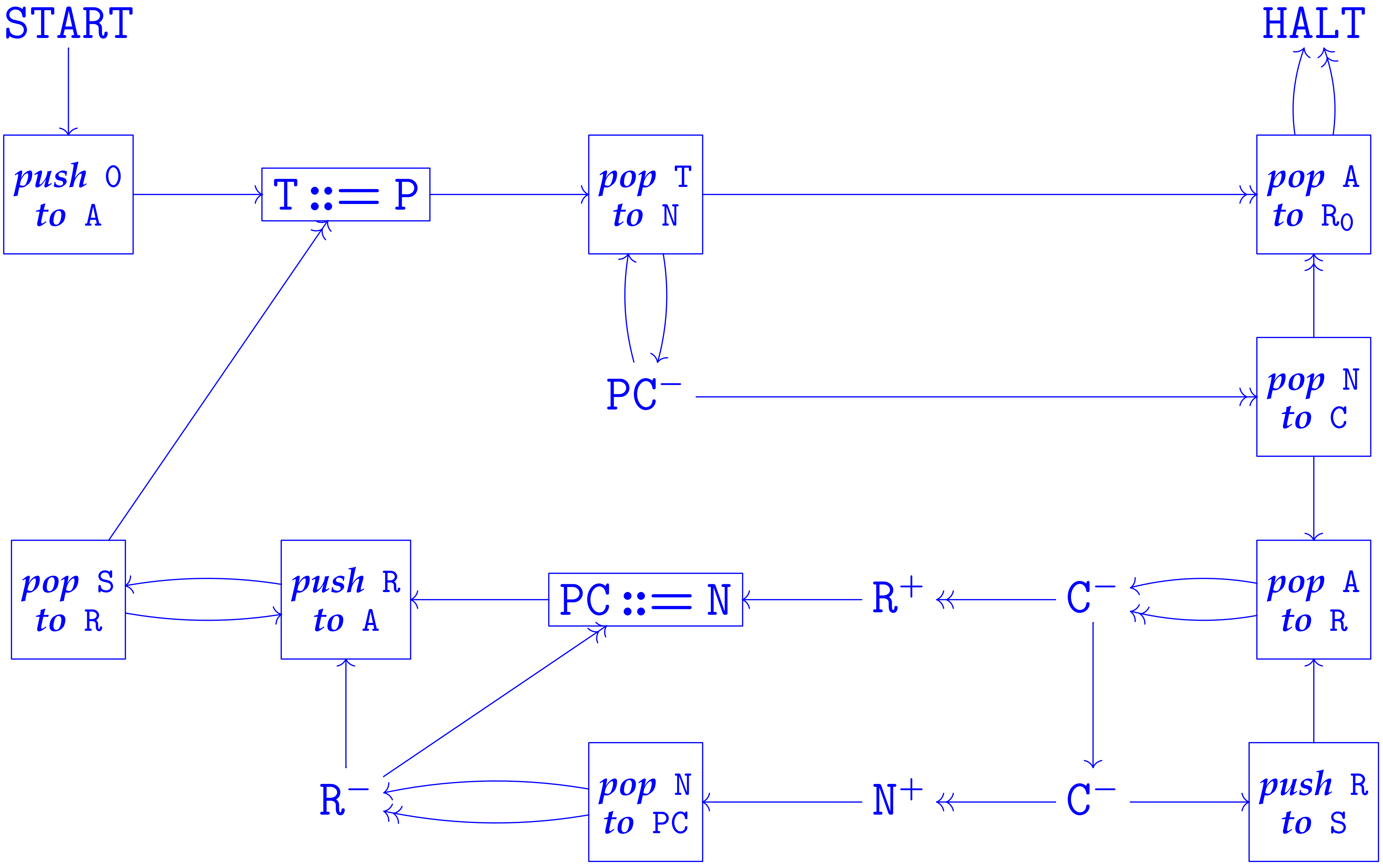
The program for $\boldsymbol{U}$



Overall structure of U's program

$\boxed{1}$ copy $\mathtt{PC}$th item of list in $\mathtt{P}$ to $\mathtt{N}$ (halting if $\mathtt{PC} >$ length of list); goto $\boxed{2}$

$\boxed{2}$ if $\mathtt{N} = 0$ then copy 0th item of list in $\mathtt{A}$ to $\mathtt{R}_0$ and halt, else (decode $\mathtt{N}$ as $\langle\langle y, z\rangle\rangle$; $\mathtt{C} ::= y$; $\mathtt{N} ::= z$; goto $\boxed{3}$)

{at this point either $\mathtt{C} = 2i$ is even and current instruction is $\mathtt{R}_i^+ \to \mathtt{L}_z$,
or $\mathtt{C} = 2i + 1$ is odd and current instruction is $\mathtt{R}_i^- \to \mathtt{L}_j, \mathtt{L}_k$ where $z = \langle j, k\rangle$}

$\boxed{3}$ copy ith item of list in $\mathtt{A}$ to $\mathtt{R}$; goto $\boxed{4}$

$\boxed{4}$ execute current instruction on $\mathtt{R}$; update $\mathtt{PC}$ to next label; restore register values to $\mathtt{A}$; goto $\boxed{1}$

Halting

For a finite computation $c_0, c_1, \ldots, c_m$, the last configuration $c_m = (\ell, r, \ldots)$ is a halting configuration, i.e. instruction labelled $\mathtt{L}_\ell$ is

<u>either</u> HALT (a "proper halt")

<u>or</u> $R^+ \to L$, or $R^- \to L, L'$ with $R > 0$, or $R^- \to L', L$ with $R = 0$ and there is no instruction labelled L in the program (an "erroneous halt")

E.g. $\boxed{\begin{array}{l} \mathtt{L}_0 : \mathtt{R}_0^+ \to \mathtt{L}_2 \\ \mathtt{L}_1 : \mathtt{HALT} \end{array}}$ halts erroneously. (has computation sequence $[(0,x)]$)

Halting

For a finite computation $c_0, c_1, \ldots, c_m$, the last configuration $c_m = (\ell, r, \ldots)$ is a halting configuration, i.e.

<u>either</u> $\ell \geq$ number of instructions in program (an "<u>erroneous halt</u>")

<u>or</u> ℓ^{th} instruction in program has body HALT (a "<u>proper halt</u>")

E.g. $\boxed{\begin{array}{l} L_0 : R_0^+ \to L_2 \\ L_1 : \mathrm{HALT} \end{array}}$ halts erroneously (has computation sequences $[(0, x), (2, x+1)]$)

The halting problem

Definition. A register machine H decides the Halting Problem if for all $e, a_1, \ldots, a_n \in \mathbb{N}$, starting H with

$$\mathtt{R}_0 = 0 \qquad \mathtt{R}_1 = e \qquad \mathtt{R}_2 = \ulcorner [a_1, \ldots, a_n] \urcorner$$

and all other registers zeroed, the computation of H always halts with $\mathtt{R}_0$ containing 0 or 1; moreover when the computation halts, $\mathtt{R}_0 = 1$ if and only if

> the register machine program with index e eventually halts when started with $\mathtt{R}_0 = 0, \mathtt{R}_1 = a_1, \ldots, \mathtt{R}_n = a_n$ and all other registers zeroed.

Definition. A register machine H decides the Halting Problem if for all $e, a_1, \ldots, a_n \in \mathbb{N}$, starting H with

$$\mathrm{R}_0 = 0 \qquad \mathrm{R}_1 = e \qquad \mathrm{R}_2 = \ulcorner [a_1, \ldots, a_n] \urcorner$$

and all other registers zeroed, the computation of H always halts with R_0 containing 0 or 1; moreover when the computation halts, $\mathrm{R}_0 = 1$ if and only if

> the register machine program with index e eventually halts when started with $\mathrm{R}_0 = 0, \mathrm{R}_1 = a_1, \ldots, \mathrm{R}_n = a_n$ and all other registers zeroed.

Theorem. No such register machine H can exist.

Proof of the theorem

Assume we have a RM H that decides the Halting Problem and derive a contradiction, as follows:

- Let H' be obtained from H by replacing $\mathtt{START}\!\longrightarrow$ by

  $$\mathtt{START}\longrightarrow\boxed{\mathtt{Z} ::= \mathtt{R}_1}\longrightarrow\boxed{\begin{array}{c}\textit{push } \mathtt{Z}\\ \textit{to } \mathtt{R}_2\end{array}}\longrightarrow$$

  (where $\mathtt{Z}$ is a register not mentioned in H's program).
- Let C be obtained from H' by replacing each $\mathtt{HALT}$ (& each erroneous halt) by

  $$\begin{tikzcd} {} \arrow[r] & \mathtt{R}_0^- \arrow[r, bend left] \arrow[d] & \mathtt{R}_0^+ \arrow[l, bend left] \\ & \mathtt{HALT} & \end{tikzcd} .$$

- Let $c \in \mathbb{N}$ be the index of C's program.

Proof of the theorem

Assume we have a RM H that decides the Halting Problem and derive a contradiction, as follows:

C started with $\mathtt{R}_1 = c$ eventually halts

if & only if

H' started with $\mathtt{R}_1 = c$ halts with $\mathtt{R}_0 = 0$

Proof of the theorem

Assume we have a RM H that decides the Halting Problem and derive a contradiction, as follows:

$$C \text{ started with } \mathtt{R}_1 = c \text{ eventually halts}$$

if & only if

$$H' \text{ started with } \mathtt{R}_1 = c \text{ halts with } \mathtt{R}_0 = 0$$

if & only if

$$H \text{ started with } \mathtt{R}_1 = c, \mathtt{R}_2 = \ulcorner [c] \urcorner \text{ halts with } \mathtt{R}_0 = 0$$

Proof of the theorem

Assume we have a RM H that decides the Halting Problem and derive a contradiction, as follows:

$$C \text{ started with } \mathtt{R}_1 = c \text{ eventually halts}$$

if & only if

$$H' \text{ started with } \mathtt{R}_1 = c \text{ halts with } \mathtt{R}_0 = 0$$

if & only if

$$H \text{ started with } \mathtt{R}_1 = c, \mathtt{R}_2 = \ulcorner [c] \urcorner \text{ halts with } \mathtt{R}_0 = 0$$

if & only if

$$prog(c) \text{ started with } \mathtt{R}_1 = c \text{ does not halt}$$

Proof of the theorem

Assume we have a RM H that decides the Halting Problem and derive a contradiction, as follows:

C started with $\mathrm{R}_1 = c$ eventually halts

if & only if

H' started with $\mathrm{R}_1 = c$ halts with $\mathrm{R}_0 = 0$

if & only if

H started with $\mathrm{R}_1 = c, \mathrm{R}_2 = \ulcorner [c] \urcorner$ halts with $\mathrm{R}_0 = 0$

if & only if

$prog(c)$ started with $\mathrm{R}_1 = c$ does not halt

if & only if

C started with $\mathrm{R}_1 = c$ does not halt

Proof of the theorem

Assume we have a RM H that decides the Halting Problem and derive a contradiction, as follows:

C started with $\mathtt{R}_1 = c$ eventually halts

if & only if

H' started with $\mathtt{R}_1 = c$ halts with $\mathtt{R}_0 = 0$

if & only if

H started with $\mathtt{R}_1 = c, \mathtt{R}_2 = \ulcorner [c] \urcorner$ halts with $\mathtt{R}_0 = 0$

if & only if

$prog(c)$ started with $\mathtt{R}_1 = c$ does not halt

if & only if

C started with $\mathtt{R}_1 = c$ does not halt

—contradiction!

Computable functions

Recall:

Definition. $f \in \mathbb{N}^n \rightharpoonup \mathbb{N}$ is (register machine) computable if there is a register machine M with at least $n+1$ registers R_0, R_1, ..., R_n (and maybe more) such that for all $(x_1, \ldots, x_n) \in \mathbb{N}^n$ and all $y \in \mathbb{N}$,

> the computation of M starting with $R_0 = 0$, $R_1 = x_1$, ..., $R_n = x_n$ and all other registers set to 0, halts with $R_0 = y$

if and only if $f(x_1, \ldots, x_n) = y$.

Note that the same RM M could be used to compute a unary function ($n = 1$), or a binary function ($n = 2$), etc. From now on we will concentrate on the unary case...

Enumerating computable functions

For each $e \in \mathbb{N}$, let $\varphi_e \in \mathbb{N} \rightharpoonup \mathbb{N}$ be the unary partial function computed by the RM with program $prog(e)$. So for all $x, y \in \mathbb{N}$:

> $\varphi_e(x) = y$ holds iff the computation of $prog(e)$ started with $R_0 = 0, R_1 = x$ and all other registers zeroed eventually halts with $R_0 = y$.

Thus

$$e \mapsto \varphi_e$$

defines an <u>onto</u> function from $\mathbb{N}$ to the collection of all computable partial functions from $\mathbb{N}$ to $\mathbb{N}$.

Enumerating computable functions

For each $e \in \mathbb{N}$, let $\varphi_e \in \mathbb{N} \rightharpoonup \mathbb{N}$ be the unary partial function computed by the RM with program $prog(e)$. So for all $x, y \in \mathbb{N}$:

> $\varphi_e(x) = y$ holds iff the computation of $prog(e)$ started with $R_0 = 0, R_1 = x$ and all other registers zeroed eventually halts with $R_0 = y$.

Thus

So this is <u>countable</u>

$$e \mapsto \varphi_e$$

defines an <u>onto</u> function from $\mathbb{N}$ to the collection of all computable partial functions from $\mathbb{N}$ to $\mathbb{N}$.

So $\mathbb{N} \rightharpoonup \mathbb{N}$ (uncountable, by Cantor) contains uncomputable functions

An uncomputable function

Let $f \in \mathbb{N} \rightharpoonup \mathbb{N}$ be the partial function with graph $\{(x, 0) \mid \varphi_x(x)\uparrow\}$.

Thus $f(x) \ = \begin{cases} 0 & \text{if } \varphi_x(x)\uparrow \\ \mathit{undefined} & \text{if } \varphi_x(x)\downarrow \end{cases}$

An uncomputable function

Let $f \in \mathbb{N} \rightharpoonup \mathbb{N}$ be the partial function with graph $\{(x, 0) \mid \varphi_x(x)\uparrow\}$.

Thus $f(x) = \begin{cases} 0 & \text{if } \varphi_x(x)\uparrow \\ \mathit{undefined} & \text{if } \varphi_x(x)\downarrow \end{cases}$

f is not computable, because if it were, then $f = \varphi_e$ for some $e \in \mathbb{N}$ and hence

- if $\varphi_e(e)\uparrow$, then $f(e) = 0$ (by def. of f); so $\varphi_e(e) = 0$ (since $f = \varphi_e$), hence $\varphi_e(e)\downarrow$
- if $\varphi_e(e)\downarrow$, then $f(e)\downarrow$ (since $f = \varphi_e$); so $\varphi_e(e)\uparrow$ (by def. of f)

—contradiction! So f cannot be computable.

(Un)decidable sets of numbers

Given a subset $S \subseteq \mathbb{N}$, its characteristic function $\chi_S \in \mathbb{N} \to \mathbb{N}$ is given by: $\chi_S(x) \triangleq \begin{cases} 1 & \text{if } x \in S \\ 0 & \text{if } x \notin S. \end{cases}$

(Un)decidable sets of numbers

Definition. $S \subseteq \mathbb{N}$ is called (register machine) decidable if its characteristic function $\chi_S \in \mathbb{N} \to \mathbb{N}$ is a register machine computable function. Otherwise it is called undecidable.

So S is decidable iff there is a RM M with the property: for all $x \in \mathbb{N}$, M started with $\mathrm{R}_0 = 0, \mathrm{R}_1 = x$ and all other registers zeroed eventually halts with R_0 containing 1 or 0; and $\mathrm{R}_0 = 1$ on halting iff $x \in S$.

(Un)decidable sets of numbers

Definition. $S \subseteq \mathbb{N}$ is called (register machine) decidable if its characteristic function $\chi_S \in \mathbb{N} \to \mathbb{N}$ is a register machine computable function. Otherwise it is called undecidable.

So S is decidable iff there is a RM M with the property: for all $x \in \mathbb{N}$, M started with $R_0 = 0, R_1 = x$ and all other registers zeroed eventually halts with R_0 containing 1 or 0; and $R_0 = 1$ on halting iff $x \in S$.

Basic strategy: to prove $S \subseteq \mathbb{N}$ undecidable, try to show that decidability of S would imply decidability of the Halting Problem.

For example. . .

Claim: $S_0 \triangleq \{e \mid \varphi_e(0){\downarrow}\}$ is undecidable.

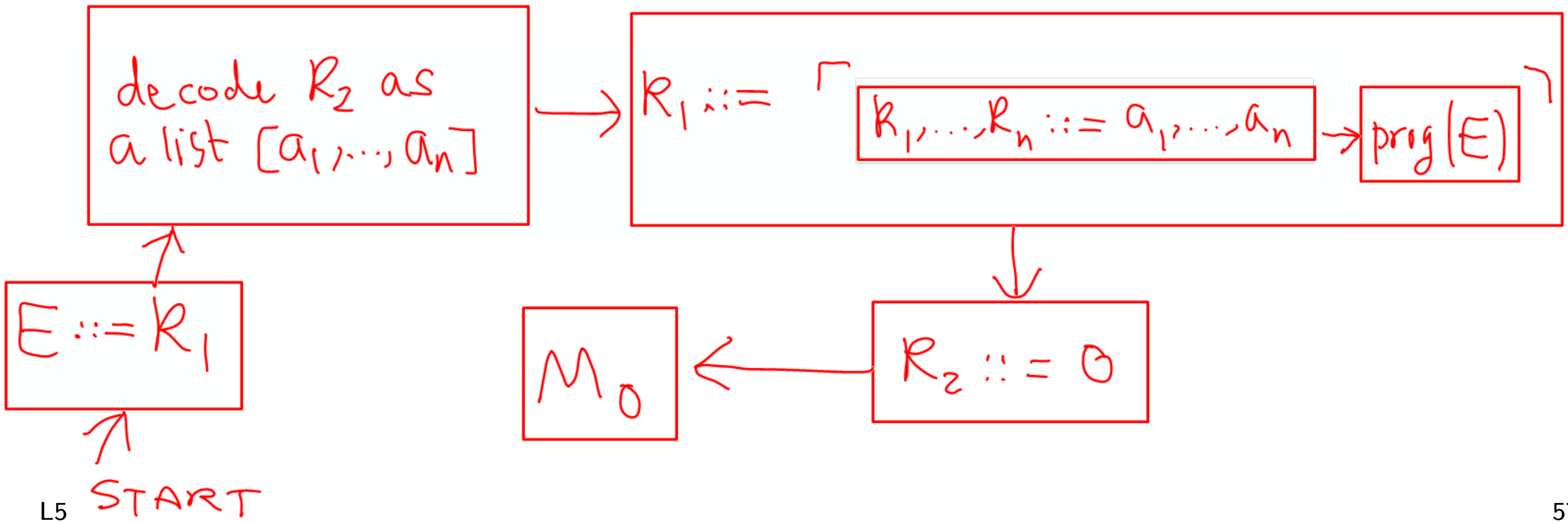
Proof (sketch): Suppose M_0 is a RM computing χ_{S_0}. From M_0's program (using the same techniques as for constructing a universal RM) we can construct a RM H to carry out:

$$
\begin{array}{l}
\mathit{let}\ e = \mathtt{R}_1\ \mathit{and}\ \ulcorner[a_1, \dots, a_n]\urcorner = \mathtt{R}_2\ \mathit{in} \\
\qquad \mathtt{R}_1 ::= \ulcorner(\mathtt{R}_1 ::= a_1)\,; \cdots ; (\mathtt{R}_n ::= a_n)\,; \mathit{prog}(e)\urcorner\,; \\
\mathtt{R}_2 ::= 0\,; \\
\mathit{run}\ M_0
\end{array}
$$

Claim: $S_0 \triangleq \{e \mid \varphi_e(0)\downarrow\}$ is undecidable.

Proof (sketch): Suppose M_0 is a RM computing χ_{S_0}. From M_0's program (using the same techniques as for constructing a universal RM) we can construct a RM H to carry out:

$$
\begin{array}{l}
\mathit{let}\ e = \mathtt{R}_1\ \mathit{and}\ \ulcorner [a_1, \ldots, a_n] \urcorner = \mathtt{R}_2\ \mathit{in} \\
\qquad \mathtt{R}_1 ::= \ulcorner (\mathtt{R}_1 ::= a_1) \,; \cdots ; (\mathtt{R}_n ::= a_n) \,; \mathit{prog}(e) \urcorner ; \\
\mathtt{R}_2 ::= 0 \,; \\
\mathit{run}\ M_0
\end{array}
$$



Claim: $S_0 \triangleq \{e \mid \varphi_e(0)\downarrow\}$ is undecidable.

Proof (sketch): Suppose M_0 is a RM computing χ_{S_0}. From M_0's program (using the same techniques as for constructing a universal RM) we can construct a RM H to carry out:

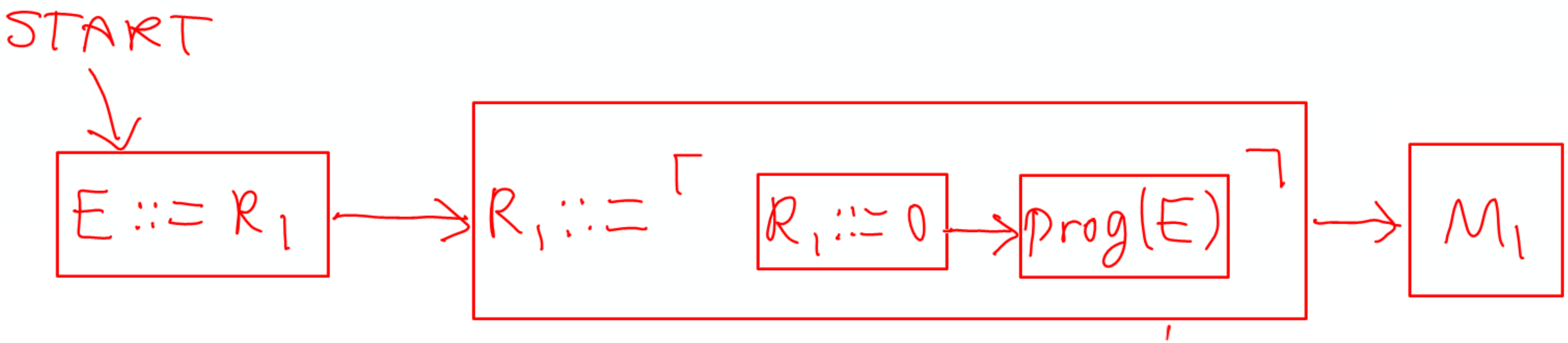
$$
\begin{array}{l}
\textit{let } e = \mathtt{R}_1 \textit{ and } \ulcorner[a_1,\ldots,a_n]\urcorner = \mathtt{R}_2 \textit{ in} \\
\qquad \mathtt{R}_1 ::= \ulcorner(\mathtt{R}_1 ::= a_1)\,;\cdots;(\mathtt{R}_n ::= a_n)\,;\textit{prog}(e)\urcorner; \\
\mathtt{R}_2 ::= 0\,; \\
\textit{run } M_0
\end{array}
$$

Then by assumption on M_0, H decides the Halting Problem—contradiction. So no such M_0 exists, i.e. χ_{S_0} is uncomputable, i.e. S_0 is undecidable.

Claim: $S_1 \triangleq \{e \mid \varphi_e \text{ a total function}\}$ is undecidable.

Proof (sketch): Suppose M_1 is a RM computing χ_{S_1}. From M_1's program we can construct a RM M_0 to carry out:

$$\begin{array}{l} \mathit{let}\ e = \mathtt{R}_1\ \mathit{in}\ \mathtt{R}_1 ::= \ulcorner \mathtt{R}_1 ::= 0\,;\mathit{prog}(e)\urcorner\,; \\ \mathit{run}\ M_1 \end{array}$$



Claim: $S_1 \triangleq \{e \mid \varphi_e \text{ a total function}\}$ is undecidable.

Proof (sketch): Suppose M_1 is a RM computing χ_{S_1}. From M_1's program we can construct a RM M_0 to carry out:

$$\begin{array}{l} \mathit{let}\ e = \mathtt{R}_1\ \mathit{in}\ \mathtt{R}_1 ::= \ulcorner \mathtt{R}_1 ::= 0\,;\mathit{prog}(e)\urcorner\,; \\ \mathit{run}\ M_1 \end{array}$$

Then by assumption on M_1, M_0 decides membership of S_0 from previous example (i.e. computes χ_{S_0})—contradiction. So no such M_1 exists, i.e. χ_{S_1} is uncomputable, i.e. S_1 is undecidable.

Exercise 5 If $f : \mathbb{N} \to \mathbb{N}$ is a RM computable function, $S_0 \subseteq \mathbb{N}$ & $S_1 \subseteq \mathbb{N}$ satisfy

$$\boxed{\forall e \in \mathbb{N}.\ e \in S_0 \Leftrightarrow f(e) \in S_1}$$

then if S_1 is decidable, then so is S_0

Exercise 5 If $f : \mathbb{N} \to \mathbb{N}$ is a RM computable function, $S_0 \subseteq \mathbb{N}$ & $S_1 \subseteq \mathbb{N}$ satisfy

$$\boxed{\forall e \in \mathbb{N}.\ e \in S_0 \Leftrightarrow f(e) \in S_1}$$

then if S_1 is decidable, then so is S_0

For S_1 & S_2 as on Slides 57 & 58 we have:

$$e \in S_0 \Leftrightarrow \varphi_e(0)\downarrow$$

$$f(e) \in S_1 \Leftrightarrow \forall x \in \mathbb{N}.\ \varphi_{f(e)}(x)\downarrow$$

Exercise 5 If $f : \mathbb{N} \to \mathbb{N}$ is a RM computable function, $S_0 \subseteq \mathbb{N}$ & $S_1 \subseteq \mathbb{N}$ satisfy

$$\boxed{\forall e \in \mathbb{N}.\ e \in S_0 \Leftrightarrow f(e) \in S_1}$$

then if S_1 is decidable, then so is S_0

For S_1 & S_2 as on Slides 57 & 58 we have:

$$e \in S_0 \Leftrightarrow \varphi_e(0)\downarrow$$

$$f(e) \in S_1 \Leftrightarrow \forall x \in \mathbb{N}.\ \varphi_{f(e)}(x)\downarrow$$

So can apply the Exercise to deduce undecidability of S_1 from undecidability of S_0 by finding RM computable $f : \mathbb{N} \to \mathbb{N}$ with

$$\forall e, x.\ \varphi_{f(e)}(x) \equiv \varphi_e(0)$$

Exercise 5 If $f : \mathbb{N} \to \mathbb{N}$ is a RM computable function, $S_0 \subseteq \mathbb{N}$ & $S_1 \subseteq \mathbb{N}$ satisfy

$$\boxed{\forall e \in \mathbb{N}.\ e \in S_0 \Leftrightarrow f(e) \in S_1}$$

then if S_1 is decidable, then so is S_0

For S_1 & S_2 as on Slides 57 & 58 we have:

$$e \in S_0 \Leftrightarrow \varphi_e(0)\downarrow$$

$$f(e) \in S_1 \Leftrightarrow \forall x \in \mathbb{N}.\ \varphi_{f(e)}(x)\downarrow$$

So can apply the Exercise to deduce <u>un</u>decidability of S_1 from <u>un</u>decidability of S_0 by finding RM computable $f : \mathbb{N} \to \mathbb{N}$ with

$$\forall e, x.\ \varphi_{f(e)}(x) \equiv \varphi_e(0)$$

"Kleene equivalence" (p 82): either LHS & RHS are undefined, or both are defined and equal