d

Numerical Coding of Lists

Let $List\mathbb{N}$ be the set of all finite lists of natural numbers, defined by:
Slide 20

- empty list: $[]$
- list cons: $x :: \ell \in \mathrm{List}\,\mathbb{N}$ if $x \in \mathbb{N}$ and $\ell \in \mathrm{List}\,\mathbb{N}$

Notation: $[x_1, x_2, \ldots, x_n] \triangleq x_1 :: (x_2 :: (\cdots x_n :: [] \cdots))$

Numerical Coding of Lists

Let List $\mathbb{N}$ be the set of all finite lists of natural numbers.
Slide 21
For $\ell \in L$ List $\mathbb{N}$, define $\ulcorner \ell \urcorner \in \mathbb{N}$ by induction on the length of the list
$\ell : \begin{cases} \ulcorner [] \urcorner \triangleq 0 \\ \ulcorner x :: \ell \urcorner \triangleq \langle x, \ulcorner \ell \urcorner \rangle = 2^x(2 \cdot \ulcorner \ell \urcorner + 1) \end{cases}$
Thus, $\ulcorner [x_1, x_2, \ldots, x_n] \urcorner = \langle\langle x_1, \langle\langle x_2, \cdots \langle\langle x_n, 0 \rangle\rangle \cdots \rangle\rangle$

Numerical Coding of Lists

Let List $\mathbb{N}$ be the set of all finite lists of natural numbers.
For $\ell \in L$ List $\mathbb{N}$, define $\ulcorner \ell \urcorner \in \mathbb{N}$ by induction on the length of the list
$\ell : \begin{cases} \ulcorner [] \urcorner \triangleq 0 \\ \ulcorner x :: \ell \urcorner \triangleq \langle x, \ulcorner \ell \urcorner \rangle = 2^x(2 \cdot \ulcorner \ell \urcorner + 1) \end{cases}$
Examples
$\ulcorner [3] \urcorner = \ulcorner 3 :: [] \urcorner = \big\langle \langle 3, 0 \rangle = 2^3(2 \cdot 0 + 1) = 8$
$\ulcorner [1, 3] \urcorner = \langle\langle 1, \ulcorner [3] \urcorner \rangle = \langle\langle 1, 8 \rangle = 34$
$\ulcorner [2, 1, 3] \urcorner = \langle 2, \ulcorner [1, 3] \urcorner \rangle = \langle 2, 34 \rangle = 276$

Numerical Coding of Lists

Let List $\mathbb{N}$ be the set of all finite lists of natural numbers.
Slide 23
For $\ell \in \mathrm{List}\,\mathbb{N}$, define $\ulcorner \ell \urcorner \in \mathbb{N}$ by induction on the length of the list
$\ell : \begin{cases} \ulcorner [] \urcorner \triangleq 0 \\ \ulcorner x :: \ell \urcorner \triangleq \langle\langle x, \ulcorner \ell \urcorner \rangle = 2^x(2 \cdot \ulcorner \ell \urcorner + 1) \end{cases}$
Result The function $\ell \mapsto \ulcorner \ell \urcorner$ gives a bijection from List $\mathbb{N}$ to $\mathbb{N}$.

Numerical Coding of Lists

Let List $\mathbb{N}$ be the set of all finite lists of natural numbers.
For $\ell \in L$ List $\mathbb{N}$, define $\ulcorner \ell \urcorner \in \mathbb{N}$ by induction on the length of the list ℓ : $\begin{cases} \ulcorner [] \urcorner \triangleq 0 \\ \ulcorner x :: \ell \urcorner \triangleq \langle x, \ulcorner \ell \urcorner \rangle = 2^x(2 \cdot \ulcorner \ell \urcorner + 1) \end{cases}$
Result The function $\ell \mapsto \ulcorner \ell \urcorner$ gives a bijection from List $\mathbb{N}$ to $\mathbb{N}$.

Sketch Proof

The proof follows by observing that

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

To prove $0\mathrm{bb}\ulcorner [x_1, x_2, \ldots, x_n] \urcorner = \boxed{1} \boxed{0 \cdots 0} \boxed{1} \boxed{0 \cdots 0}$ use induction on the structure of $L = [x_1, \ldots, x_n]$.

Base Case This is trivial as $0\ \mathrm{b}\ulcorner [] \urcorner = 0$.

Inductive step Assume

$$\boxed{0\ \mathrm{b}\ulcorner [x_1, x_2, \ldots, x_k]}$$

$= \boxed{1} \boxed{0 \cdots 0} \boxed{1} \boxed{0 \cdots 0} \boxed{1}$

By the definitions, we have
$0\ \mathrm{b}\ulcorner [x, x_1, x_2, \ldots, x_k] \urcorner = 0\ \mathrm{b}\langle \langle x, \ulcorner [x_1, \ldots, x_k] \urcorner \rangle = 0\ \mathrm{b}\ulcorner [x_1, \ldots, x_n] \urcorner 1 \underbrace{0 \ldots 0}_{x 0s}$.

The induction hypothesis now gives the result. Using this result, $\ulcorner L \urcorner$ is clearly one-to-one and onto. To convince yourself of this, choose a few binary numbers n and give the corresponding list L_n such that $0\ \mathrm{b}\ulcorner L_n \urcorner = n$.

Slide 25

Recall Register Machines

Definition

A register machine (sometimes abbreviated to RM) is specified by:

- finitely many registers $R_0, R_1, \ldots, R_n$, each capable of storing a natural number;
- a program consisting of a finite list of instructions of the form label: body where, for $i = 0, 1, 2, \ldots$, the $(i+1)^{\text{th}}$ instruction has label L_i. The instruction body takes the form:

$R^+ \to L'$	add 1 to contents of register R and jump to instruction labelled L'
$R^- \to L', L''$	if contents of R is > 0, then subtract 1 and jump to L', else jump to L''
$HALT$	stop executing instructions

Slide 26

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:

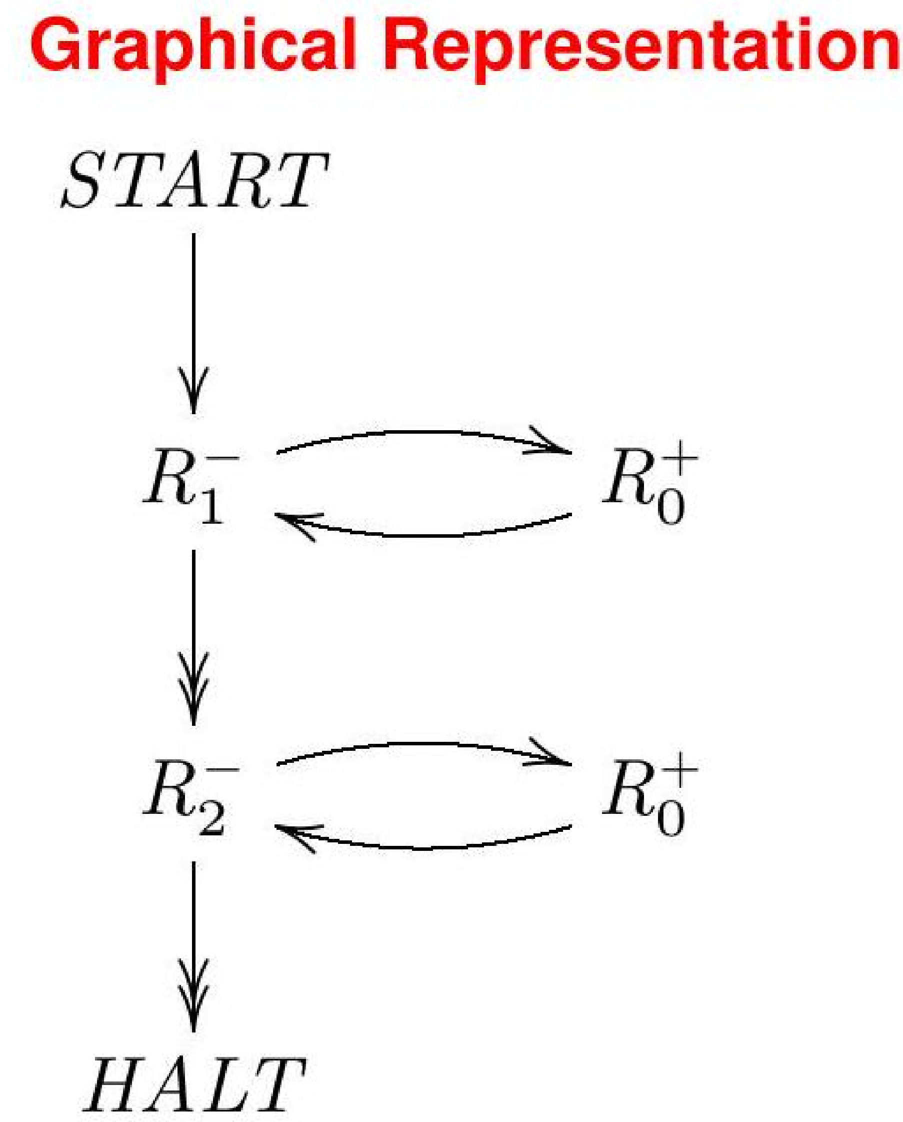
$$\begin{cases} \ulcorner R_i^+ \to L_j \urcorner & \triangleq \langle\!\langle 2i, j \rangle\!\rangle \\ \ulcorner R_i^- \to L_j, L_k \urcorner & \triangleq \langle\!\langle 2i+1, \langle j, k \rangle \rangle\!\rangle \\ \ulcorner HALT \urcorner & \triangleq 0 \end{cases}$$

Since $\langle -, - \rangle : \mathbb{N} \times \mathbb{N} \to \mathbb{N}^+$, $\langle -, - \rangle : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ and $\ulcorner - \urcorner : \text{List}\, \mathbb{N} \to \mathbb{N}$ are bijections, the functions $\ulcorner - \urcorner$ from bodies to natural numbers and $\ulcorner - \urcorner$ from RM programs to $\mathbb{N}$ are bijections.

Recall Addition $f(x, y) \triangleq x + y$ is Computable

Slide 27

Registers
$R_0 R_1 R_2$
Program
$L_0 : R_1^- \to L_1, L_2$
$L_1 : R_0^+ \to L_0$
$L_2 : R_2^- \to L_3, L_4$
$L_3 : R_0^+ \to L_2$
$L_4 : HALT$



If the machine starts with registers $(R_0, R_1, R_2) = (0, x, y)$, it halts with registers $(R_0, R_1, R_2) = (x + y, 0, 0)$.

Slide 28

Coding of the RM for Addition

$\ulcorner P \urcorner \triangleq \ulcorner [\ulcorner B_0 \urcorner, \ldots, \ulcorner B_4 \urcorner] \urcorner$ where

$$
\begin{aligned}
\ulcorner B_0 \urcorner &= \ulcorner R_1^- \to L_1, L_2 \urcorner = \langle\langle (2\times 1)+1, \langle 1,2\rangle\rangle \\
&= \langle\langle 3,9\rangle = 8\times(18+1) = 152 \\
\ulcorner B_1 \urcorner &= \ulcorner R_0^+ \to L_0 \urcorner = \langle 2\times 0, 0\rangle = 1 \\
\ulcorner B_2 \urcorner &= \ulcorner R_2^- \to L_3, L_4 \urcorner = \langle\langle (2\times 2)+1, \langle 3,4\rangle\rangle \\
&= \langle\langle 5, (8\times 9)-1\rangle\rangle = \langle\langle 5, 71\rangle \\
&= 2^5 \times ((2\times 71)+1) = 32\times 143 = 4576 \\
\ulcorner B_3 \urcorner &= \ulcorner R_0^+ \to L_2 \urcorner = \langle 2\times 0, 2\rangle = 5 \\
\ulcorner B_4 \urcorner &= \ulcorner HALT \urcorner = 0
\end{aligned}
$$

In the next section, we will introduce the Universal Register Machine. The Universal Register Machine carries out the following computation: starting with $R_0 = 0, R_1 = e$ (the code of the program), $R_2 = a$ (code of the list of arguments), and all other registers zeroed:

- decode e as a RM program P
- decode a as a list of register values $a_1, \ldots, a_n$
- carry out the computation of the RM program P starting with $R_0 = 0, R_1 = a_1, \ldots, R_n = a_n$ (and any other registers occurring in P set to 0).
  It is therefore important for you to understand what it means for a number $x \in \mathbb{N}$ to decode to a unique instruction body(x), and for a number $e \in \mathbb{N}$ to decode to a unique program prog(e).

Decoding Numbers as Bodies and Programs

Any $x \in \mathbb{N}$ decodes to a unique instruction bod $y(x)$:
if $x = 0$ then bod $y(x)$ is HALT, else ($x > 0$ and) let $x = \langle\langle y, z\rangle$ in if $y = 2i$ is even, then bod $y(x)$ is $R_i^+ \to L_z$,
Slide 29 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 :

$$
\operatorname{prog}(e) \triangleq \left|\begin{array}{c} L_0 : \text{body}\,(x_0) \\ \vdots \\ L_n : \text{body}\,(x_n) \end{array}\right] \text{ where } e = \ulcorner [x_0, \ldots, x_n] \urcorner
$$

Example of prog(e)

- $786432 = 2^{19} + 2^{18} = 0\text{ b}11\underbrace{0\ldots 0}_{18''0^{"}s} = \ulcorner [18, 0] \urcorner$

Slide 30

- $18 = 0\,\mathrm{b}10010 = \langle 1, 4 \rangle = \big\langle \langle 1, \langle 0, 2 \rangle \rangle \big\rangle = \ulcorner R_0^- \to L_0, L_2 \urcorner$
- $0 = \ulcorner HALT \urcorner$

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

Notice that, when $e = 0$, 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). Also, notice in slide 26 the jump to a label with no body (an erroneous halt). Again, choose some numbers and see what the register-machine programs they correspond to.