Universal Turing machines

So far: Each Turing machine does just one thing.

Compare: A digital computer can carry out any program (expressed in its language), i.e., it is a stored program device.

Question: Can a TM (as a model) act as a stored program machine?

Answer: Yes!

We'll construct a TM M_u that takes:

1. description of any TM M,
2. any input x to M,

then *simulates* M on x.

Example of a *universal Turing machine*.

Problem: M_u must have a fixed input and tape alphabet.

But: Arbitrary TM can have *any* finite alphabet.

Solution: We restrict M to be binary, i.e.,

$$\Sigma = \{ \mathtt{0}, \mathtt{1} \}, \qquad \Gamma = \{ \mathtt{0}, \mathtt{1}, \bar{\mathtt{b}} \}$$

only!

Also no final state, acceptance is assumed if M gets stuck in a certain (agreed) state.

Simplifies matters a lot:

- not a restriction,
- can convert any TM to this form.

Running example: A TM that computes

$$f(n) = \begin{cases} 2n, & \text{if } n \text{ is odd;} \\ 2n+1, & \text{if } n \text{ is even.} \end{cases}$$

$$\begin{aligned} Q &= \{\, q_0, q_1, q_2, q_3 \,\}, \\ I &= q_0, \\ D &= \{(q_0, 0, q_1, 0, R), (q_0, 1, q_2, 1, R), (q_0, \bar{b}, q_3, \bar{b}, R), \\ &\quad\; (q_1, 0, q_1, 0, R), (q_1, 1, q_2, 1, R), (q_1, \bar{b}, q_3, 1, L), \\ &\quad\; (q_2, 0, q_1, 0, R), (q_2, 1, q_2, 1, R), (q_2, \bar{b}, q_3, 0, L)\} \end{aligned}$$

Encoding Turing machines

$M = (Q, q_I, \delta)$ encoded as a string over $\{\,\mathtt{0}, \mathtt{1}, \mathtt{B}, \mathtt{*}\,\}$.

(a) States of M encoded as elements of $\{\mathtt{0}, \mathtt{1}\}^k$, e.g., $k = \lceil \lg |Q| \rceil$.

- For simplicity q_I receives code $\mathtt{0}^k$,
- otherwise assignment of codes to states is arbitrary.

Running example: $k = 2$,

$$\begin{array}{cccc} q_0 & q_1 & q_2 & q_3 \\ \downarrow & \downarrow & \downarrow & \downarrow \\ 00 & 01 & 10 & 11 \end{array}$$

(b) Tape symbols encoded as:

$$\begin{array}{ccc} \mathtt{0} & \mathtt{1} & \not{b} \\ \downarrow & \downarrow & \downarrow \\ \mathtt{0} & \mathtt{1} & \mathtt{B} \end{array}$$

(Don't want to confuse $\not{b}$ of M_u with that of machine being simulated.)

(c) Directions encoded as:

$$\begin{array}{cc} L & R \\ \downarrow & \downarrow \\ \mathtt{0} & \mathtt{1} \end{array}$$

(d) Quintuples encoded as:

$$\begin{array}{ccccccc} (& q, & s, & q', & s', & d &) \\ & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \\ & \langle q \rangle & \langle s \rangle & \langle q' \rangle & \langle s' \rangle & \langle d \rangle & \end{array}$$

Length is $2k+3$ and $\langle x \rangle$ denotes code of x.

Running example: $\langle (q_0, 0, q_1, 0, R) \rangle = 00\,0\,01\,0\,1$.

Encode δ as

$$\langle M \rangle = \langle t_0 \rangle * \langle t_1 \rangle * \cdots * \langle t_{m-1} \rangle$$

where $t_1, t_1, \ldots, t_{m-1}$ are the quintuples (in some order).

Running example:

$$\begin{aligned}\langle M \rangle = & \; 0000101 * 0011011 * 00B11B1 * 0100101 * \\ & \; 0111011 * 01B1110 * 1000101 * 1011011 * \\ & \; 10B1100.\end{aligned}$$

Universal Turing machine M_u: input alphabet is

$$\{\,\texttt{0}, \texttt{1}, \texttt{B}, \texttt{*}, \texttt{\$}, \texttt{\^{}}\,\}.$$

To simulate M on input x start M_u with

$$\texttt{\$0}^{k+1}\texttt{*}\langle M\rangle\texttt{\$\^{}}\,x.$$

Shorter running example:

$$\langle M\rangle = \texttt{0001011*0010111*01B1010*1000111},$$
$$x = 0010.$$

Start M_u with

$$\texttt{\$000*}\langle\texttt{M}\rangle\texttt{\$\^{}0010}$$

Each step of simulated machine, M, involves M_u in one *cycle*.

Each cycle has six *phases*.

Start:

```
$000*0001011*0010111*01B1010*1000111$^0010
```

Just before second cycle, i.e., just after simulating `00 0 10 1 1`, tape of M_u is:

```
$101*0001011*0010111*01B1010*1000111$1^010
```

(i) Read scanned symbol and copy just before first `*`

```
$100*0001011*0010111*01B1010*1000111$1^010
```

(ii) Locate quintuple: stuff between first `$` and `*` is $\langle q\rangle\langle s\rangle$. Use `0` $\to X$, `1` $\to Y$, `B` $\to Z$ as markers

```
$100*XXXYXYY*XXYXYYY*XYZYXYX*YXX0111$1^010
```

(iii) Fetch new state and symbol

```
$XYY*XXXYXYY*XXYXYYY*XYZYXYX*YXXXYYY$1^010
```

(iv) Print the new symbol

```
$XYY*XXXYXYY*XXYXYYY*XYZYXYX*YXXXYYY$1^110
```

(v) Move tape head

```
$XYY*XXXYXYY*XXYXYYY*XYZYXYX*YXXXYYY$11^10
```

(vi) Tidy tape X $\longrightarrow$ 0, Y $\longrightarrow$ 1, Z $\longrightarrow$ B, ready for next cycle

```
$011*0001011*0010111*01B1010*1000111$11^10
```