

# Chap. 8 Introduction to Turing Machine

## 8.2 The Turing Machine

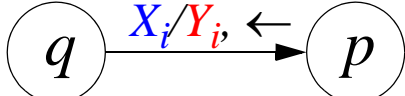
A Turing Machine(TM)  $M$  is a 7-tuples,  $M = (Q, T, \Gamma, \delta, q_0, B, F)$  where

1.  $Q$ : finite set of states,
2.  $T$ : a finite set of input symbols,
3.  $\Gamma$ : a finite set of tape symbols, ( $T \subseteq \Gamma$ )
4.  $\delta: Q \times \Gamma \rightarrow 2^{Q \times \Gamma \times \{L, R\}}$  or  $\delta \subseteq (Q \times \Gamma) \times (Q \times \Gamma \times \{L, R\})$ .  
 $(p, Y, D) \in \delta(q, X)$  where  $q, p \in Q$ ;  $X, Y \in \Gamma$ ;  $D \in \{L, R\}$   
 in the state  $q$  with the tape symbol  $X$ ,  
 go to the state  $p$ , the tape symbol is replaced to  $Y$ ,  
 tape head moves to left(L) or right(R).
5.  $q_0 \in Q$ , initial state,
6.  $B$ : blank symbol,  $B \in \Gamma$ ,  $B \notin T$ ,
7.  $F$ : a set of final or accepting states,  $F \subseteq Q$ .

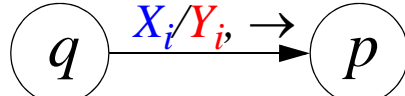
### *Instantaneous Description for Turing Machine*

$$(X_1X_2\dots X_{i-1}, q, X_iX_{i+1}\dots X_n) \in \Gamma^* \times Q \times \Gamma^*. (1 \leq i \leq n: X_i \in \Gamma)$$

- 1) Current state is  $q$ ,
- 2) The tape head is scanning the  $i$ -th symbol from left ( $X_i \in \Gamma$ ), and
- 3) Tape string,  $X_1\dots X_n \in \Gamma^*$ , is surrounded by infinite blanks.

If  $(p, Y_i, L) \in \delta(q, X_i)$  or  $q \xrightarrow{X_i/Y_i} \leftarrow p$ ,  then

$$(X_1X_2\dots X_{i-1}, q, X_iX_{i+1}\dots X_n) \Rightarrow_M \overset{X_i/Y_i}{\leftarrow} (X_1X_2\dots X_{i-2}, p, X_{i-1}Y_iX_{i+1}\dots X_n).$$

If  $(p, Y_i, R) \in \delta(q, X_i)$  or  $q \xrightarrow{X_i/Y_i} p$ ,  then

$$(X_1X_2\dots X_{i-1}, q, X_iX_{i+1}\dots X_n) \Rightarrow_M \overset{X_i/Y_i}{\rightarrow} (X_1X_2\dots X_{i-1}Y_i, p, X_{i+1}\dots X_n).$$

or assume  $\alpha = \alpha'Z$  where  $Z \in \Gamma$  and  $\alpha' \in \Gamma^*$ .

$$\text{If } q \xrightarrow{X/Y} \leftarrow p, \quad (\alpha, q, X\beta) = (\alpha'Z, q, X\beta) \Rightarrow_M \overset{X/Y}{\leftarrow} (\alpha', p, ZY\beta).$$

$$\text{If } q \xrightarrow{X/Y} p, \quad (\alpha, q, X\beta) \Rightarrow_M \overset{X/Y}{\rightarrow} (\alpha Y, p, \beta).$$

*TM also is a finite automaton with read/write tape*

$$L(M) = \{w \in T^* \mid (\varepsilon, q_0, w) \Rightarrow_M^* (\alpha, f, \beta), \alpha, \beta \in \Gamma^*, f \in F\}$$

*L is recursively enumerable, if there is a TM M such that  $L = L(M)$ .*

*Class of recursively enumerable(RE) languages*

*Type 0 languages in Chomsky's hierarchy*

*We say TM halt, if  $\nexists \delta(q, X)$ (and does **not accept**) or  $q \in F$ (and **accepts**).*

*TM may runs forever(does not halt) for some  $x \notin L(M)$*

*Three cases*

*i) TM halts and accepts  $x$   $x \in L(M)$*

*ii) TM halts and does not accept  $x$   $x \notin L(M)$*

*iii) TM runs forever for  $x$   $x \notin L(M)$*

*If  $x \in L(M)$ , i) TM always halts and accepts  $x$ .*

*If  $x \notin L(M)$ , ii) TM may halt and does not accept  $x$*

*or iii) TM runs forever(does not halts) for  $x$ .*

**Example 8.2**  $L = \{0^n 1^n \mid n \geq 1\} \Rightarrow_M^* \{X^n Y^n \mid n \geq 1\}$

$q_0 \xrightarrow{0/X, \rightarrow} q_1, q_1 \xrightarrow{0/0, \rightarrow} q_1, q_1 \xrightarrow{Y/Y, \rightarrow} q_1, q_1 \xrightarrow{1/Y, \leftarrow} q_2, q_2 \xrightarrow{Y/Y, \leftarrow} q_2,$   
 $q_2 \xrightarrow{0/0, \leftarrow} q_2, q_2 \xrightarrow{X/X, \rightarrow} q_0, (X^i, q_0, 0^{n-i} Y^i 1^{n-i}) \Rightarrow (X^{i+1}, q_0, 0^{n-i-1} Y^{i+1} 1^{n-i-1})$   
 $q_0 \xrightarrow{Y/Y, \rightarrow} q_3, q_3 \xrightarrow{Y/Y, \rightarrow} q_3, q_3 \xrightarrow{B/B, \rightarrow} q_4. (X^n, q_0, Y^n) \Rightarrow (X^n Y^n B, q_4.B)$

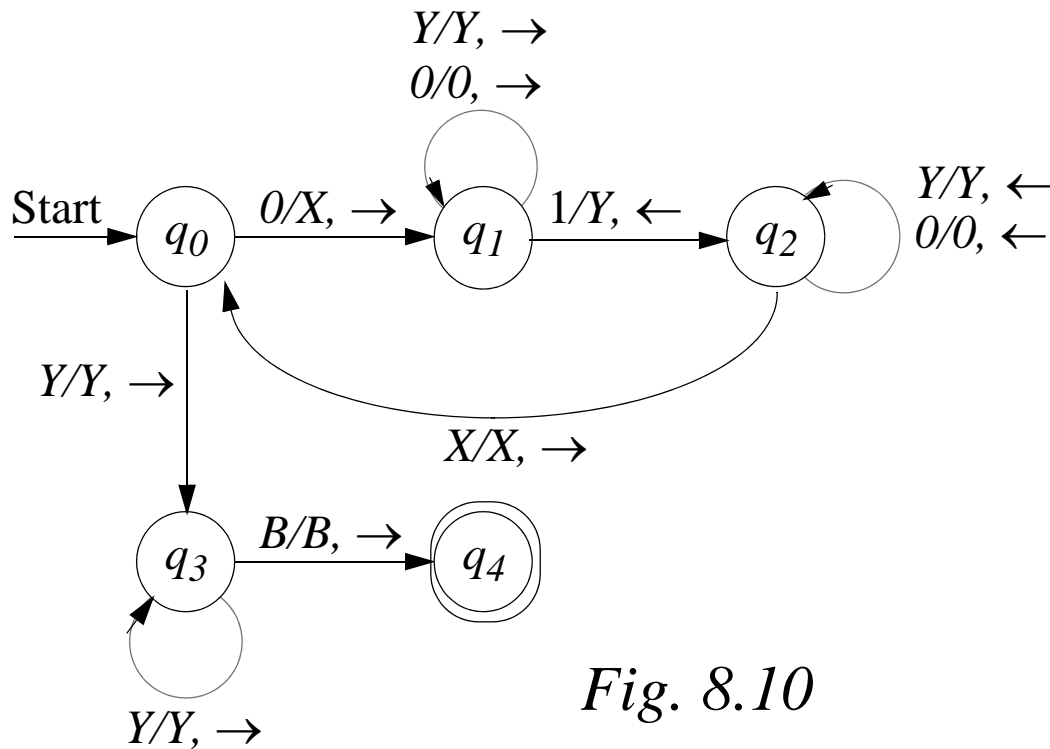


Fig. 8.10

$$\begin{aligned}
& (B, q_0, 0011) \Rightarrow^{0/X, \rightarrow} (X, q_1, 011) \Rightarrow^{0/0, \rightarrow} (X0, q_1, 11) \\
& \Rightarrow^{1/Y, \leftarrow} (X, q_2, 0Y1) \Rightarrow^{0/0, \leftarrow} (B, q_2, X0Y1) \Rightarrow^{X/X, \rightarrow} (X, q_0, 0Y1) \\
& \Rightarrow^{0/X, \rightarrow} (XX, q_1, Y1) \Rightarrow^{Y/Y, \rightarrow} (XXY, q_1, 1) \Rightarrow^{1/Y, \leftarrow} (XX, q_2, YY) \\
& \Rightarrow^{Y/Y, \rightarrow} (X, q_2, XY Y) \Rightarrow^{X/X, \rightarrow} (XX, q_0, YY) \Rightarrow^{Y/Y, \rightarrow} (XXY, q_3, Y) \\
& \Rightarrow^{Y/Y, \rightarrow} (XXYY, q_3, B) \Rightarrow^{B/B, \rightarrow} (XXYYB, q_4, B) \quad \textit{Accept for 0011.}
\end{aligned}$$

$$\begin{aligned}
& (B, q_0, 0010) \Rightarrow^{0/X, \rightarrow} (X, q_1, 010) \Rightarrow^{0/0, \rightarrow} (X0, q_1, 10) \\
& \Rightarrow^{1/Y, \leftarrow} (X, q_2, 0Y0) \Rightarrow^{0/0, \leftarrow} (B, q_2, X0Y0) \Rightarrow^{X/X, \rightarrow} (X, q_0, 0Y0) \\
& \Rightarrow^{0/X, \rightarrow} (XX, q_1, Y0) \Rightarrow^{Y/Y, \rightarrow} (XXY, q_1, 0) \Rightarrow^{0/0, \rightarrow} (XXY0, q_2, B) \\
& \Rightarrow^{B/?, ?} \quad \textit{Fail for 0010.}
\end{aligned}$$

**Example 8.4**  $m \dot{-} n = \max(m - n, 0)$     *monus or proper subtraction*

*if*  $m \geq n \rightarrow m \dot{-} n = m - n$  |  $m \leq n \rightarrow m \dot{-} n = 0$  *fi*

$0^m 10^n \Rightarrow_M^* 0^{m \dot{-} n}$ . (TM as an *integer function*)

	$0$	$1$	$B$
$q_0$	$(q_1, B, \rightarrow)$	$(q_5, B, \rightarrow)$	—
$q_1$	$(q_1, 0, \rightarrow)$	$(q_2, 1, \rightarrow)$	—
$q_2$	$(q_3, 1, \leftarrow)$	$(q_2, 1, \rightarrow)$	$(q_4, B, \leftarrow)$
$q_3$	$(q_3, 0, \leftarrow)$	$(q_3, 1, \leftarrow)$	$(q_0, B, \rightarrow)$
$q_4$	$(q_4, 0, \leftarrow)$	$(q_4, B, \leftarrow)$	$(q_6, 0, \rightarrow)$
$q_5$	$(q_5, B, \rightarrow)$	$(q_5, B, \rightarrow)$	$(q_6, B, \rightarrow)$
$q_6$	—	—	—

## 8.3 Programming Techniques for Turing Machines

### 8.3.1 Storage in the state

*state*                      *exercising control*  
                                  *storing symbols*                      **Fig. 8.13**

$$Q \Rightarrow Q \times \Gamma$$

*Exa. 8.6*(p. 338)  $01^* + 10^*$

$$Q \subseteq \{q_0, q_1\} \times \{0, 1, B\}$$

### 8.3.2 Multiple track (Fig. 8.13 in p. 338)

$\delta \subseteq (Q \times \Gamma^n) \times (Q \times \Gamma^n \times \{L, R\})$  *one control and storing  $n$  symbols*

*Exa. 8.7*(p. 339)  $L = \{wcw \mid w \in \{0, 1\}^+\}$

### 8.3.3 Subroutine

*Example 8.8 Multiply*                       $0^m 10^n 1 \Rightarrow 0^{mn}$ .

## 8.4 Extension to the Basic Turing Machine

### Multitape Turing Machine

$$\delta \subseteq (Q \times \Gamma^n) \times (Q \times \Gamma^n \times \{L, R, S\}^n).$$

*S: no head move*

**Theorem 8.9** *Every language accepted by multitape TM is recursively enumerable.  $O(n^2)$ .*

**proof** *If multi tape TM  $M$  has  $k$  tapes, single tape TM  $N$  with  $2k$  tracks*

*$k$  tracks: **simulate** the contents of  $k$  tapes.*

*$k$  tracks: mark the **head position** of  $k$  tapes.*

*One move in  $M =$  two sweeps of in  $N$ .  $O(n^2)$*

*left to right sweep*

***update** tape contents and head position*

*count number of heads to be updated(right bound)*

*right to left sweep*

*restore the head position of TM  $N$  to the leftmost head position*



## ***Nondeterministic Turing Machine***

$$\delta: Q \times \Gamma \rightarrow 2^Q \times \Gamma \times \{L, R\}.$$

***Theorem 8.11*** *If  $M_N$  is a nondeterministic TM, then*

*there is a deterministic TM  $M_D$  such that  $L(M_N) = L(M_D)$ .*

***proof*** *Every nondeterministic moves of  $M_N$ , path in the decision tree.*

*Assume the degree of the tree  $\leq k$ .*

*Let  $M_D$  has a two tapes.*

*tape 1: sequence of choices in the decision tree*

*tape 2: simulate the content of  $M_N$ .*

***systematically simulate all moves of  $M_N$ .***

$O(k^n)$

NP

## 8.5 Restricted Turing Machine

### 8.5.1 Turing Machine with semi-infinite tapes

#### Theorem 8.12

*A two tracks of semi-infinite tapes simulates a two-way infinite tape.*

### 8.5.2 Multistack machine

*a read only input tape*

*multiple stacks*

$$\delta: Q \times T \times \Gamma^n \rightarrow Q \times \Gamma^* \times \dots \times \Gamma^*.$$

**Theorem 8.13** *If  $L$  is accepted by a TM,  $L$  is accepted by two-stack machine.*

**proof** *Left of head      one stack*  
*Right of head      another stack*

### 8.5.3 Counter Machine

*stack machine*

**stack alphabet** =  $\{Z_0, X\}$

$Z_0$ : bottom stack marker

$X$ : # of  $B$ 's represents a number

*we can test if number is zero*

*we can not directly test if two numbers are same*

**Theorem 8.14** *A three-counter machine can simulate TM*

**proof** *two-stack machine = TM*

*Suppose stack vocabulary has  $r-1$  symbols.*

*stack contents:  $X_1, X_2, \dots, X_n \leftrightarrow i = X_n r^{n-1} + X_{n-1} r^{n-2} + \dots + X_1$ .*

*$r$ -nary number with LSB on top of the stack*

*two counter = two stack contents ( $i$ )*

1. *pop X*:  $i \rightarrow i/r$                        $counter3 := 0;$   
     **while**  $i = 0$  **do**  $i := i - r; counter3 := counter3 + 1$  **od**  
     */\* counter3 = i/r (pop X) \*/*
2. *change X to Y*:  $i := i + Y - X$
3. *push Y*:  $i \rightarrow ir + Y$                        $counter3 := 0;$   
     **while**  $i = 0$  **do**  $i \rightarrow i - 1; counter3 := counter3 + r$  **od**  
     */\* counter3 = ir \*/*  
      $counter3 := counter3 + Y (= ir + Y, push Y)$

**Theorem 8.15** *A two-counter machine can simulate TM*

*Three counters  $i, j, k$  are represented by one counter  $m = 2^i 3^j 5^k$ .*

1. *increase  $i, j, k$ : multiply  $m$  by 2, 3, or 5*
2. *test if  $i, j, k = 0$ : divisible by 2, 3, 5: if decrease by 2, 3, 5 until zero*
3. *decrease  $i, j, k$ : divide  $m$  by 2, 3, or 5*

***Turing Machine is a number.***

## **Enumeration machine**

*work tape: move in either direction, read/write any symbol in  $\Gamma$ .*

*output tape: move right only, write symbols in  $\Sigma$  and # (separator).*

1. Generate  $\forall x \in \Sigma^*$  in systematic way (lexicographic order)

$\varepsilon, (a_1, a_2, \dots, a_n), (a_1a_1, a_1a_2, \dots, a_na_n), \dots$

2. simulates  $x$  for  $M$

3. If  $M$  accepts  $x$ , output  $x$

*If  $x_i$  runs forever,  $x_{i+1} \notin O(M)$ , but  $x_{i+1} \in L(M)$*

### **time sharing**

*Instead of simulating  $M$  on the input string one at a time, working a few steps in one string and moves to another*

### **pair generator**

$(0, 0), (0, 1), (1, 0), (0, 2), (1, 1), (2, 0), \dots, (i, j), \dots$

*Simulate for  $x_i$  for  $j$ -steps*

## ***8.6 Turing Machine and Computer***

### ***Simulating Turing machine by computer***

*infinite tape*

*storage device*

### ***Turing Machine(multi tape) as a computer***

*Arithmetic and Logic Unit*

*fa(decoder) with multi-tapes(registers, PC, ...)*

*memory*

*multi-tapes(cache, memory, disk, ...)*

*input devices*

*input tapes*

*output devices*

*output tapes*

## *Turing machine as a program(operational semantics)*

*A Turing Machine(TM)  $M$  is a transition(rewriting) system,*

*$M = (C, \rightarrow)$  where*

- 1.  $C$  is a set of **configurations**(state of memories, numbers),*
  - 2.  $\rightarrow \subseteq C \times C$ .*
- $c, c' \in C, c \rightarrow c'$ .*

*The transition system is **deterministic(monogenic)**, if*

*$\forall c, c_1, c_2 \in C, \text{ if } c \rightarrow c_1 \wedge c \rightarrow c_2, \text{ then } c_1 = c_2$ .*

*Let  $I, T \subseteq C$  be two sets of **initial** and **terminate(final)** configurations.*

*$i \in I, t \in T$ , when  $i \rightarrow^* t$ ,  $(i, t)$  is the **run** of the transition system*

*It is usual to arrange that if  $t \in T$ , then  $\nexists t' \in C . \exists . t \rightarrow t'$ .*

*If  $c \notin T \wedge \nexists c' \in C . \exists . c \rightarrow c'$ , the configuration  $c$  is said to be **stuck***