Coding Programs as Numbers

Turing/Church solution of the Etscheidungsproblem uses the idea that (formal descriptions of) algorithms can be the data on which algorithms act.

To realize this idea with Register Machines we have to be able to code RM programs as numbers. (In general, such codings are often called Gödel numberings.)

To do that, first we have to code pairs of numbers and lists of numbers as numbers. There are many ways to do that. We fix upon one...

Numerical coding of pairs

For $x, y \in \mathbb{N}$, define $\begin{cases} \langle\langle x, y \rangle\rangle \triangleq 2^x(2y+1) \\ \langle x, y \rangle \triangleq 2^x(2y+1) - 1 \end{cases}$

"equals, by definition"

Numerical coding of pairs

For $x, y \in \mathbb{N}$, define $\begin{cases} \langle\langle x, y \rangle\rangle \triangleq 2^x(2y+1) \\ \langle x, y \rangle \triangleq 2^x(2y+1) - 1 \end{cases}$

So

x 0's

$0\mathrm{b}\langle\langle x, y\rangle\rangle$	$=$	$0\mathrm{b}y$	1	$0 \cdots 0$
$0\mathrm{b}\langle x, y\rangle$	$=$	$0\mathrm{b}y$	0	$1 \cdots 1$

x 1's

(Notation: $0\mathrm{b}x \triangleq$ *x in binary*.)

E.g. $27 = 0\mathrm{b}11011 = \langle\langle 0, 13 \rangle\rangle = \langle 2, 3 \rangle$

Numerical coding of pairs

For $x, y \in \mathbb{N}$, define $\begin{cases} \langle\!\langle x, y \rangle\!\rangle \triangleq 2^x(2y+1) \\ \langle x, y \rangle \triangleq 2^x(2y+1) - 1 \end{cases}$

So

$0\mathrm{b}\langle\!\langle x, y \rangle\!\rangle$	$=$	$0\mathrm{b}y$	1	$0 \cdots 0$
$0\mathrm{b}\langle x, y \rangle$	$=$	$0\mathrm{b}y$	0	$1 \cdots 1$

$\langle -, - \rangle$ gives a bijection (one-one correspondence) between $\mathbb{N} \times \mathbb{N}$ and $\mathbb{N}$.

$\langle\!\langle -, - \rangle\!\rangle$ gives a bijection between $\mathbb{N} \times \mathbb{N}$ and $\{n \in \mathbb{N} \mid n \neq 0\}$.

Numerical coding of lists

$\mathit{list}\,\mathbb{N} \triangleq$ set of all finite lists of natural numbers, using ML notation for lists:

- empty list: $[]$
- list-cons: $x :: \ell \in \mathit{list}\,\mathbb{N}$ (given $x \in \mathbb{N}$ and $\ell \in \mathit{list}\,\mathbb{N}$)
- $[x_1, x_2, \ldots, x_n] \;\triangleq\; x_1 :: (x_2 :: (\cdots x_n :: [] \cdots))$

Numerical coding of lists

$list\,\mathbb{N} \triangleq$ set of all finite lists of natural numbers, using ML notation for lists.

For $\ell \in list\,\mathbb{N}$, define $\ulcorner \ell \urcorner \in \mathbb{N}$ by induction on the length of the list ℓ:

$$\begin{cases} \ulcorner [] \urcorner & \triangleq & 0 \\ \ulcorner x :: \ell \urcorner & \triangleq & \langle\langle x, \ulcorner \ell \urcorner \rangle\rangle = 2^x(2 \cdot \ulcorner \ell \urcorner + 1) \end{cases}$$

Thus $\ulcorner [x_1, x_2, \ldots, x_n] \urcorner = \langle\langle x_1, \langle\langle x_2, \cdots \langle\langle x_n, 0 \rangle\rangle \cdots \rangle\rangle \rangle\rangle$

Numerical coding of lists

$\mathit{list}\,\mathbb{N} \triangleq$ set of all finite lists of natural numbers, using ML notation for lists.

For $\ell \in \mathit{list}\,\mathbb{N}$, define $\ulcorner \ell \urcorner \in \mathbb{N}$ by induction on the length of the list ℓ:

$$\begin{cases} \ulcorner [] \urcorner & \triangleq & 0 \\ \ulcorner x :: \ell \urcorner & \triangleq & \langle\!\langle x, \ulcorner \ell \urcorner \rangle\!\rangle = 2^x (2 \cdot \ulcorner \ell \urcorner + 1) \end{cases}$$

For example:

$\ulcorner [3] \urcorner = \ulcorner 3 :: [] \urcorner = \langle\!\langle 3, 0 \rangle\!\rangle = 2^3 (2 \cdot 0 + 1) = 8 = 0\mathrm{b}1000$

$\ulcorner [1,3] \urcorner = \langle\!\langle 1, \ulcorner [3] \urcorner \rangle\!\rangle = \langle\!\langle 1, 8 \rangle\!\rangle = 34 = 0\mathrm{b}100010$

$\ulcorner [2,1,3] \urcorner = \langle\!\langle 2, \ulcorner [1,3] \urcorner \rangle\!\rangle = \langle\!\langle 2, 34 \rangle\!\rangle = 276 = 0\mathrm{b}100010100$

Numerical coding of lists

$\mathit{list}\,\mathbb{N} \triangleq$ set of all finite lists of natural numbers, using ML notation for lists.

For $\ell \in \mathit{list}\,\mathbb{N}$, define $\ulcorner \ell \urcorner \in \mathbb{N}$ by induction on the length of the list ℓ:

$$\begin{cases} \ulcorner [] \urcorner & \triangleq \mathbf{0} \\ \ulcorner x :: \ell \urcorner & \triangleq \langle\!\langle x, \ulcorner \ell \urcorner \rangle\!\rangle = 2^x(2 \cdot \ulcorner \ell \urcorner + 1) \end{cases}$$

For example:

$\ulcorner [3] \urcorner = \ulcorner 3 :: [] \urcorner = \langle\!\langle 3, 0 \rangle\!\rangle = 2^3(2 \cdot 0 + 1) = 8 = \mathtt{0b}1000$

$\ulcorner [1,3] \urcorner = \langle\!\langle 1, \ulcorner [3] \urcorner \rangle\!\rangle = \langle\!\langle 1, 8 \rangle\!\rangle = 34 = \mathtt{0b}100010$

$\ulcorner [2,1,3] \urcorner = \langle\!\langle 2, \ulcorner [1,3] \urcorner \rangle\!\rangle = \langle\!\langle 2, 34 \rangle\!\rangle = 276 = \mathtt{0b}100010100$

Numerical coding of lists

$\textit{list}\,\mathbb{N} \triangleq$ set of all finite lists of natural numbers, using ML notation for lists.

For $\ell \in \textit{list}\,\mathbb{N}$, define $\ulcorner \ell \urcorner \in \mathbb{N}$ by induction on the length of the list ℓ:

$$\begin{cases} \ulcorner [] \urcorner \triangleq 0 \\ \ulcorner x :: \ell \urcorner \triangleq \langle\langle x, \ulcorner \ell \urcorner \rangle\rangle = 2^x(2 \cdot \ulcorner \ell \urcorner + 1) \end{cases}$$

$$\boxed{0\mathrm{b}\ulcorner [x_1, x_2, \ldots, x_n] \urcorner} = \boxed{1 \mid \underbrace{0 \cdots 0}_{x_n \text{ 0's}}}\ \boxed{1 \mid \underbrace{0 \cdots 0}_{x_{n-1} \text{ 0's}}} \cdots \boxed{1 \mid \underbrace{0 \cdots 0}_{x_1 \text{ 0's}}}$$

Numerical coding of lists

$\mathit{list}\,\mathbb{N} \triangleq$ set of all finite lists of natural numbers, using ML notation for lists.

For $\ell \in \mathit{list}\,\mathbb{N}$, define $\ulcorner \ell \urcorner \in \mathbb{N}$ by induction on the length of the list ℓ:

$$\begin{cases} \ulcorner [] \urcorner & \triangleq \; \mathbf{0} \\ \ulcorner x :: \ell \urcorner & \triangleq \; \langle\!\langle x, \ulcorner \ell \urcorner \rangle\!\rangle = 2^x (2 \cdot \ulcorner \ell \urcorner + 1) \end{cases}$$

$$\boxed{0\mathrm{b}\ulcorner [x_1, x_2, \ldots, x_n] \urcorner} = \boxed{1 \mid 0 \cdots 0}\;\boxed{1 \mid 0 \cdots 0} \cdots \boxed{1 \mid 0 \cdots 0}$$

Hence $\ell \mapsto \ulcorner \ell \urcorner$ gives a bijection from $\mathit{list}\,\mathbb{N}$ to $\mathbb{N}$.

Numerical coding of programs

If P is the RM program $\boxed{\begin{array}{c} L_0 : body_0 \\ L_1 : body_1 \\ \vdots \\ L_n : body_n \end{array}}$

then its numerical code is

$$\ulcorner P \urcorner \triangleq \ulcorner [\ulcorner body_0 \urcorner, \ldots, \ulcorner body_n \urcorner] \urcorner$$

where the numerical code $\ulcorner body \urcorner$ of an instruction body is defined by:

$$\begin{cases} \ulcorner R_i^+ \to L_j \urcorner & \triangleq & \langle\!\langle 2i, j \rangle\!\rangle \\ \ulcorner R_i^- \to L_j, L_k \urcorner & \triangleq & \langle\!\langle 2i+1, \langle j, k \rangle \rangle\!\rangle \\ \ulcorner \mathrm{HALT} \urcorner & \triangleq & 0 \end{cases}$$

Any $x \in \mathbb{N}$ decodes to a unique instruction $body(x)$:

if $x = 0$ then $body(x)$ is $\mathtt{HALT}$,
else ($x > 0$ and) let $x = \langle\langle y, z \rangle\rangle$ in
 if $y = 2i$ is even, then
  $body(x)$ is $R_i^+ \to L_z$,
 else $y = 2i + 1$ is odd, let $z = \langle j, k \rangle$ in
  $body(x)$ is $R_i^- \to L_j, L_k$

So any $e \in \mathbb{N}$ decodes to a unique program $prog(e)$, called the register machine program with index e:

$$prog(e) \triangleq \boxed{\begin{array}{c} L_0 : body(x_0) \\ \vdots \\ L_n : body(x_n) \end{array}} \quad \text{where } e = \ulcorner [x_0, \dots, x_n] \urcorner$$

Example of $prog(e)$

- $786432 = 2^{19} + 2^{18} = 0b11\underbrace{0\ldots0}_{18\ ''0''s} = \ulcorner[18, 0]\urcorner$
- $18 = 0b10010 = \langle\langle 1, 4\rangle\rangle = \langle\langle 1, \langle 0, 2\rangle\rangle\rangle = \ulcorner R_0^- \to L_0, L_2\urcorner$
- $0 = \ulcorner \mathtt{HALT}\urcorner$

So $prog(786432) =$ $\boxed{\begin{array}{l} L_0 : R_0^- \to L_0, L_2 \\ L_1 : \mathtt{HALT} \end{array}}$

Example of $prog(e)$

- $786432 = 2^{19} + 2^{18} = 0b11\underbrace{0\ldots0}_{18\ "0"s} = \ulcorner[18,0]\urcorner$
- $18 = 0b10010 = \langle\!\langle 1,4 \rangle\!\rangle = \langle\!\langle 1, \langle 0,2 \rangle \rangle\!\rangle = \ulcorner R_0^- \to L_0, L_2 \urcorner$
- $0 = \ulcorner \mathtt{HALT} \urcorner$

So $prog(786432) =$

$$\boxed{\begin{array}{l} L_0 : R_0^- \to L_0, L_2 \\ L_1 : \mathtt{HALT} \end{array}}$$

N.B. jump to label with no body (erroneous halt)

What function is computed by a RM with program(786432) as its program?

Example of $prog(e)$

- $786432 = 2^{19} + 2^{18} = 0b11\underbrace{0\ldots0}_{18\ \text{"0"s}} = \ulcorner[18,0]\urcorner$
- $18 = 0b10010 = \langle\langle 1,4\rangle\rangle = \langle\langle 1,\langle 0,2\rangle\rangle\rangle = \ulcorner R_0^- \to L_0, L_2\urcorner$
- $0 = \ulcorner \mathtt{HALT}\urcorner$

So $prog(786432) =$

$L_0 : R_0^- \to L_0, L_2$
$L_1 : \mathtt{HALT}$

N.B. In case $e = 0$ we have $0 = \ulcorner[\,]\urcorner$, so $prog(0)$ is the program with an empty list of instructions, which by convention we regard as a RM that does nothing (i.e. that halts immediately).

$$666 = \text{0b}1010011010$$

$$= \ulcorner [1, 1, 0, 2, 1] \urcorner$$

$$\text{prog}(666) = \begin{array}{|l|} \hline L_0 : R_0^+ \to L_0 \\ L_1 : R_0^+ \to L_0 \\ L_2 : \text{HALT} \\ L_3 : R_0^- \to L_0, L_0 \\ L_4 : R_0^+ \to L_0 \\ \hline \end{array}$$

(never halts!)

What partial function does this compute?