Universal register machine, $U$
High-level specification

Universal RM $U$ carries out the following computation, starting with $R_0 = 0$, $R_1 = e$ (code of a program), $R_2 = a$ (code of a 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).
Mnemonics for the registers of \( U \) and the role they play in its program:

\[
\begin{align*}
R_1 & \equiv P \quad \text{code of the RM to be simulated} \\
R_2 & \equiv A \quad \text{code of current register contents of simulated RM} \\
R_3 & \equiv PC \quad \text{program counter—number of the current instruction} \\
 & \quad \text{(counting from 0)} \\
R_4 & \equiv N \quad \text{code of the current instruction body} \\
R_5 & \equiv C \quad \text{type of the current instruction body} \\
R_6 & \equiv R \quad \text{current value of the register to be incremented or decremented by current instruction (if not \texttt{HALT})} \\
R_7 & \equiv S, \quad R_8 \equiv T \quad \text{and} \quad R_9 \equiv Z \quad \text{are auxiliary registers.}
\end{align*}
\]
Overall structure of U’s program

1. copy PCth item of list in P to N; goto 2

2. if \( N = 0 \) then copy 0th item of list in A to \( R_0 \) and halt, else (decode \( N \) as \( \langle y, z \rangle \); \( C := y \); \( N := z \); goto 3)

{at this point either \( C = 2i \) is even and current instruction is \( R_i^+ \rightarrow L_z \), or \( C = 2i + 1 \) is odd and current instruction is \( R_i^- \rightarrow L_j, L_k \) where \( z = \langle j, k \rangle \)}

3. copy ith item of list in A to R; goto 4

4. execute current instruction on R; update PC to next label; restore register values to A; goto 1

To implement this, we need RMs for manipulating (codes of) lists of numbers...
The program $\text{START} \rightarrow [S ::= R] \rightarrow \text{HALT}$

to copy the contents of $R$ to $S$ can be implemented by

\[
\begin{align*}
\text{START} &\rightarrow S^- \quad \rightarrow R^- \quad \rightarrow Z^- \quad \rightarrow \text{HALT} \\
&\quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \\
&\quad Z^+ \quad \quad \quad \quad R^+ \\
&\quad \quad \quad \quad \quad \downarrow \\
&\quad S^+ 
\end{align*}
\]
The program $\text{START} \rightarrow [S ::= R] \rightarrow \text{HALT}$

to copy the contents of $R$ to $S$ can be implemented by

\begin{align*}
S &:= 0 \\
S^- &\rightarrow S^+ \\
R^- &\rightarrow Z^- \\
Z^+ &\rightarrow R^+ \\
R^+ &\rightarrow \text{HALT}
\end{align*}
The program \( \text{START} \rightarrow [S ::= R] \rightarrow \text{HALT} \)

to copy the contents of \( R \) to \( S \) can be implemented by

\[
\begin{align*}
(R, s, z) & := (0, s + R, z + R) \\
S & := 0
\end{align*}
\]
The program \( \text{START} \rightarrow \boxed{S := R} \rightarrow \text{HALT} \)

to copy the contents of \( R \) to \( S \) can be implemented by

\[
\begin{align*}
S := 0 \\
(R, S, Z) := (0, S + R, Z + R) \\
(R, Z) := (R + Z, 0)
\end{align*}
\]
The program $\text{START} \rightarrow [S := R] \rightarrow \text{HALT}$ to copy the contents of $R$ to $S$ can be implemented by

$\text{START} \rightarrow S^- \rightarrow R^- \rightarrow Z^- \rightarrow \text{HALT}$

precondition:
- $R = x$
- $S = y$
- $Z = 0$

postcondition:
- $R = x$
- $S = x$
- $Z = 0$
The program \[ \text{START} \rightarrow \text{push } X \] to \( L \) \( \rightarrow \text{HALT} \]

\[ 2^X(2L + 1) \]

to carry out the assignment \( (X, L) ::= (0, X :: L) \) can be implemented by

\[ \begin{align*}
\text{START} & \rightarrow Z^+ \rightarrow L^- \rightarrow Z^- \rightarrow X^- \rightarrow \text{HALT} \\
Z^+ & \leftarrow \text{START} \rightleftharpoons L^+ \\
Z^- & \leftarrow \text{HALT} \\
L^- & \leftarrow X^- \\
\end{align*} \]
The program $\text{START} \rightarrow \begin{array}{c} \text{push} \\ X \\ \text{to} \\ L \end{array} \rightarrow \text{HALT}$

to carry out the assignment $(X, L) ::= (0, X :: L)$ can be implemented by

$\begin{array}{c} \text{START} \rightarrow Z^+ \rightarrow L^- \rightarrow Z^- \rightarrow X^- \rightarrow \text{HALT} \\ (L, Z) := (2L + 1 + Z, 0) \end{array}$
The program $\text{START} \rightarrow \text{push}_X \rightarrow \text{L} \rightarrow \text{HALT}$

to carry out the assignment $(X, L) ::= (0, X :: L)$ can be implemented by

$\text{START} \rightarrow Z^+ \rightarrow L^- \rightarrow Z^- \rightarrow X^- \rightarrow \text{HALT}$

$(L, Z) ::= (2L + Z, 0)$
The program \( \text{START} \rightarrow \text{push } X \rightarrow L \rightarrow \text{HALT} \)

to carry out the assignment \((X, L) ::= (0, X :: L)\) can be implemented by

\[
\begin{align*}
\text{START} & \rightarrow Z^+ \\
Z^+ & \rightarrow L^- \\
L^- & \rightarrow Z^- \\
Z^- & \rightarrow X^- \\
X^- & \rightarrow \text{HALT}
\end{align*}
\]

precondition:
\[
\begin{align*}
X &= x \\
L &= \ell \\
Z &= 0
\end{align*}
\]

postcondition:
\[
\begin{align*}
X &= 0 \\
L &= \langle x, \ell \rangle = 2^x (2\ell + 1) \\
Z &= 0
\end{align*}
\]
The program \( \text{START} \xrightarrow{\text{pop } L \text{ to } X} \text{HALT} \xrightarrow{} \text{EXIT} \) specified by

\[
\text{"if } L = 0 \text{ then } (X ::= 0; \text{ goto EXIT}) \text{ else let } L = \langle x, \ell \rangle \text{ in } (X ::= x; \ L ::= \ell; \text{ goto HALT})\]

can be implemented by
if $Z+L$ even then
$(Z,L) := (0, \frac{1}{2}(Z+L))$ & goto E

else
$(Z,L) := (0, \frac{1}{2}(Z+L-1))$ & goto 0

START

$X^-$

$X^+$

E

HALT

0

START

$X^-$

$X^+$

E

HALT

0

EXIT

$Z^+$

$L^+$

$Z^-$

$Z^-$

$L^-$

$L^-$

$L^+$

$X^-$
\{ assuming Z=0 \& L>0 \} \\
(While L even do \( L := \frac{1}{2} L \); \( X := X+1 \)); \\
\( L := \frac{1}{2} (L-1) \)
The program \textbf{START} $\rightarrow \begin{array}{c} \text{pop } L \\ \text{to } X \end{array} \rightarrow \text{HALT}$ specified by

"if $L = 0$ then ($X ::= 0; \text{goto EXIT}$) else
let $L = \langle x, \ell \rangle$ in ($X ::= x; L ::= \ell; \text{goto HALT}$)"

can be implemented by
Overall structure of $U$'s program

1. copy $PC$th item of list in $P$ to $N$; goto 2

2. if $N = 0$ then copy 0th item of list in $A$ to $R_0$ and halt, else (decode $N$ as $\langle y, z \rangle$; $C := y$; $N := z$; goto 3)

\{ at this point either $C = 2i$ is even and current instruction is $R_i^+ \rightarrow L_z$, or $C = 2i + 1$ is odd and current instruction is $R_i^- \rightarrow L_j, L_k$ where $z = \langle j, k \rangle$ \}

3. copy $i$th item of list in $A$ to $R$; goto 4

4. execute current instruction on $R$; update $PC$ to next label; restore register values to $A$; goto 1
The program for $U$

START

push 0 to A

T ::= P

pop T to N

PC

pop A to R₀

HALT

pop A to R

pop N to C

pop A to R

pop S to R

pop N to PC

R

push R to A

PC ::= N

R

push R to A

N

push R to S

C
The program for $U$

```
The program for $U$

\[ \text{START} \]

\[ \text{push 0 to } A \]

\[ T ::= P \]

\[ \text{pop to } N \]

\[ PC^- \]

\[ T = 0 \]

\[ \text{HALT} \]

\[ \text{pop A to } R_0 \]

\[ \text{pop N to } C \]

\[ \text{pop A to } R \]

\[ \text{push R to } S \]

\[ \text{push R to } PC \]

\[ \text{pop N to } R \]

\[ \text{PC ::= N} \]

\[ \text{R}^+ \]

\[ \text{C}^- \]

\[ \text{N}^+ \]

\[ \text{C}^- \]

\[ \text{push R to } S \]
```
The program for $U$

START

$push \ 0$ to $A$

$T ::= P$

$pop$ to $T$

$pop$ to $N$

$T = 0$

$pop$ to $A$

$push$ to $R$

$R -$

$push$ to $R$

$R +$

$PC ::= N$

$R -$

$pop$ to $PC$

$N +$

$pop$ to $N$

$N -$

$pop$ to $S$

HALT

$pop$ to $R_0$

$C -$

$pop$ to $C$

$push$ to $R$

$S -$

$push$ to $S$
The program for $U$

1. **START**
   - $push \ 0$ to $A$
   - $T ::= P$
   - $pop \ T$ to $N$
   - $PC^{-}$

2. **HALT**
   - $T = 0$
   - $pop \ A$ to $R_0$
   - $pop \ N$ to $C$
   - $pop \ A$ to $R$
   - $push \ R$ to $S$

3. **PC**
   - $PC ::= N$
   - $R^+$
   - $C^-$
   - $pop \ N$ to $PC$
   - $N^+$
   - $C^-$
   - $push \ R$

- $C_{even}$
- $C_{odd}$
The program for $U$

START

1

$push \ 0 \ to \ A$

$T ::= P$

$pop \ T \ to \ N$

PC$

$pop \ N \ to \ C$

$pop \ S \ to \ R$

$push \ R \ to \ A$

$PC ::= N$

$R^+$

C even

$R^-$

C odd

$pop \ N \ to \ PC$

$pop \ R \ to \ A$

$N^+$

$S$

HALT

2

$pop \ A \ to \ R_0$

$pop \ N \ to \ C$

$pop \ A \ to \ R$

$push \ R \ to \ S$

$T = 0$