COSE215: Theory of Computation

Lecture 15 — Turing Machines

Hakjoo Oh
2018 Spring

Turing Machine

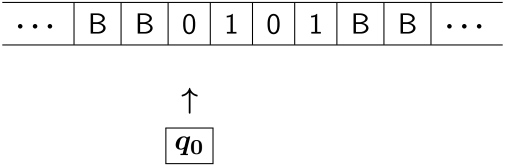
A *minimal* yet *complete* model for digital computers.

- "minimal": with further restriction, no more as powerful as computers
- "complete": every algorithm has a Turing machine

Informal Overview of Turing Machines

A Turing Machine (TM) is a finite automaton with a tape. Three parts:

- a control unit (i.e., finite automaton)
- a tape
- a tape head



Informal Overview of Turing Machines

The Turing Machine moves based on

- the state of the control unit,
- the tape symbol, and
- the transition function.

For instance, the following transition

$$\delta(q_0, 0) = (q_1, 1, R)$$

means that

- Change the state from q_0 to q_1.
- Write 1 to the current tape cell.
- Move the tape head to the right.

Formal Definition of Turing Machines

Definition

A Turing machine M is defined by

$$M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$$

- Q: The finite set of internal *states*.
- Σ: The finite set of *input symbols*. ($\Sigma \subseteq \Gamma - \{B\}$)
- Γ: The finite set of *tape symbols*.
- δ: The transition function.
- $q_0 \in Q$: The *initial state*.
- $B \in \Gamma$: The *blank* symbol. Assume $B \notin \Sigma$.
- $F \subseteq Q$: The set of final states.

Notes on Transition Function

- The type of δ:

$$\delta \in Q \times \Gamma \to Q \times \Gamma \times \{R, L\}$$

- δ is a partial function.
- Assume that δ is undefined for final states.

Example 1

$$M_1 = (\{q_0, q_1\}, \{a, b\}, \{a, b, B\}, \delta, q_0, B, \{q_1\})$$

$$\begin{aligned}
\delta(q_0, a) &= (q_0, b, R) \\
\delta(q_0, b) &= (q_0, b, R) \\
\delta(q_0, B) &= (q_1, B, L)
\end{aligned}$$

Example 1

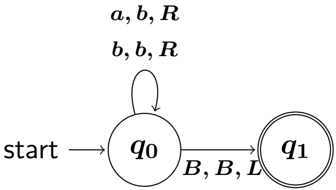
$$M_1 = (\{q_0, q_1\}, \{a, b\}, \{a, b, B\}, \delta, q_0, B, \{q_1\})$$

$$\begin{array}{rcl} \delta(q_0, a) & = & (q_0, b, R) \\ \delta(q_0, b) & = & (q_0, b, R) \\ \delta(q_0, B) & = & (q_1, B, L) \end{array}$$

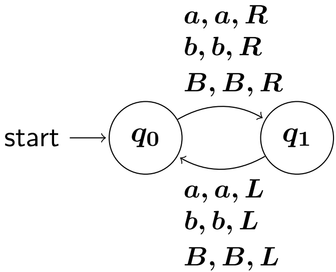
cf) compare with the same algorithm in C:

```
void f(char *str) {
  for (i = 0; i < strlen(str); i++)
    if (str[i] == 'a') str[i] = 'b';
}
```

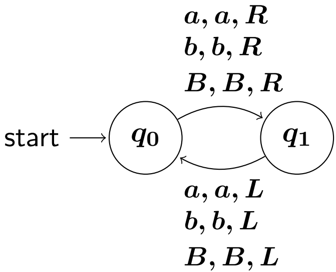
Transition Graph



Example 2



Example 2



cf)

```
void f(char *str) {
  while (1) ;
}
```

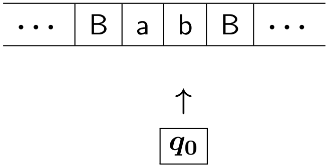
Instantaneous Description for TMs

An instantaneous description for a TM:

$$X_1 X_2 \cdots X_{i-1} q X_i X_{i+1} \cdots X_n$$

- $X_1 X_2 \cdots X_n$: the contents of tape (non-blanks only)
- q: the state
- The tape head is on X_i

E.g.,



Moves of TMs

- $\vdash$: one-step move
- $\vdash^*$: zero or more moves

E.g.,

$$abq_1cd \vdash abeq_2d$$

if

$$\delta(q_1, c) = (q_2, e, R)$$

Formal Definition of Moves

Definition

Let $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$ be a Turing machine. Then, any string $X_1 \cdots X_{i-1} q X_i \cdots X_n$ is an ID.

- Suppose $\delta(q, X_i) = (p, Y, L)$. Then

$$X_1 \cdots X_{i-1} q X_i \cdots X_n \vdash X_1 \cdots X_{i-2} p X_{i-1} Y X_{i+1} \cdots X_n$$

- Suppose $\delta(q, X_i) = (p, Y, R)$. Then

$$X_1 \cdots X_{i-1} q X_i \cdots X_n \vdash X_1 \cdots X_{i-1} Y p X_{i+1} \cdots X_n$$

M is said to halt from some initial configuration $X_1 \cdots X_{i-1} q X_i \cdots X_n$ if

$$X_1 \cdots X_{i-1} q X_i \cdots X_n \vdash^* Y_1 \cdots Y_{j-1} q Y_j \cdots X_m$$

and $\delta(q, Y_j)$ is undefined.

The Language of Turing Machines

Definition

Let $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$ be a Turing machine. Then the language accepted by M is

$$L(M) = \{w \in \Sigma^* \mid q_0 w \vdash^* x_1 q_f x_2 \text{ for some } q_f \in F, x_1, x_2 \in \Gamma^*\}$$

- The set of languages that can be accepted by some Turing machine is called *recursively enumerable*.

Turing Machines as Computing Machines

For a function

$$w' = f(w),$$

we can design a Turing machine that works as follows:

$$q_0 w \vdash^* q_f w'$$

for some final state q_f.

Definition

A function $f : D \to D$ is said to be Turing-computable or just computable if there exists some Turing machine $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$ such that

$$q_0 w \vdash^* q_f f(w)$$

for some $q_f \in F$ and for all $w \in D$.

Examples

- Design a Turing machine that accepts $L = \{a^n b^n \mid n \geq 1\}$.
- Given two positive integers x and y, design a Turing machine that computes $x + y$.
- Design a Turing machine that copies stirngs of 1's; find a machine that transforms w into ww.
- Design a Turing machine that computes $f(m, n)$:

$$f(m, n) = max(m - n, 0) = \text{if } m \geq n \text{ then } m - n \text{ else } 0$$