mathbb{N}^0 \longrightarrow \mathbb{N}$ contains all mappings that assign one integer to the empty function:

$$\mathbb{N}^0 \longrightarrow \mathbb{N} = \left\{ \begin{array}{lccc} f: & \{\varnothing\} & \longrightarrow & \mathbb{N} \\ & \varnothing & \longrightarrow & n \end{array} \middle| \, n \in \mathbb{N} \right\}.$$

So, as may be expected, mappings in $\mathbb{N}^{(\mathbb{N}^0)}$ are identified with elements of $\mathbb{N}$.

3.1 Primitive Recursive Functions

Definition 1.1

projection: If i is any integer such that $1 \leqslant i \leqslant p$ holds, the i^{th} *projection* π_i^p is the function of $\mathbb{N}^{(\mathbb{N}^p)}$ defined by

$$\pi_i^p(x_1, \ldots, x_p) = x_i.$$

successor: $S \in \mathbb{N}^{\mathbb{N}}$ is the *successor* function[a].

composition: Given $f_1, \dots, f_n \in \mathbb{N}^{(\mathbb{N}^p)}$ and $g \in \mathbb{N}^{(\mathbb{N}^n)}$, the *composition* $h = g(f_1, \dots, f_n) \in \mathbb{N}^{(\mathbb{N}^p)}$ is defined by

$$h(x_1, \dots, x_p) = g\big(f_1(x_1, \dots, x_p), \dots, f_n(x_1, \dots, x_p)\big)$$

We often make use of the notation $\overrightarrow{x}$ for $(x_1, \dots, x_p)$ so that for instance

$$g\big(f_1(\overrightarrow{x}), \dots, f_n(\overrightarrow{x})\big)$$

stands for

$$g\big(f_1(x_1, \dots, x_p), \dots, f_n(x_1, \dots, x_p)\big).$$

recursion: Given $g \in \mathbb{N}^{(\mathbb{N}^p)}$ and $h \in \mathbb{N}^{(\mathbb{N}^{p+2})}$, there exists a unique $f \in \mathbb{N}^{(\mathbb{N}^{p+1})}$ such that for all $\overrightarrow{x} \in \mathbb{N}^p$ and $y \in \mathbb{N}$ satisfies

(1) $f(\overrightarrow{x}, 0) = g(\overrightarrow{x})$

(2) $f(\overrightarrow{x}, y+1) = h\big(\overrightarrow{x}, y, f(\overrightarrow{x}, y)\big)$

We say f is defined by *recursion* on both g (for the initial step) and h (for the successor steps).

[a] $S(n) = n + 1$.

Definition 1.2

The set of *primitive recursive* ($\mathcal{P}rim.\ \mathcal{R}ec.$) functions is the least that

(1) contains:

(a) All constants $\mathbb{N}^p \longrightarrow \mathbb{N}$ (all $\bar{i} \in \mathbb{N}^{(\mathbb{N}^p)}$ s.t. $\bar{i}(\overrightarrow{x}) = i$ – any $i, p \in \mathbb{N}$).

(b) All projections π_i^p (any $p \in \mathbb{N}$, any $1 \leqslant i \leqslant p$)

(c) The successor function $S \in \mathbb{N}^{\mathbb{N}}$.

(2) and is closed under

(a) composition

(b) recursion

We set up these functions in a hierarchy $(R_n)_{n\in\mathbb{N}}$:

(1) R_0 is the set of all functions in (1)(a), (1)(b) and (1)(c).

(2) R_{n+1} is the closure[a] of R_n under (2)(a) and (2)(b).

Clearly

$$\mathcal{P}rim.\ \mathcal{R}ec. = \bigcup_{n\in\mathbb{N}} R_n.$$

[a] $R_{n+1} = R_n \cup \{h \text{ obtained by composition on the basis of functions in } R_n\} \cup \{h \text{ obtained by induction on the basis of functions in } R_n\}$.

Example 1.1

(1) Addition: $(x, y) \longrightarrow x + y$
We have:

$$\begin{cases} x + 0 = x \\ x + (y + 1) = (x + y) + 1. \end{cases} \tag{3.1}$$

Formally:

$$\begin{cases} add(x, 0) = \pi_1^1(x) \\ add(x, y + 1) = S\Big(\pi_3^3\big(x, y, add(x, y)\big)\Big). \end{cases} \tag{3.2}$$

(2) Multiplication: $(x, y) \longrightarrow x \cdot y$
We have

$$\begin{cases} x \cdot 0 = 0 \\ x \cdot (y + 1) = x \cdot y + x. \end{cases} \tag{3.3}$$

Formally:

$$\begin{cases} mult(x, 0) = \overline{0}(x) \\ mult(x, y + 1) = add\Big(\pi_3^3\big(x, y, mult(x, y)\big), \pi_1^3\big(x, y, mult(x, y)\big)\Big). \end{cases} \tag{3.4}$$

(3) Exponentiation: $x \longrightarrow n^x$
We have

$$\begin{cases} n^0 = 1 \\ n^{x+1} = n^x \cdot n. \end{cases} \tag{3.5}$$

Formally:

$$\begin{cases} exp_n(0) = 1 \\ exp_n(x + 1) = mult\Big(\pi_2^2\big(x, exp_n(x)\big), \overline{n}\Big). \end{cases} \tag{3.6}$$

(4) Factorial: $x \longrightarrow x!$
We have

$$\begin{cases} 0! = 1 \\ (x + 1)! = x! \cdot (x + 1). \end{cases} \tag{3.7}$$

Formally:

$$\begin{cases} fact(0) = 1 \\ fact(x+1) = mult\Big(\pi_2^2\big(x, fact(x)\big), S\Big(\pi_1^2\big(x, fact(x)\big)\Big)\Big). \end{cases} \tag{3.8}$$

Example 1.2

We define $\dot{-} \in \mathbb{N}^{(\mathbb{N}^2)}$ by

$$\begin{cases} x \dot{-} y &= x - y \quad \textit{if } x > y, \\ &= 0 \quad \textit{otherwise.} \end{cases}$$

To show $\dot{-} \in \mathbb{N}^{(\mathbb{N}^2)}$ belongs to $\mathcal{P}rim.\ \mathcal{R}ec.$, we first show $x \longrightarrow x \dot{-} 1$ belongs to $\mathcal{P}rim.\ \mathcal{R}ec.$

$$\begin{cases} 0 \dot{-} 1 = 0 \\ (x+1) \dot{-} 1 = x \end{cases} \tag{3.9}$$

Formally:

$$\begin{cases} 0 \dot{-} 1 = \overline{0}(x) \\ (x+1) \dot{-} 1 = \pi_1^2\big(x, x \dot{-} 1\big) \end{cases} \tag{3.10}$$

$$\begin{cases} x \dot{-} 0 = x \\ x \dot{-} (y+1) = \big(x \dot{-} y\big) \dot{-} 1 \end{cases} \tag{3.11}$$

Formally:

$$\begin{cases} x \dot{-} 0 = \pi_1^1(x) \\ x \dot{-} (y+1) = \Big(\pi_3^3\big(x, y, x \dot{-} y\big)\Big) \dot{-} 1 \end{cases} \tag{3.12}$$

Definition 1.3

A set $A \subseteq \mathbb{N}^p$ is *primitive recursive* ($\mathcal{P}rim.\ \mathcal{R}ec.$) if its characteristic function ($\chi_A \in \mathbb{N}^{(\mathbb{N}^p)}$) is primitive recursive.

Example 1.3

(1) The set $\varnothing$ is $\mathcal{P}rim.\ \mathcal{R}ec.$ since $\chi_\varnothing = \overline{0}$ is $\mathcal{P}rim.\ \mathcal{R}ec.$

(2) The set $\mathbb{N}$ is $\mathcal{P}rim.\ \mathcal{R}ec.$ since $\chi_{\mathbb{N}} = \overline{1}$ is $\mathcal{P}rim.\ \mathcal{R}ec.$

(3) The set $<_{\mathbb{N}} = \{(x, y) \mid x < y\}$ is $\mathcal{P}rim.\ \mathcal{R}ec.$ $\chi_{<_{\mathbb{N}}}(x, y) = 1 \dot{-} (1 \dot{-} (y \dot{-} x))$.

On computable and partial functions

Definition 1.4

(1) (dom_f, f) is a *partial function* $\mathbb{N}^p \longrightarrow \mathbb{N}$ if f is a mapping $dom_f \longrightarrow \mathbb{N}$ where $dom_f \subseteq \mathbb{N}^p$.

(2) (dom_f, f) is a *total function* $\mathbb{N}^p \longrightarrow \mathbb{N}$ if $dom_f = \mathbb{N}^p$ holds.

We say that f is undefined on x – or $f(x)$ is undefined – if $x \notin dom_f$.

Notation 1.1

We write $f \in \mathbb{N}^{(dom \subseteq \mathbb{N}^p)}$ for (dom_f, f) is a *partial function* $\mathbb{N}^p \longrightarrow \mathbb{N}$ whose domain is dom_f.

Notice that given any two partial functions $f, g \in \mathbb{N}^{(dom \subseteq \mathbb{N}^p)}$,

$$f = g \iff \begin{cases} dom_f = dom_g \\ \text{and} \\ \forall x \quad f(x) = g(x). \end{cases}$$

Definition 1.5

A partial function $f \in \mathbb{N}^{(dom \subseteq \mathbb{N}^p)}$ is *"Turing Computable"* (TC) if there exists a TM $\mathcal{M}$ such that on input $\overrightarrow{x} = (x_1, \ldots, x_p)$:

(1) if $f(\overrightarrow{x})$ is not defined, then $\mathcal{M}(\overrightarrow{x}) \uparrow$;

(2) if $(\overrightarrow{x}) \in dom_f$, $\mathcal{M}(\overrightarrow{x}) \downarrow$ with $f(\overrightarrow{x})$ written on its tape.

Proposition 1.1

Given any partial function $f \in \mathbb{N}^{(dom \subseteq \mathbb{N}^p)}$,

$$f \text{ is Turing Computable} \iff \mathcal{G}_f = \left\{ \left(\overrightarrow{x}, f(x) \right) \mid \overrightarrow{x} \in dom_f \right\} \text{ is Turing Recognizable.}$$

	$f \in \mathbb{N}^{(\mathbb{N}^p)}$	$f \in \mathbb{N}^{(dom \subseteq \mathbb{N}^p)}$
$\mathcal{G}_f \subseteq \mathbb{N}^{p+1}$	*recursive* $\|\|$ *decidable*	*rec. enum.* $\|\|$ *Turing rec.*

Figure 3.1: Relations between Turing computable functions and their graphs

Proof of Proposition 1.1:

($\Rightarrow$) From $\mathcal{M}$ that computes f it is immediate to build $\mathcal{N}$ that recognises $\mathcal{G}_f$. On input $(\overrightarrow{x}, y)$ it simulates $\mathcal{M}$ if $\mathcal{M}(\overrightarrow{x}, y) \overset{\text{acc.}}{\downarrow}$ it compares $f(x)$ with y and if $f(x) = y$ it accepts, otherwise it rejects.

($\Leftarrow$) From $\mathcal{N}$ that recognises $\mathcal{G}_f$, we build $\mathcal{M}$ that computes f as follows, on input $\overrightarrow{x}$ repeatedly for $i = 1, 2, 3, \ldots$ it recursively simulates $\mathcal{N}$ on $(\overrightarrow{x}, 0), (\overrightarrow{x}, 1), (\overrightarrow{x}, 2), \ldots, (\overrightarrow{x}, i)$ for i many steps. If $\mathcal{N}$ accepts $(\overrightarrow{x}, n)$, then $\mathcal{M}$ prints out the value n.

□

Corollary 1.1

Given any function $f : \mathbb{N}^p \longrightarrow \mathbb{N}$,

$$f \textit{ is both total and Turing Computable} \implies \mathcal{G}_f \textit{ is recursive (decidable).}$$

Proof of Corollary 1.1:

Left as an immediate exercise.

□

We notice tha following:
All functions in R_0 are total and Turing computable. By induction on n, it is easy to show that all functions in R_n are also total and Turing computable. Therefore

- All $\mathcal{P}rim.\ \mathcal{R}ec.$ functions are total and Turing computable.

- All graphs of $\mathcal{P}rim.\ \mathcal{R}ec.$ functions are *recursive*

Even though the class of all $\mathcal{P}rim.\ \mathcal{R}ec.$ functions is included in the class of total and turing computable functions, the inverse inclusion does not hold[1].

3.2 Variable Substitution

Proposition 2.1: $\mathcal{P}rim.\ \mathcal{R}ec.$ closed under variable substitution

If $f \in \mathbb{N}^{(\mathbb{N}^p)}$, is $\mathcal{P}rim.\ \mathcal{R}ec.$, then given any $\sigma : \{1, \dots, p\} \longrightarrow \{1, \dots, p\}$, the function $g \in \mathbb{N}^{(\mathbb{N}^p)}$ defined by

$$g(x_1, \dots, x_p) = f(x_{\sigma(1)}, \dots, x_{\sigma(p)})$$

is also $\mathcal{P}rim.\ \mathcal{R}ec.$

Proof of Proposition 2.1:

The result is proved for all $g \in R_n$ by induction on n. Left as an exercise.

□

Proposition 2.2

If $A \subseteq \mathbb{N}^n$ is $\mathcal{P}rim.\ \mathcal{R}ec.$ and $f_1, \dots, f_n \in \mathbb{N}^{(\mathbb{N}^p)}$ are $\mathcal{P}rim.\ \mathcal{R}ec.$ then

$$\{\overrightarrow{x} \in \mathbb{N}^p \mid (f_1(\overrightarrow{x}), \dots, f_n(\overrightarrow{x})) \in A\}$$

is $\mathcal{P}rim.\ \mathcal{R}ec.$

Proof of Proposition 2.2:

Set $B = \{\overrightarrow{x} \in \mathbb{N}^p \mid (f_1(\overrightarrow{x}), \dots, f_n(\overrightarrow{x})) \in A\}$. We have

$$\chi_B(\overrightarrow{x}) = \chi_A\Big(f_1(\overrightarrow{x}), \dots, f_n(\overrightarrow{x})\Big).$$

[1] see exercise on the Ackermann function $A \in \mathbb{N}^{(\mathbb{N}^2)}$ defined by

$$A(m,n) = \begin{cases} n+1 & \text{if} \quad m = 0, \\ A(m-1,1) & \text{if} \quad m > 0 \text{ and } n = 0, \\ A\big(m-1, A(m,n-1)\big) & \text{if} \quad m > 0 \text{ and } n > 0. \end{cases}$$

$A(m,n)$ is fast growing ; for instance $2 \cdot 10^{19728} < A(4,2) < 3 \cdot 10^{19728}$.

□

Example 2.1

If $f, g \in \mathbb{N}^{(\mathbb{N}^p)}$ are $\mathcal{P}rim.\ \mathcal{R}ec.$, then the following sets are also $\mathcal{P}rim.\ \mathcal{R}ec.$:

(1) $\{\overrightarrow{x} \mid f(\overrightarrow{x}) > g(\overrightarrow{x})\}$ (2) $\{\overrightarrow{x} \mid f(\overrightarrow{x}) = g(\overrightarrow{x})\}$ (3) $\{\overrightarrow{x} \mid f(\overrightarrow{x}) < g(\overrightarrow{x})\}$.

Proposition 2.3

If $A, B \subseteq \mathbb{N}^p$ are $\mathcal{P}rim.\ \mathcal{R}ec.$ then $A \cup B$, $A \cap B$, $A \setminus B$, $A \Delta B$ and $\mathbb{N}^p \setminus A$ are all $\mathcal{P}rim.\ \mathcal{R}ec.$

Proof of Proposition 2.3:

$$\begin{aligned}
\chi_{A\cup B} &= 1 \dot{-} \left(1 \dot{-} (\chi_A + \chi_B)\right) \\
\chi_{A\cap B} &= \chi_A \cdot \chi_B \\
\chi_{A\setminus B} &= \chi_A \cdot \left(1 \dot{-} \chi_B\right) \\
\chi_{A\Delta B} &= \left(1 \dot{-} \chi_A \cdot \chi_B\right) \cdot \left(1 \dot{-} \left(\left(1 \dot{-} \chi_A\right) \cdot \left(1 \dot{-} \chi_B\right)\right)\right) \\
\chi_{A^c} &= 1 \dot{-} \chi_A.
\end{aligned}$$

□

Proposition 2.4: Case study

If $f_1, \ldots, f_{n+1} \in \mathbb{N}^{(\mathbb{N}^p)}$ and $A_1, \ldots, A_n \in \mathbb{N}^p$ are all $\mathcal{P}rim.\ \mathcal{R}ec.$, then $g \in \mathbb{N}^{(\mathbb{N}^p)}$ defined by:

$$g(\overrightarrow{x}) = \begin{cases}
f_1(\overrightarrow{x}) & \text{if} \quad \overrightarrow{x} \in A_1 \\
f_2(\overrightarrow{x}) & \text{if} \quad \overrightarrow{x} \in A_2 \setminus A_1 \\
f_3(\overrightarrow{x}) & \text{if} \quad \overrightarrow{x} \in A_3 \setminus (A_1 \cup A_2) \\
\vdots & \vdots \qquad\qquad \vdots \\
f_i(\overrightarrow{x}) & \text{if} \quad \overrightarrow{x} \in A_i \setminus (A_1 \cup A_2 \cup \ldots \cup A_{i-1}) \\
\vdots & \vdots \qquad\qquad \vdots \\
f_{n+1}(\overrightarrow{x}) & \text{if} \quad \overrightarrow{x} \notin (A_1 \cup \ldots \cup A_n)
\end{cases}$$

is also $\mathcal{P}rim.\ \mathcal{R}ec.$

Proof of Proposition 2.4:

$g = f_1 \cdot \chi_{A_1} + f_2 \cdot \chi_{(A_2 \setminus A_1)} + \ldots + f_{n+1} \cdot \chi_{\left(A_1 \cup A_2 \cup \ldots \cup A_n\right)^{\complement}}$. □

Corollary 2.1

$\sup(x_1, \ldots, x_n)$ and $\inf(x_1, \ldots, x_n)$ are $\mathcal{P}rim.\ \mathcal{R}ec.$

Proof of Corollary 2.1:

Left as an exercise. □

Proposition 2.5

$f \in \mathbb{N}^{(\mathbb{N}^{p+1})}$ is $\mathcal{P}rim.\ \mathcal{R}ec.$, then $g, h \in \mathbb{N}^{(\mathbb{N}^{p})}$ below are $\mathcal{P}rim.\ \mathcal{R}ec.$:

$$g(x_1, \ldots, x_p, y) = \sum_{t=0}^{t=y} f(x_1, \ldots, x_p, t),$$

$$h(x_1, \ldots, x_p, y) = \prod_{t=0}^{t=y} f(x_1, \ldots, x_p, t).$$

Proof of Proposition 2.5:

Left as an exercise (both are easily defined by induction).

□

3.3 Bounded Minimisation and Bounded Quantification

Proposition 3.1: Bounded minimisation

If $A \subseteq \mathbb{N}^{p+1}$ is $\mathcal{P}rim.\ \mathcal{R}ec.$, then $f \in \mathbb{N}^{(\mathbb{N}^{p+1})}$ defined below is also $\mathcal{P}rim.\ \mathcal{R}ec.$:

$$f(\overrightarrow{x}, z) = \begin{cases} 0 \text{ if } \forall t \leqslant z \quad (\overrightarrow{x}, t) \notin A, \\ \text{the least } t \leqslant z \text{ such that } (\overrightarrow{x}, t) \in A \text{ otherwise.} \end{cases}$$

$f(\overrightarrow{x}, z)$ is denoted by $\mu t \leqslant z\ (\overrightarrow{x}, t) \in A$.

Proof of Proposition 3.1:

f is defined by:

$$\begin{cases} f(\overrightarrow{x}, 0) &= 0 \\ f(\overrightarrow{x}, z+1) &= \begin{cases} f(\overrightarrow{x}, z) & \text{if } \sum_{y=0}^{y=z} \chi_A(\overrightarrow{x}, y) \geqslant 1 \\ z+1 & \text{if } \sum_{y=0}^{y=z} \chi_A(\overrightarrow{x}, y) = 0 \text{ and } (\overrightarrow{x}, z+1) \in A \\ 0 & \text{if } \sum_{y=0}^{y=z+1} \chi_A(\overrightarrow{x}, y) = 0. \end{cases} \end{cases}$$

□

Proposition 3.2: *Prim. Rec.* closed under bounded quantification

The set of all *Prim. Rec.* predicates is closed under bounded quantification: i.e., If $A \subseteq \mathbb{N}^{p+1}$ is *Prim. Rec.*, then

- $\{(\overrightarrow{x}, z) : \exists t \leqslant z\ (\overrightarrow{x}, t) \in A\}$
- $\{(\overrightarrow{x}, z) : \forall t \leqslant z\ (\overrightarrow{x}, t) \in A\}$

are both *Prim. Rec.*

Proof of Proposition 3.2:

Set

- $B = \{(\overrightarrow{x}, z) : \exists t \leqslant z\ (\overrightarrow{x}, t) \in A\}$,
- $C = \{(\overrightarrow{x}, z) : \forall t \leqslant z\ (\overrightarrow{x}, t) \in A\}$.

We have

- $\chi_B(\overrightarrow{x}, z) = 1 \dot{-} (1 \dot{-} \sum_{t=0}^{t=z} \chi_A(\overrightarrow{x}, t))$,
- $\chi_C(\overrightarrow{x}, z) = \prod_{t=0}^{t=z} \chi_A(\overrightarrow{x}, t))$.

□

Example 3.1

- $\{2n : n \in \mathbb{N}\}$ is $\mathcal{P}rim.\ \mathcal{R}ec.$ It is defined by recursion

$$\begin{cases} \chi_{(0)} & = & 1 \\ \chi_{(n+1)} & = & 1 \dot{-} \chi_{(n)} \end{cases}$$

- The mapping $\in \mathbb{N}^{(\mathbb{N}^2)}$ $(x,y) \longrightarrow \left[\dfrac{x}{y}\right]$ defined below is $\mathcal{P}rim.\ \mathcal{R}ec.$:

$$\begin{cases} \left[\dfrac{x}{y}\right] & = & 0 \ \textit{if}\ y = 0 \\ & = & \textit{integer part of}\ \dfrac{x}{y}\ \textit{otherwise.} \end{cases}$$

Formally

$$\begin{cases} \left[\dfrac{x}{y}\right] & = & 0 \ \textit{if}\ y = 0 \\ & = & \mu t \leqslant x \quad y \cdot (t+1) > x \ \textit{otherwise.} \end{cases}$$

- $\left\{(x,y) \mid y \ \textit{divides}\ x\right\} \in \mathcal{P}rim.\ \mathcal{R}ec.$:

$$\chi(x,y) = 1 \dot{-} \left(x \dot{-} \left(y \cdot \left[\frac{x}{y}\right]\right)\right).$$

- $\textit{Prime} = \{x \in \mathbb{N} \mid x\ \textit{is a prime number}\} \in \mathcal{P}rim.\ \mathcal{R}ec.$:

$$x \in \textit{Prime} \iff \begin{cases} x > 1 \\ \\ \textit{and} \\ \\ \forall y \leqslant x \begin{cases} y = 1 \\ \textit{or} \\ y = x \\ \textit{or} \\ y\ \textit{does not divide}\ x. \end{cases} \end{cases}$$

- $\Pi : \mathbb{N} \longrightarrow \mathbb{N}$ *defined by* $\Pi(n) = n + 1^{th}$ *prime number* $\in \mathcal{P}rim.\ \mathcal{R}ec.$.

$$\begin{cases} \Pi(0) & = & 2 \\ \Pi(n+1) & = & \mu z \leqslant (\Pi(n)! + 1) \quad z > \Pi(n)\ \textit{and}\ z \in\ \textit{Prime.} \end{cases}$$

3.4 Coding Sequences of Integers

We define $\mathcal{P}rim.\ \mathcal{R}ec.$ functions that allow to treat finite sequences of integers as integers. Every sequence $\langle x_1, \ldots, x_p \rangle$ will be *"coded"* by a single integer $\alpha_p(x_1, \ldots, x_p)$. And from this single integer $\alpha_p(x_1, \ldots, x_p)$ one will be able to recover the elements of the original sequence by having $\mathcal{P}rim.\ \mathcal{R}ec.$ functions β_p^i that satisfy

$$\beta_p^i\Big(\alpha_p(x_1, \ldots, x_p)\Big) = x_i.$$

Proposition 4.1

For every non-zero $p \in \mathbb{N}$ there exists $\mathcal{P}rim.\ \mathcal{R}ec.$ functions $\beta_p^1, \beta_p^2, \ldots, \beta_p^p \in \mathbb{N}^{\mathbb{N}}$ and $\alpha_p \in \mathbb{N}^{(\mathbb{N}^p)}$ such that

$$\begin{cases} \alpha_p : \mathbb{N}^p \xleftrightarrow{1\text{–}1 \text{ and onto}} \mathbb{N} \\ \\ \text{and} \\ \\ \alpha_p^{-1}(x) = \big(\beta_p^1(x), \ldots, \beta_p^p(x)\big). \end{cases}$$

Proof of Proposition 4.1:

We start by defining $\alpha_1 = \beta_1^1 = id$. Then we move on to

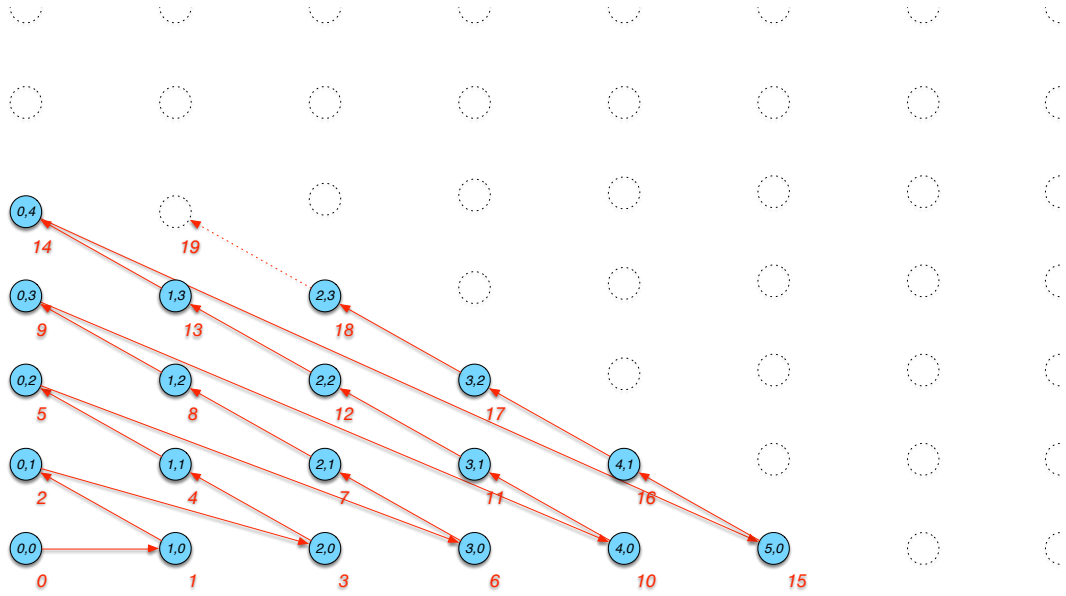
$$\alpha_2(x, y) = \frac{(x + y) \cdot (x + y + 1)}{2} + y.$$

This is obtained by looking at the following picture and noticing that

(1) $\alpha_2(x, y) = \alpha_2(x + y, 0) + y$, and

(2) $\alpha_2(x + y, 0) = 1 + 2 + \cdots + (x + y)$

$$= \frac{1}{2}\begin{pmatrix} 1 & & 2 & & & & x+y \\ + & + & + & + & \cdots & + & + \\ x+y & & x+y-1 & & & & 1 \end{pmatrix}.$$



We have

(1) $\beta_2^1(n) = \mu x \leqslant n \quad \exists t \leqslant n \;\; \alpha_2(x,t) = n$

(2) $\beta_2^2(n) = \mu y \leqslant n \quad \exists t \leqslant n \;\; \alpha_2(t,y) = n.$

Then we define α_{p+1}, β_{p+1}^1, $\beta_{p+1}^2, \ldots\ldots$, β_{p+1}^{p-1}, β_{p+1}^p and β_{p+1}^{p+1} by induction on $p \in \mathbb{N}$:

- $\alpha_{p+1}(x_1, \ldots, x_p, x_{p+1}) = \alpha_p\big(x_1, \ldots, x_{p-1}, \alpha_2(x_p, x_{p+1})\big)$
- $\beta_{p+1}^1 = \beta_p^1$;
- $\beta_{p+1}^2 = \beta_p^2$;

$\vdots$

- $\beta_{p+1}^{p-1} = \beta_p^{p-1}$;
- $\beta_{p+1}^p = \beta_2^1 \circ \beta_p^p$;
- $\beta_{p+1}^{p+1} = \beta_2^2 \circ \beta_p^p$.

□

Example 4.1

A different way of coding sequences of integers:

$$\begin{cases} c(\varepsilon) & = & 1 \\ c(x_0,\ldots,x_p) & = & \Pi\,(0)^{x_0+1}\cdot\Pi\,(1)^{x_1+1}\cdots\Pi\,(p)^{x_p+1}. \end{cases}$$

From $n \in \mathbb{N}\setminus\{0\}$ we recover the sequence $\langle x_0,\ldots,x_p\rangle$ such that $c(x_0,\ldots,x_p)=n$ by considering the $\mathcal{P}rim.\ \mathcal{R}ec.$ function $d \in \mathbb{N}^{(\mathbb{N}^2)}$ which yields the exponents of the prime numbers:

$$d(i,n) = \mu x \leqslant n \quad \Pi\,(i)^{x+1} \textit{ does not divide } n.$$

3.5 Partial Recursive Functions

We recall that

(1) (dom_f, f) is a *partial function* $\mathbb{N}^p \longrightarrow \mathbb{N}$ if f is a mapping $dom_f \longrightarrow \mathbb{N}$ where $dom_f \subseteq \mathbb{N}^p$.

(2) (dom_f, f) is a *total function* $\mathbb{N}^p \longrightarrow \mathbb{N}$ if $dom_f = \mathbb{N}^p$ holds.

We say that f is undefined on x – or $f(x)$ is undefined – if $x \notin dom_f$. We use the notation $f \in \mathbb{N}^{(dom\subseteq\mathbb{N}^p)}$ to signify that (dom_f, f) is a *partial function* $\mathbb{N}^p \longrightarrow \mathbb{N}$ whose domain is dom_f. Notice that for any two partial functions $f, g \in \mathbb{N}^{(dom\subseteq\mathbb{N}^p)}$:

$$f = g \textit{ holds} \iff \begin{cases} dom_f & = & dom_g \\ & \textit{and} & \\ \forall x & f(x) = g(x). \end{cases}$$

Definition 5.1: Composition

Given $f_1,\ldots,f_n \in \mathbb{N}^{(dom\subseteq\mathbb{N}^p)}$ and $g \in \mathbb{N}^{(dom\subseteq\mathbb{N}^n)}$, the *composition* $h = g(f_1,\ldots,f_n) \in \mathbb{N}^{(\mathbb{N}^p)}$ is

$$\begin{cases} h(\overrightarrow{x}) \textit{ is undefined iff} \begin{cases} \overrightarrow{x} \notin \bigcap\limits_{1\leqslant i\leqslant n} dom_{f_i} \\ \textit{or otherwise} \\ \big(f_1(\overrightarrow{x}),\ldots,f_n(\overrightarrow{x})\big) \notin dom_g. \end{cases} \\ h(\overrightarrow{x}) \textit{ is defined otherwise and } h(\overrightarrow{x}) = g\big(f_1(\overrightarrow{x}),\ldots,f_n(\overrightarrow{x})\big). \end{cases}$$

$$= g\big(f_1(x_1,\ldots,x_p),\ldots,f_n(x_1,\ldots,x_p)\big)$$

Definition 5.2: Recursion

Given $g \in \mathbb{N}^{(dom \subseteq \mathbb{N}^p)}$ and $h \in \mathbb{N}^{(dom \subseteq \mathbb{N}^{p+2})}$, there exists a unique $f \in \mathbb{N}^{(dom \subseteq \mathbb{N}^{p+1})}$ such that for all $\overrightarrow{x} \in \mathbb{N}^p$ and $y \in \mathbb{N}$:

(1)
$$\begin{cases} f(\overrightarrow{x}, 0) \text{ is undefined if } \overrightarrow{x} \notin dom_g \\ \text{and} \\ f(\overrightarrow{x}, 0) \text{ is defined otherwise with } f(\overrightarrow{x}, 0) = g(\overrightarrow{x}). \end{cases}$$

(2)
$$\begin{cases} f(\overrightarrow{x}, y+1) \text{ is undefined if } \begin{cases} (\overrightarrow{x}, y) \notin dom_f \\ \text{or} \\ \big(\overrightarrow{x}, y, f(\overrightarrow{x}, y)\big) \notin dom_h. \end{cases} \\ \text{and} \\ \text{otherwise } f(\overrightarrow{x}, y+1) \text{ is defined and } f(\overrightarrow{x}, y+1) = h\big(\overrightarrow{x}, y, f(\overrightarrow{x}, y)\big). \end{cases}$$

Definition 5.3: Minimization

Given $f \in \mathbb{N}^{(dom \subseteq \mathbb{N}^{p+1})}$, we define $g \in \mathbb{N}^{(dom \subseteq \mathbb{N}^p)}$ by:

$$g(\overrightarrow{x}) = \mu y \quad f(\overrightarrow{x}, y) = 0.$$

Notice that

$$g(\overrightarrow{x}) = y \iff \begin{cases} \forall z < y \begin{cases} f(\overrightarrow{x}, z) \text{ is defined!} \\ \text{and} \\ f(\overrightarrow{x}, z) > 0 \end{cases} \\ \text{and} \\ f(\overrightarrow{x}, y) = 0. \end{cases}$$

Definition 5.4: Partial recursive functions

The set of *partial recursive* ($\mathcal{P}art.\ \mathcal{R}ec.$) functions is the least that

(1) contains:

(a) All constants $\mathbb{N}^p \longrightarrow \mathbb{N}$ (all $\bar{i} \in \mathbb{N}^{(\mathbb{N}^p)}$ s.t. $\bar{i}(\vec{x}) = i$ – any $i, p \in \mathbb{N}$).

(b) All projections π_i^p (any $p \in \mathbb{N}$, any $1 \leqslant i \leqslant p$)

(c) The successor function $S \in \mathbb{N}^{\mathbb{N}}$.

(2) and is closed under

(a) composition

(b) recursion

(c) minimisation.

Our next goal is to show that a function f is in $\mathcal{P}art.\ \mathcal{R}ec.$ if and only if it is Turing computable. One direction is easy, the other one is more involved. One side effect of our proof will show that every partial recursive function can be obtained by applying the minimisation at most once.

Lemma 5.1

Every partial recursive function is Turing computable.

Proof of Lemma 5.1:

We need to show that given any $p \in \mathbb{N} \setminus \{0\}$ and any $f \in \mathbb{N}^{(dom \subseteq \mathbb{N}^p)}$ there exists some Turing machine $\mathcal{M}$ that computes f. This means that on input $(n_1, \ldots, n_p)$ it stops in configuration $q_{acc.} f(n_1, \ldots, n_p)$ if $(n_1, \ldots, n_p) \in dom_f$, and it never stops otherwise. Of course, we need to fix a certain representation of both integers and finite sequence of integers. For simplicity, let us say that the integers are represented in base-ten and the sequences as $(n_1, \ldots, n_p)$ so that the input alphabet is

$$\Sigma = \Big\{0,\ 1,\ 2,\ 3,\ 4,\ 5,\ 6,\ 7,\ 8,\ 9,\ (,\ ),\ ,\Big\}.$$

So, for instance a TM computes *Add* if on input word "$(385, 218)$" it returns the word "603".
We do the proof by induction on the number of operation among

- composition
- recursion
- minimisation

that are necessary to obtain $f \in \mathbb{N}^{(dom \subseteq \mathbb{N}^p)}$ on the basis of

- all constants
- all projections
- the successor function.

(1) It is quite obvious that

- if $f \in \mathbb{N}^{(dom \subseteq \mathbb{N}^p)}$ is constant, then there exists some basic TM that computes it.
- Every projection π_i^p (any $1 \leqslant i \leqslant p$) is also trivially computable.
- The successor function is clearly computable as well.

(2) (a) Assume $f_1, \ldots, f_n \in \mathbb{N}^{(dom \subseteq \mathbb{N}^p)}$ are computed respectively by $\mathcal{M}_{f_1}, \ldots, \mathcal{M}_{f_n}$ and $g \in \mathbb{N}^{(dom \subseteq \mathbb{N}^n)}$ is computed by $\mathcal{M}_g$. Then $f = g(f_1, \ldots, f_n) \in \mathbb{N}^{(\mathbb{N}^p)}$ is computed by $\mathcal{M}_f$ which works as follows:
on input $\overrightarrow{x} = (n_1, \ldots, n_p)$:

successively for each i $:= \mathbf{1}, \ldots, \mathbf{n}$ $\mathcal{M}_f$ simulates $\mathcal{M}_{f_i}$ on input $\overrightarrow{x}$, if $\mathcal{M}_{f_i}(\overrightarrow{x}) \downarrow \overset{\text{acc.}}{}$ with some output m_i it stores m_i.

(In case all simulation of machines $\mathcal{M}_{f_1}, \ldots, \mathcal{M}_{f_n}$ do stop) $\mathcal{M}_f$ finally simulates $\mathcal{M}_g$ on input $(m_1, \ldots, m_n)$.

It is clear that if either

$$\text{(A)}\ \ \overrightarrow{x} \notin \bigcap_{1 \leqslant i \leqslant n} dom_{f_i} \qquad \text{or} \qquad \text{(B)}\ \ \Big(f_1(\overrightarrow{x}), \ldots, f_n(\overrightarrow{x})\Big) \notin dom_g,$$

then $\mathcal{M}_f(\overrightarrow{x}) \uparrow$. In the opposite case, $\mathcal{M}_f(\overrightarrow{x}) \overset{\text{acc.}}{\downarrow}$ with the right answer.

(b) Assume $g \in \mathbb{N}^{(dom \subseteq \mathbb{N}^p)}$ and $h \in \mathbb{N}^{(dom \subseteq \mathbb{N}^{p+2})}$ are computed respectively by $\mathcal{M}_g$ and $\mathcal{M}_h$. Then f defined by recursion:

(A) $f(\overrightarrow{x}, 0) = g(\overrightarrow{x})$

(B) $f(\overrightarrow{x}, y+1) = h\big(\overrightarrow{x}, y, f(\overrightarrow{x}, y)\big)$

is computed by $\mathcal{M}_f$ which works as follows:

- on input $(\overrightarrow{x}, 0)$ it simply simulates $\mathcal{M}_g$ on input $\overrightarrow{x}$, and
- on input $(\overrightarrow{x}, n+1)$ $\mathcal{M}_f$ first simulates $\mathcal{M}_g$ on input $\overrightarrow{x}$ which gives[a] $f(\overrightarrow{x}, 0)$. Then

  recursively for i $:= \mathbf{0}, \ldots, \mathbf{n}$ $\mathcal{M}_f$ simulates $\mathcal{M}_h$ on $\big(\overrightarrow{x}, i, f(\overrightarrow{x}, i)\big)$ which yields[b] $f(\overrightarrow{x}, i+1)$.

It is clear that $\mathcal{M}_f$ brings the result if and only if every step $f(\overrightarrow{x}, i)$ ($i := 0, \ldots, n+1$) is defined. Otherwise it simply never stops.

(c) Assume $g \in \mathbb{N}^{(dom \subseteq \mathbb{N}^{p+1})}$ is computed by $\mathcal{M}_g$. We design $\mathcal{M}_f$ that on input $\overrightarrow{x}$ computes

$$\mu y \quad g(\overrightarrow{x}, y) = 0.$$

set i $:= \mathbf{0}$ $\mathcal{M}_f$ simulates $\mathcal{M}_g$ on input $(\overrightarrow{x}, i)$. If $\mathcal{M}_g$ stops and outputs the value of $g(\overrightarrow{x}, i)$,

- if $g(\overrightarrow{x}, i) = 0$ $\mathcal{M}_f$ stops and outputs i,

- if $g(\overrightarrow{x}, i) \neq 0$, $\mathcal{M}_f$ starts over again with $i := i + 1$.

Notice that $\mathcal{M}_f$ stops and outputs $n = \mu y \quad g(\overrightarrow{x}, y) = 0$ if and only if

- $\forall i \leqslant n \quad (\overrightarrow{x}, i) \in dom_g$,
- $\forall i < n \quad g(\overrightarrow{x}, i) > 0$ and
- $g(\overrightarrow{x}, n) = 0$.

□

[a] in case $\mathcal{M}_g(\overrightarrow{x}) \overset{\text{acc.}}{\downarrow}$.

[b] in case $\mathcal{M}_h\big(\overrightarrow{x}, i, f(\overrightarrow{x}, i)\big) \overset{\text{acc.}}{\downarrow}$.

Lemma 5.2

Every Turing computable partial function $f \in \mathbb{N}^{(dom \subseteq \mathbb{N}^p)}$ is $\mathcal{P}art.\ \mathcal{R}ec.$

Proof of Lemma 5.2:

We show an even stronger result: given any Turing Machine $\mathcal{M}$ and any recursive coding of words on the tape alphabet Γ we show that the partial function $\Sigma^* \longrightarrow \Gamma^*$ which maps $v \in \Sigma^*$ to $w \in \Gamma^*$ if and only if the Turing machine from the initial configuration $q_0 v$ stops in some configuration $w_0 q_{acc.} w_1$ with $w_0 w_1 = w$ is partial recursive in the code. This means the function $f' \in \mathbb{N}^{(dom \subseteq \mathbb{N}^1)}$ defined by

$$\begin{cases} f'\big(code(v)\big) \text{ is undefined if } \mathcal{M}(v) \uparrow \text{ or } \mathcal{M}(v) \overset{\text{fin.}}{\downarrow}; \\ f'\big(code(v)\big) = code(w_0 w_1) \text{ if } \mathcal{M} \overset{\text{acc.}}{\downarrow} \text{ in config. } w_0 q_{acc.} w_1. \end{cases}$$

We first choose a coding of the configurations of $\mathcal{M}$: We assume

- $\Sigma = \{1, \ldots, k-1\}$ and $\Gamma = \{0, \ldots, k-1\}$ with $k > 1$ and necessarily $0 = \sqcup$.
- $Q = \{q_0, \ldots, q_m\}$ with q_0, q_1, q_2 being respectively the initial state, the rejecting state and the accepting state.

The coding of a word $w = a_0 \ldots a_n$ that we choose is

$$\ulcorner a_0 \ldots a_n \urcorner = \sum_{0 \leqslant i \leqslant n} a_i \cdot k^i.$$

This coding is not injective (this will not matter for our purpose) for any two word which differ by the tailing blanks in their prefix will have the same encoding:

$$\ulcorner a_0 \dots a_n \urcorner = \ulcorner a_0 \dots a_n \sqcup \urcorner.$$

But on the other hand, this encoding is surjective.
A configuration $w_0 q_r w_1$ of the Turing machine will be coded by

$$\ulcorner w_0 q_r w_1 \urcorner = \alpha_4(r, \ulcorner w_0 \urcorner, \ulcorner w_1 \urcorner, |w_0|).$$

For instance, the initial configuration of the Turing machine with input w is:

$$\begin{aligned}
\ulcorner q_0 w \urcorner &= \alpha_4(0, 0, \ulcorner w \urcorner, 0) \\
&= \alpha_2\Big(0, \alpha_2\big(0, \alpha_2(\ulcorner w \urcorner, 0)\big)\Big) \\
&= \alpha_2\Big(0, \alpha_2\big(0, \tfrac{\ulcorner w \urcorner(\ulcorner w \urcorner+1)}{2}\big)\Big) \\
&= \alpha_2\Big(0, \tfrac{\left(\frac{\ulcorner w \urcorner(\ulcorner w \urcorner+1)}{2}\right)\left(\frac{\ulcorner w \urcorner(\ulcorner w \urcorner+1)}{2}+1\right)}{2}\Big) \\
&= \frac{\left(\frac{\left(\frac{\ulcorner w \urcorner(\ulcorner w \urcorner+1)}{2}\right)\left(\frac{\ulcorner w \urcorner(\ulcorner w \urcorner+1)}{2}+1\right)}{2}\right).\left(\frac{\left(\frac{\ulcorner w \urcorner(\ulcorner w \urcorner+1)}{2}\right)\left(\frac{\ulcorner w \urcorner(\ulcorner w \urcorner+1)}{2}+1\right)}{2}+1\right)}{2}.
\end{aligned}$$

To say that w is an input word is to say that $w \in \Sigma^*$ (we will identify words of the form $w \in \Sigma^*$ with words of the form $w \sqcup \dots \sqcup$ since our coding will not be able to distinguish them[a] and also because the input word really is the infinite word $w \sqcup \dots \sqcup \dots$).
So we see that a word $w = a_0 \dots a_n \in \Gamma^*$ is an input word if for no $i < n$ we have both $a_i = \sqcup$ and $a_{i+1} \neq \sqcup$. This means that $\ulcorner a_0 \dots a_n \urcorner = a_0 \cdot k^0 + a_1 \cdot k^1 + \dots + a_n \cdot k^n$ satisfies $a_i = 0 \Rightarrow a_{i+1} = 0$ (any $i < n$). With our coding, we recover the coefficient a_i as

$$a_i = \left[\frac{\ulcorner a_0 \dots a_n \urcorner}{k^i}\right] \dot{-} \left(\left[\frac{\ulcorner a_0 \dots a_n \urcorner}{k^{i+1}}\right] \cdot k\right).$$

Therefore the set $Input_{\mathcal{M}}$ of all the codes of input words of $\mathcal{M}$ is $\mathcal{P}rim.\ \mathcal{R}ec.$:

$$\begin{aligned}
\chi_{Input_{\mathcal{M}}}(m) &= 1 \quad \textit{if} \quad \forall i \leqslant m \left(\left[\frac{m}{k^i}\right] \dot{-} \left(\left[\frac{m}{k^{i+1}}\right] \cdot k\right) = 0 \Longrightarrow \left[\frac{m}{k^{i+1}}\right] \dot{-} \left(\left[\frac{m}{k^{i+2}}\right] \cdot k\right) = 0\right) \\
&= 0 \quad \textit{otherwise}
\end{aligned}$$

Since the coding of words that we choose is surjective, it comes that a configuration C is an initial configuration if and only if there exists $m \in Input_{\mathcal{M}}$

$$\ulcorner C \urcorner = \frac{\left(\frac{\left(\frac{m(m+1)}{2}\right)\left(\frac{m(m+1)}{2}+1\right)}{2}\right) \cdot \left(\frac{\left(\frac{m(m+1)}{2}\right)\left(\frac{m(m+1)}{2}+1\right)}{2}+1\right)}{2}.$$

Therefore the set $Init_{\mathcal{M}}$ of all the codes of initial configurations of $\mathcal{M}$ is $\mathcal{P}rim.\ \mathcal{R}ec.$

$$\chi_{Init_{\mathcal{M}}}(c) = \begin{cases} 1 & if \quad \exists m \leqslant c \quad \big(\chi_{Input_{\mathcal{M}}}(m) = 1 \ \wedge \ \alpha_4(0,0,m,0) = c\big), \\ 0 & otherwise. \end{cases}$$

We now describe the transition from a given configuration C to the next configuration C' ($C \Rightarrow C'$), in other words, we analyse the we obtain the code of C' on the basis of both the code of C and the transition function δ.
A transition yields a move of the head either to the right or to the left: $\delta(q_r, a_l) = (q_{r'}, a_{l'}, R)$ or $\delta(q_r, a_l) = (q_{r'}, a_{l'}, L)$. We need to consider differently these two forms, together with differentiating also whether the head can or cannot move to the left when the transition function says so[b].

When $\delta(\mathbf{q_r}, \mathbf{a_l}) = (\mathbf{q_{r'}}, \mathbf{a_{l'}}, \mathbf{R})$, we have $C \Rightarrow C'$ is $w_0 q_r w_1 \Rightarrow w'_0 q_{r'} w'_1$ with

(1) $w'_0 = w_0 a_{l'}$ (2) $|w'_0| = 1 + |w_0|$ (3) $a_l w'_1 = w_1$.

so that

(1) $\ulcorner w'_0 \urcorner = \ulcorner w_0 \urcorner + l' \cdot k^{|w_0|} = \beta_4^2(\ulcorner C \urcorner) + l' \cdot k^{\beta_4^4(\ulcorner C \urcorner)}$

(2) $|w'_0| = |w_0| + 1 = \beta_4^4(\ulcorner C \urcorner) + 1$

(3) $a_l w'_1 = w_1$ so that $\ulcorner w'_1 \urcorner = \left[\dfrac{\beta_4^3(\ulcorner C \urcorner)}{k}\right]$.

So, we obtain:

$$if \begin{cases} \beta_4^1(\ulcorner C \urcorner) = r \\ and \\ \beta_4^3(\ulcorner C \urcorner) \dot{-} \left(k \cdot \left[\dfrac{\beta_4^3(\ulcorner C \urcorner)}{k}\right]\right) = l, \end{cases}$$

then

$$\ulcorner C' \urcorner = \alpha_4\left(r', \ \beta_4^2(\ulcorner C \urcorner) + l' \cdot k^{\beta_4^4(\ulcorner C \urcorner)}, \ \left[\frac{\beta_4^3(\ulcorner C \urcorner)}{k}\right], \ \beta_4^4(\ulcorner C \urcorner) + 1\right).$$

When $\delta(\mathbf{q_r}, \mathbf{a_l}) = (\mathbf{q_{r'}}, \mathbf{a_{l'}}, \mathbf{L})$, there are two different cases depending on whether the head can move to the left or not.

if $\mathbf{w_0} = \varepsilon$, then we have $C \Rightarrow C'$ is $w_0 q_r w_1 \Rightarrow w'_0 q_{r'} w'_1$ with

(1) $w'_0 = \varepsilon$
(2) $|w'_0| = |w_0| = |\varepsilon|$
(3) $w'_1 = a_{l'} v$ and $w_1 = a_l v$ for some word v.

So that we get

(1) $\ulcorner w'_0 \urcorner = \ulcorner \varepsilon \urcorner = 0$
(2) $|w'_0| = |\varepsilon| = 0$

(3) $\ulcorner w'_1 \urcorner = l' + \left[\dfrac{\beta_4^3(\ulcorner C \urcorner)}{k} \right] \cdot k.$

So, all in all we obtain:

$$if \begin{cases} \beta_4^1(\ulcorner C \urcorner) = r \\ and \\ \beta_4^3(\ulcorner C \urcorner) \dot{-} \left(k \cdot \left[\dfrac{\beta_4^3(\ulcorner C \urcorner)}{k} \right] \right) = l, \\ and \\ \beta_4^4(\ulcorner C \urcorner) = 0 \end{cases}$$

then

$$\ulcorner C' \urcorner = \alpha_4 \left(r', \quad 0, \quad l' + \left[\frac{\beta_4^3(\ulcorner C \urcorner)}{k} \right] \cdot k, \quad 0 \right).$$

if $\mathbf{w_0} \neq \varepsilon$, then we have $C \Rightarrow C'$ is $w_0 q_r w_1 \Rightarrow w'_0 q_{r'} w'_1$ with

(1) $w'_0 a_l = w_0$
(2) $|w'_0| = |w_0| - 1$

(3) $w'_1 = a_{l'} w_1.$

so that

(1) $\ulcorner w'_0 \urcorner = \beta_4^2(\ulcorner C \urcorner) \dot{-} \left[\dfrac{\beta_4^2(\ulcorner C \urcorner)}{k^{(\beta_4^4(\ulcorner C \urcorner) \dot{-} 1)}} \right] \cdot k^{(\beta_4^4(\ulcorner C \urcorner) \dot{-} 1)}$

(2) $|w'_0| = \beta_4^4(\ulcorner C \urcorner) \dot{-} 1$

(3) $\ulcorner w'_1 \urcorner = l' + \beta_4^3(\ulcorner C \urcorner) \cdot k.$

So, we have found:

$$\textit{if} \left[\begin{array}{c} \beta_4^1(\ulcorner C \urcorner) = r \\ \textit{and} \\ \beta_4^3(\ulcorner C \urcorner) \dot{-} \left(k \cdot \left[\dfrac{\beta_4^3(\ulcorner C \urcorner)}{k}\right]\right) = l \\ \textit{and} \\ \beta_4^4(\ulcorner C \urcorner) \neq 0 \end{array}\right], \text{ then}$$

$$\ulcorner C' \urcorner = \alpha_4 \left(r', \quad \beta_4^2(\ulcorner C \urcorner) \dot{-} \left[\frac{\beta_4^2(\ulcorner C \urcorner)}{k^{(\beta_4^4(\ulcorner C \urcorner) \dot{-} 1)}}\right] \cdot k^{(\beta_4^4(\ulcorner C \urcorner) \dot{-} 1)}, \quad l' + \beta_4^3(\ulcorner C \urcorner) \cdot k, \quad \beta_4^4(\ulcorner C \urcorner) \dot{-} 1\right).$$

To wrap up everything that we did so far, we recursively define a mapping $f : \mathbb{N}^2 \longrightarrow \mathbb{N}$ such that if n codes a word on Σ – i.e., $n = \ulcorner w \urcorner$ for some $w \in \Sigma^*$ – then $f(n,t) = \ulcorner C_{n,t} \urcorner$ where $C_{n,t}$ stands for the configuration that the Turing machine reaches after t-many steps from the initial configuration $q_0 w$.
Since the machine stops if it reaches an accepting or a rejecting configuration, we will simply assume that in any of these two cases the configuration of the Turing machines remains the same: if $C_{n,t}$ is either an accepting or a rejecting configuration, then $C_{n,t+x} = C_{n,t}$ holds for every $x \in \mathbb{N}$.
We set δ_L and δ_R are the two following finite – hence $\mathcal{P}rim.\ \mathcal{R}ec.$ – subsets of $\mathbb{N}^4$:

$$\delta_L = \Big\{(r,l,r',l') \in \{0,\dots m\} \times \{0 \dots k-1\} \times \{0,\dots m\} \times \{0 \dots k-1\} \ \mid \ \delta(r,l) = (r',l',L)\Big\}$$

and

$$\delta_R = \Big\{(r,l,r',l') \in \{0,\dots m\} \times \{0 \dots k-1\} \times \{0,\dots m\} \times \{0 \dots k-1\} \ \mid \ \delta(r,l) = (r',l',R)\Big\}$$

For our convenience we assume that $\Gamma = \Sigma \cup \{\sqcup\}$. This way the mapping $f : \mathbb{N}^2 \longrightarrow \mathbb{N}$ we currently construct is total ($f \in \mathbb{N}^{(\mathbb{N}^2)}$). For readability we use the notation $\overline{k}^i$ for $\beta_4^i(k)$ (any $k \in \mathbb{N}$, $i \leqslant 4$).

initial case

$$f(n,0) = \begin{cases} 1 & \text{if} \quad n \notin \mathit{Input}_{\mathcal{M}} \\ \alpha_4(0,0,n,0) & \text{if} \quad n \in \mathit{Input}_{\mathcal{M}} \end{cases}$$

successor case

$$f(n,t+1) =$$

$$\begin{cases} f(n,t) & \text{if} \quad \overline{f(n,t)}^1 = 1 \quad \text{or} \quad \overline{f(n,t)}^1 = 2; \\ \alpha_4\left(r',\ \overline{f(n,t)}^2 + l' \cdot k^{\overline{f(n,t)}^4},\ \left[\frac{\overline{f(n,t)}^3}{k}\right],\ \overline{f(n,t)}^4 + 1\right) & \text{if} \quad \left(\overline{f(n,t)}^1,\ \overline{f(n,t)}^3 \dot{-} \left(k \cdot \left[\frac{\overline{f(n,t)}^3}{k}\right]\right),\ r',\ l'\right) \in \delta_R; \\ \alpha_4\left(r',\ 0,\ l' + \left[\frac{\overline{f(n,t)}^3}{k}\right] \cdot k,\ 0\right) & \text{if} \quad \begin{cases} \left(\overline{f(n,t)}^1,\ \overline{f(n,t)}^3 \dot{-} \left(k \cdot \left[\frac{\overline{f(n,t)}^3}{k}\right]\right),\ r',\ l'\right) \in \delta_L \\ \text{and} \\ \overline{f(n,t)}^4 = 0; \end{cases} \\ \alpha_4\left(r',\ \overline{f(n,t)}^2 \dot{-} \left[\frac{\overline{f(n,t)}^2}{k^{\left(\overline{f(n,t)}^4 \dot{-} 1\right)}}\right] \cdot k^{\left(\overline{f(n,t)}^4 \dot{-} 1\right)},\ l' + \overline{f(n,t)}^3 \cdot k,\ \overline{f(n,t)}^4 \dot{-} 1\right) & \text{if} \quad \begin{cases} \left(\overline{f(n,t)}^1,\ \overline{f(n,t)}^3 \dot{-} \left(k \cdot \left[\frac{\overline{f(n,t)}^3}{k}\right]\right),\ r',\ l'\right) \in \delta_L \\ \text{and} \\ \overline{f(n,t)}^4 \neq 0. \end{cases} \end{cases}$$

Notice that $f \in \mathbb{N}^{(\mathbb{N}^2)}$ is $\mathcal{P}rim.\ \mathcal{R}ec.$ and that $\mathcal{M}(w) \overset{\text{acc}}{\downarrow}$ if and only if there exists some $t \in \mathbb{N}$ such that $\beta_4^1\big(f(\ulcorner w \urcorner, t)\big) = 2$.
Moreover, in this case, we recover the code of the content of the tape w_t from $f(\ulcorner w \urcorner, t)$ by

$$\ulcorner w_t \urcorner \quad = \quad \beta_4^2\big(f(\ulcorner w \urcorner, t)\big) \quad + \quad k^{\beta_4^4\left(f(\ulcorner w \urcorner, t)\right)} \cdot \beta_4^3\big(f(\ulcorner w \urcorner, t)\big).$$

Finally, we only need to fix both (1) a recursive representation of the natural numbers and (2) what it means for a machine to compute a partial function.

(1) we fix our coding of integers: every integer n is coded by the word $\underbrace{11\ldots1}_{n}$. The function

$$\begin{array}{rccc} \ulcorner\ \urcorner : & \mathbb{N} & \longrightarrow & \mathbb{N} \\ & n & \longmapsto & \ulcorner n \urcorner \end{array}$$

such that $\ulcorner n \urcorner = \sum_{i<n} 1 \cdot k^i$ (i.e., $\ulcorner n \urcorner$ is the code of the word $\underbrace{11\ldots1}_{n}$) is $\mathcal{P}rim.\ \mathcal{R}ec.$:

$$\begin{cases} \ulcorner 0 \urcorner & = & 0 \\ \ulcorner n+1 \urcorner & = & \ulcorner n \urcorner + k^n. \end{cases}$$

(2) We only consider Turing machines that on any input words of the form $\overbrace{11\ldots1}^{n}$, provided they reach an accepting configuration, they reach one of the form $\underbrace{11\ldots1}_{n'} q_{acc.}$.

We define the function $f_{\mathcal{M}} \in \mathbb{N}^{(dom \subseteq \mathbb{N}^r)}$ that $\mathcal{M}$ computes by:

- $f_{\mathcal{M}}(n_1, \ldots, n_r)$ is undefined if on input $\overbrace{11\ldots1}^{\alpha_r(n_1,\ldots,n_r)}$ either $\mathcal{M} \uparrow$ or $\mathcal{M} \not\downarrow$;
- $f_{\mathcal{M}}(n_1, \ldots, n_r) = n$ if on input $\overbrace{11\ldots1}^{\alpha_r(n_1,\ldots,n_r)}$ $\mathcal{M} \downarrow$ in configuration $\underbrace{11\ldots1}_{n} q_{acc.}$.

Then the function $least_{\mathcal{M}} \in \mathbb{N}^{(dom \subseteq \mathbb{N}^r)}$ that picks the minimum number of steps – if any – the machine takes before reaching an accepting configuration when starting from the initial one $q_0 \underbrace{11\ldots\ldots1}_{\alpha_r(n_1,\ldots,n_r)}$ is defined by

$$least_{\mathcal{M}}(n_1, \ldots, n_r) = \mu t \quad \beta_4^1 \circ f\big(\ulcorner \alpha_r(n_1, \ldots, n_r) \urcorner, t\big) = 2.$$

It is undefined if the machine never halts or halts on the rejecting state.

At last, we are ready to provide the desired $f_{\mathcal{M}} \in \mathbb{N}^{(dom \subseteq \mathbb{N}^r)}$. We make use of the fact the position of the head in an accepting configuration indicates precisely the number of 1's there are on the tape:

$$f_{\mathcal{M}}(n_1, \ldots, n_r) = \beta_4^4 \circ f\Big(\ulcorner \alpha_r(n_1, \ldots, n_r) \urcorner, least_{\mathcal{M}}(n_1, \ldots, n_r)\Big).$$

□

[a] remember that this is the reason why we chose 0 for the coding of the blank symbol.

[b] this means whether or not the head is already in position 0 and the transition is of the form $\delta(q_r, a_l) = (q_{r'}, a_{l'}, L)$.

Corollary 5.1

Every partial recursive function $f \in \mathbb{N}^{(\mathbb{N}^r)}$ admits a construction that requires minimisation

at most once.
Moreover, one has $\forall n_1 \dots \forall n_r \forall k \quad \Big(f(n_1, \dots, n_r) = k \longleftrightarrow \exists t \quad F(n_1, \dots, n_r, k, t)\Big)$ where $F \subseteq \mathbb{N}^{r+2}$ is $\mathcal{P}rim.\ \mathcal{R}ec.$.

Proof of Corollary 5.1:

This is an immediate consequence of the whole proof of Lemma 5.2 since we proved that every partial recursive function can be computed by a TM whose function it computes is $f_{\mathcal{M}} \in \mathbb{N}^{(\mathbb{N}^r)}$ defined by

$$f_{\mathcal{M}}(n_1, \dots, n_r) = \beta_4^4 \circ f\Big(\ulcorner \alpha_r(n_1, \dots, n_r) \urcorner, \mu t \quad \beta_4^1 \circ f\big(\ulcorner \alpha_r(n_1, \dots, n_r) \urcorner, t\big) = 2\Big).$$

where all functions $f \in \mathbb{N}^{(\mathbb{N}^2)}$, $\alpha_r \in \mathbb{N}^{(\mathbb{N}^r)}$, $\beta_4^1 \in \mathbb{N}^{\mathbb{N}}$, $\beta_4^4 \in \mathbb{N}^{\mathbb{N}}$, $\ulcorner\ \urcorner \in \mathbb{N}^{\mathbb{N}}$ and the constant $\overline{2} \in \mathbb{N}^{(\mathbb{N}^0)}$ are $\mathcal{P}rim.\ \mathcal{R}ec.$ as well as the equality relation.

□

Theorem 5.1

For every $k > 0$ and every $f \in \mathbb{N}^{(dom \subseteq \mathbb{N}^k)}$ the following are equivalent

- f is $\mathcal{P}art.\ \mathcal{R}ec.$,
- f is Turing computable.

Proof of Theorem 5.1:

Follows immediately from Lemmas 5.1 and 5.2. □