# Fundamental Computer Science

Denis Trystram and Malin Rau
(inspired by Giorgio Lucarelli)

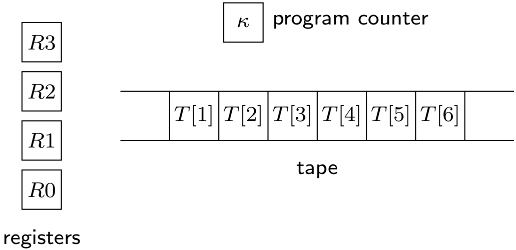
February, 2020

# Random Access Turing Machines

- Random Access Memory
  - access any position of the tape in a single step

# Random Access Turing Machines

- Random Access Memory
  - access any position of the tape in a single step
- we also need:
  - finite number of *registers* $\rightarrow$ manipulate addresses of the tape
  - *program counter* $\rightarrow$ current instruction to execute




- program: a set of instructions

# Random Access Turing Machines: Instructions set

| **instruction** | **operand** | **semantics** |
|---|---|---|
| read | $j$ | $R_0 \leftarrow T[R_j]$ |
| write | $j$ | $T[R_j] \leftarrow R_0$ |
| store | $j$ | $R_j \leftarrow R_0$ |
| load | $j$ | $R_0 \leftarrow R_j$ |
| load | $= c$ | $R_0 = c$ |
| add | $j$ | $R_0 \leftarrow R_0 + R_j$ |
| add | $= c$ | $R_0 \leftarrow R_0 + c$ |
| sub | $j$ | $R_0 \leftarrow \max\{R_0 + R_j, 0\}$ |
| sub | $= c$ | $R_0 \leftarrow \max\{R_0 + c, 0\}$ |
| half | | $R_0 \leftarrow \lfloor \frac{R_0}{2} \rfloor$ |
| jump | $s$ | $\kappa \leftarrow s$ |
| jpos | $s$ | if $R_0 > 0$ then $\kappa \leftarrow s$ |
| jzero | $s$ | if $R_0 = 0$ then $\kappa \leftarrow s$ |
| halt | | $\kappa = 0$ |

- register $R_0$: *accumulator*

# Random Access Turing Machines: Formal definition

A Random Access Turing Machine is a pair $M = (k, \Pi)$, where

- $k > 0$ is the finite number of registers, and
- $\Pi = (\pi_1, \pi_2, \ldots, \pi_p)$ is a finite sequence of instructions (program).

# Random Access Turing Machines: Formal definition

A Random Access Turing Machine is a pair $M = (k, \Pi)$, where

- $k > 0$ is the finite number of registers, and
- $\Pi = (\pi_1, \pi_2, \ldots, \pi_p)$ is a finite sequence of instructions (program).

## Notations

- the last instruction $\pi_p$ is always a *halt* instruction
- $(\kappa; R_0, R_1, \ldots, R_{k-1}; T)$: a **configuration**, where
  - $\kappa$: program counter
  - $R_j$, $0 \leq j < k$: the current value of register $j$
  - $T$: the contents of the tape
    (each $T[j]$ contains a non-negative integer, i.e. $T[j] \in \mathbb{N}$)
- **halted configuration:** $\kappa = 0$

# Examples

```
1: load 1
2: add 2
3: sub =1
4: store 1
5: halt
```

$$(1; 0, 5, 3; \emptyset)$$

# Examples

1: load 1
2: add 2
3: sub =1
4: store 1
5: halt

$$\begin{aligned}(1;0,5,3;\emptyset) &\vdash (2;5,5,3;\emptyset) \vdash (3;8,5,3;\emptyset) \vdash (4;7,5,3;\emptyset)\\ &\vdash (5;7,7,3;\emptyset) \vdash (0;7,7,3;\emptyset)\end{aligned}$$

# Examples

1: load 1
2: add 2
3: sub =1
4: store 1
5: halt

$$
\begin{aligned}
(1;0,5,3;\emptyset) \ &\vdash \ (2;5,5,3;\emptyset) \vdash (3;8,5,3;\emptyset) \vdash (4;7,5,3;\emptyset) \\
&\vdash \ (5;7,7,3;\emptyset) \vdash (0;7,7,3;\emptyset)
\end{aligned}
$$

$$R_1 \leftarrow R_2 + R_1 - 1$$

# Examples

1: load 1
2: add 2
3: sub =1
4: store 1
5: halt

$$(1;0,5,3;\emptyset) \vdash (2;5,5,3;\emptyset) \vdash (3;8,5,3;\emptyset) \vdash (4;7,5,3;\emptyset)$$
$$\vdash (5;7,7,3;\emptyset) \vdash (0;7,7,3;\emptyset)$$

$$R_1 \leftarrow R_2 + R_1 - 1$$

1: load 1
2: jzero 6
3: sub =3
4: store 1
5: jump 2
6: halt

$$(1;0,7;\emptyset)$$

# Examples

1: load 1
2: add 2
3: sub =1
4: store 1
5: halt

$$(1;0,5,3;\emptyset) \vdash (2;5,5,3;\emptyset) \vdash (3;8,5,3;\emptyset) \vdash (4;7,5,3;\emptyset)$$
$$\vdash (5;7,7,3;\emptyset) \vdash (0;7,7,3;\emptyset)$$

$$R_1 \leftarrow R_2 + R_1 - 1$$

1: load 1
2: jzero 6
3: sub =3
4: store 1
5: jump 2
6: halt

$$(1;0,7;\emptyset) \vdash (2;7,7;\emptyset) \vdash (3;7,7;\emptyset) \vdash (4;4,7;\emptyset) \vdash (5;4,4;\emptyset)$$
$$\vdash (2;4,4;\emptyset) \vdash (3;4,4;\emptyset) \vdash (4;1,4;\emptyset) \vdash (5;1,1;\emptyset)$$
$$\vdash (2;1,1;\emptyset) \vdash (3;1,1;\emptyset) \vdash (4;0,1;\emptyset) \vdash (5;0,0;\emptyset)$$
$$\vdash (2;0,0;\emptyset) \vdash (6;0,0;\emptyset) \vdash (0;0,0;\emptyset)$$

# Examples

1: load 1
2: add 2
3: sub =1
4: store 1
5: halt

$$(1;0,5,3;\emptyset) \vdash (2;5,5,3;\emptyset) \vdash (3;8,5,3;\emptyset) \vdash (4;7,5,3;\emptyset)$$
$$\vdash (5;7,7,3;\emptyset) \vdash (0;7,7,3;\emptyset)$$

$$R_1 \leftarrow R_2 + R_1 - 1$$

1: load 1
2: jzero 6
3: sub =3
4: store 1
5: jump 2
6: halt

$$(1;0,7;\emptyset) \vdash (2;7,7;\emptyset) \vdash (3;7,7;\emptyset) \vdash (4;4,7;\emptyset) \vdash (5;4,4;\emptyset)$$
$$\vdash (2;4,4;\emptyset) \vdash (3;4,4;\emptyset) \vdash (4;1,4;\emptyset) \vdash (5;1,1;\emptyset)$$
$$\vdash (2;1,1;\emptyset) \vdash (3;1,1;\emptyset) \vdash (4;0,1;\emptyset) \vdash (5;0,0;\emptyset)$$
$$\vdash (2;0,0;\emptyset) \vdash (6;0,0;\emptyset) \vdash (0;0,0;\emptyset)$$

while $R_1 > 0$ do $R_1 \leftarrow R_1 - 3$

# Exercise

- Write a program for a Random Access Turing Machine that multiplies two integers.
  Tip: assume that the initial configuration is $(1; 0, a_1, a_2, 0; \emptyset)$