CS 4410

Automata, Computability, and Formal Language

Dr. Xuejun Liang

Chapter 9

Turing Machines

1. The Standard Turing Machine
   - Definition of a Turing Machine
   - Turing Machines as Language Accepters
   - Turing Machines as Transducers
2. Combining Turing Machines for Complicated Tasks
3. Turing's Thesis

Learning Objectives

At the conclusion of the chapter, the student will be able to:

- Describe the components of a standard Turing machine
- State whether an input string is accepted by a Turing machine
- Construct a Turing machine to accept a specific language
- Trace the operation of a Turing machine transducer given a sample input string
- Construct a Turing machine to compute a simple function
- State Turing’s thesis and discuss the circumstantial evidence supporting it

Definition of a Turing Machine

Definition 9.1: A **Turing machine** M is defined by

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

where

Q is a finite set of **internal states**,

Σ is the **input alphabet**,

Γ is a finite set of symbols called the **tape alphabet**,

$\delta: Q\times\Gamma \rightarrow Q\times\Gamma\times\{L, R\}$ is the **transition function**,

$q_0 \in Q$ is the **initial state**,

$\square \in \Gamma$ is a special symbol called **blank**,

$F \subseteq Q$ is the set of **final states**

- The tape acts as the input, output, and storage medium.
- The read-write head can travel in both directions, processing one symbol per move
- Input string is surrounded by blanks, so $\Sigma \subseteq \Gamma - \{\square\}$

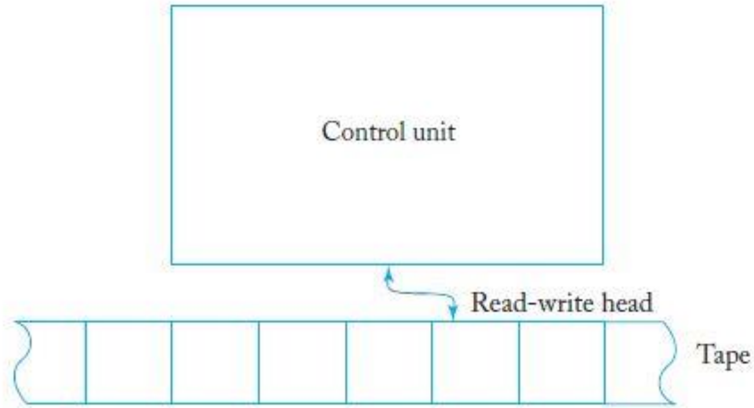


FIGURE 9.1

Definition of a Turing Machine

Transition function: δ: $Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$

- Input to δ consists of the current state of the control unit and the current tape symbol
- Output of δ consists of a new state, new tape symbol, and location of the next symbol to be read (L or R)
- δ is a partial function, so that some (state, symbol) input combinations may be undefined
- δ causes the machine to change states and possibly overwrite the tape contents

Configuration: tape symbols, state, tape head position

Halt: it reaches to a configuration for which δ is not defined

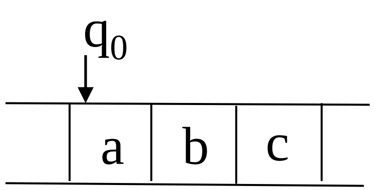
Computation: The sequence of configurations leading to a halt state.

Examples

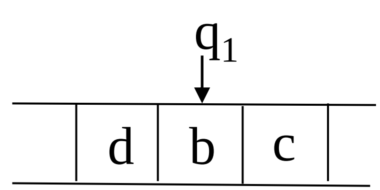
Example 9.1: Given the sample transition rule

$$\delta(q_0, a) = (q_1, d, R)$$

According to this rule, when the control unit is in state q_0 and the tape symbol is a, the new state is q_1, the symbol d replaces a on the tape, and the read-write head moves one cell to the right



Configuration before the move



Configuration after the move

$q_0abc \vdash dq_1bc$

Examples

Example 9.2: Given $Q = \{q_0, q_1\}$, $\Sigma = \{a, b\}$, $\Gamma = \{a, b, \square\}$, $F = \{q_1\}$

$$\delta(q_0, a) = (q_0, b, R)$$
$$\delta(q_0, b) = (q_0, b, R)$$
$$\delta(q_0, \square) = (q_1, \square, L)$$

FIGURE 9.4

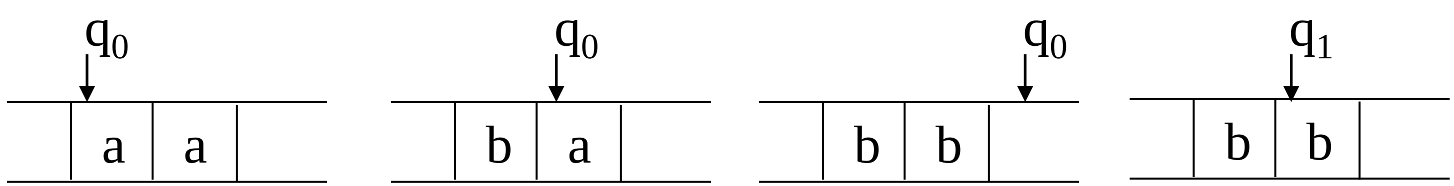
- The machine starts in q_0 and, as long as it reads a's, will replace them with b's and continue moving to the right, but b's will not be modified
- When a blank is found, the control unit switches states to q_1 and moves one cell to the left
- The machine halts whenever it reaches a configuration for which δ is not defined (in this case, state q_1)

Examples

Example 9.2: Given $Q = \{q_0, q_1\}$, $\Sigma = \{a, b\}$, $\Gamma = \{a, b, \square\}$, $F = \{q_1\}$

$$\delta(q_0, a) = (q_0, b, R)$$
$$\delta(q_0, b) = (q_0, b, R)$$
$$\delta(q_0, \square) = (q_1, \square, L)$$

FIGURE 9.4



A sequence of moves as the machine processes a tape with initial contents aa

$$q_0aa\square \vdash bq_0a\square \vdash bbq_0\square \vdash bq_1b\square$$

Transition Graphs for Turing Machines

- In a Turing machine transition graph, each edge is labeled with three items: current tape symbol, new tape symbol, and direction of the head move
- Figure 9.4 shows the transition graph for the Turing Machine in Example 9.2

FIGURE 9.4

A Turing Machine that Never Halts

It is possible for a Turing machine to never halt on certain inputs, as is the case with Example 9.3 (below) and input string ab

Example 9.3: Given $Q = \{q_0, q_1\}$, $\Sigma = \{a, b\}$, $\Gamma = \{a, b, \square\}$, $F = \{\}$

$\delta(q_0, a) = (q_1, a, R)$
$\delta(q_0, b) = (q_1, b, R)$
$\delta(q_0, \square) = (q_1, \square, R)$
$\delta(q_1, a) = (q_0, a, L)$
$\delta(q_1, b) = (q_0, b, L)$
$\delta(q_1, \square) = (q_0, \square, L)$

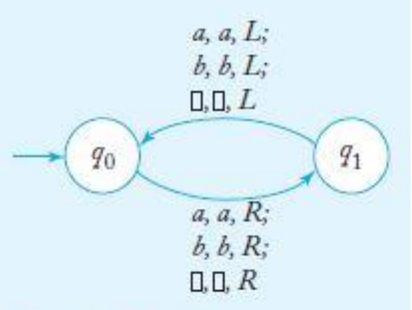


FIGURE 9.5

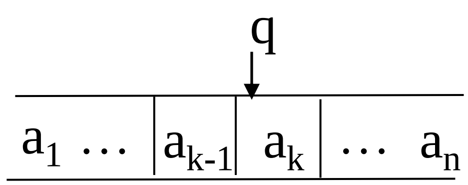
This machine with input string ab runs forever –in an infinite loop- with the read-write head moving alternately right and left, but making no modifications to the tape

Standard Turing Machine

1. One tape unbounded in both directions
2. Deterministic: At most one move for each configuration
3. No special input file and No special output device

Configuration (Instantaneous description):

$x_1 q x_2$ (or $a_1 a_2 \ldots a_{k-1} q a_k a_{k+1} \ldots a_n$)



Move from one configuration to another:

$abq_1cd \vdash abeq_2d$ (if $\delta(q_1, c) = (q_2, e, R)$)

$abq_1cd \vdash aq_2bed$ (if $\delta(q_1, c) = (q_2, e, L)$)

Examples

Example 9.4, 9.5: Configurations and moves in Example 9.2

Example 9.2

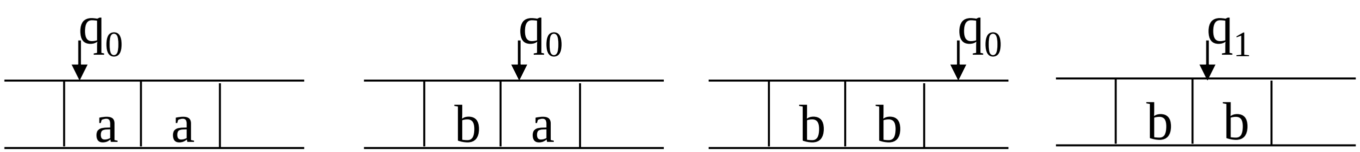
$\delta(q_0, a) = (q_0, b, R)$

$\delta(q_0, b) = (q_0, b, R)$

$\delta(q_0, \square) = (q_1, \square, L)$

FIGURE 9.4

A sequence of moves with initial tape contents aa



Configurations: $q_0aa\square$ $bq_0a\square$ $bbq_0\square$ $bq_1b\square$

Moves: $q_0aa\square \vdash bq_0a\square \vdash bbq_0\square \vdash bq_1b\square$

Turing Machines as Language Accepters

Definition 9.3: Let $M=(Q, \Sigma, \Gamma, \delta, q_0, \square, F)$ be a Turing machine. Then the language accepted by M is

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

- Turing machines can be viewed as language accepters
- The language accepted by a Turing machine is the set of all strings which cause the machine to halt in a final state, when started in its standard initial configuration (q_0, leftmost input symbol)
- A string is rejected if
  - The machine halts in a nonfinal state, or
  - The machine never halts

Examples

Example 9.6: For $\Sigma=\{0, 1\}$, design a Turing machine M such that $L(M)=L(00^*)$

$Q=\{q_0, q_1, q_2\}$, $F=\{q_2\}$, $\Gamma=\{0, 1, \square\}$,

Example 9.7: For $\Sigma=\{0, 1\}$, design a Turing machine that accept

$$L=\{a^n b^n : n \geq 1\}$$

$$Q=\{q_0, q_1, q_2, q_3, q_4\}, F=\{q_4\}, \Gamma=\{a, b, x, y, \square\}$$

Example 9.7: For $\Sigma=\{0, 1\}$, design a Turing machine that accept

$$L=\{a^n b^n : n \geq 1\}$$

$$Q=\{q_0, q_1, q_2, q_3, q_4\}, F=\{q_4\}, \Gamma=\{a, b, x, y, \square\}$$

Example 9.7: For $\Sigma=\{0, 1\}$, design a Turing machine that accept

$$L=\{a^n b^n : n \geq 1\}$$

$$Q=\{q_0, q_1, q_2, q_3, q_4\}, F=\{q_4\}, \Gamma=\{a, b, x, y, \square\}$$

Examples

Example 9.8: For $\Sigma=\{a, b, c\}$, design a Turing machine that accept

$$L=\{a^n b^n c^n : n \geq 1\}$$

$$Q=\{q_0, q_1, q_2, q_3, q_4\},\ F=\{q_4\},\ \Gamma=\{a, b, c, x, y, z, \square\}$$

x

y

Turing Machines as Transducers

Definition 9.3: A function f with domain D is said to be Turing-computable or just computable if there exists some Turing machine $M=(Q,\Sigma,\Gamma,\delta,q_0,\square,F)$ such that for all $w \in D$

$$q_0 w \overset{*}{\vdash} q_f f(w), \; q_f \in F$$

- Turing machines provide an abstract model for digital computers, acting as a transducer that transforms input into output
- A *Turing machine transducer* implements a function that treats the original contents of the tape as its input and the final contents of the tape as its output
- A function is *Turing-computable* if it can be carried out by a Turing machine capable of processing all values in the function domain

Example 9.9: Given two positive integers x and y, design a Turing machine that computes x + y.

x is encoded by its uniary representation $w(x)$

+ is represented by 0

$x + \text{y}$ is encoded *by* $w(x)0w(y)$

$$q_0 w(x) 0 w(y) \overset{*}{\vdash} q_f w(x+y)$$

- The transducer has $Q = \{ q_0, q_1, q_2, q_3, q_4 \}$ with initial state q_0 and final state q_4
- The defined values of the transition function are

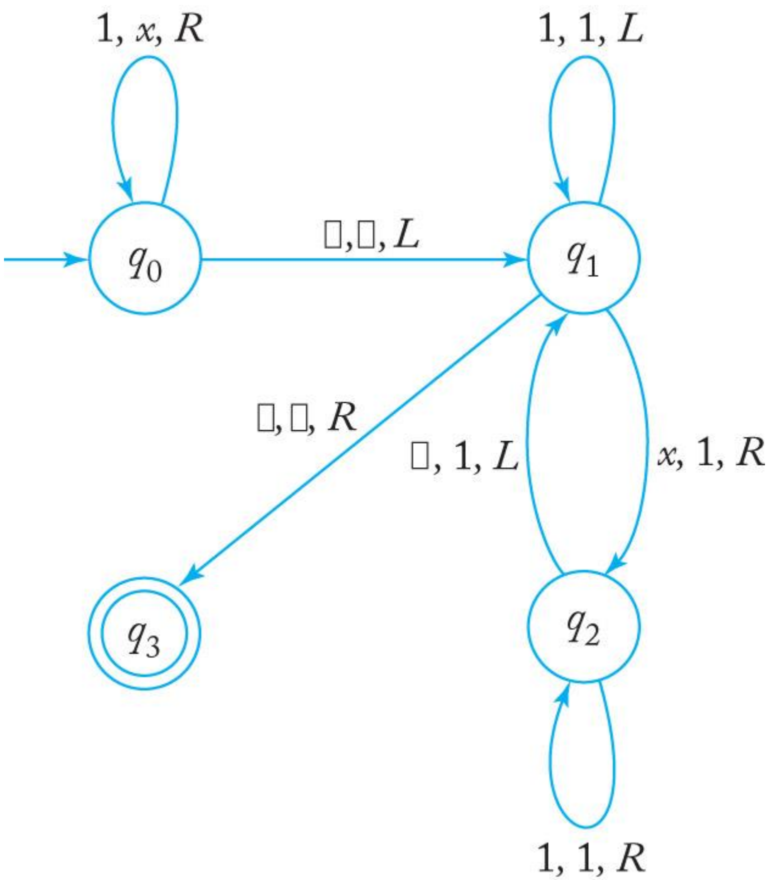
$$\delta(q_0, 1) = (q_0, 1, R) \qquad \delta(q_0, 0) = (q_1, 1, R)$$
$$\delta(q_1, 1) = (q_1, 1, R) \qquad \delta(q_1, \square) = (q_2, \square, L)$$
$$\delta(q_2, 1) = (q_3, \square, L) \qquad \delta(q_3, 1) = (q_3, 1, L)$$
$$\delta(q_3, \square) = (q_4, \square, R)$$

- When the machine halts, the read-write head is positioned on the leftmost symbol of the unary representation of $x + y$

Example 9.10: Design a Turing machine that copies strings of 1's. More precisely, find a machine that perform the computation

$$q_0w \overset{*}{\vdash} q_f\, ww$$

for any $w \in \{1\}^+$



Example 9.11: Let x and y be two positive integers represented in unary notation. Construct a Turing machine that will halt in a final state q_y if $x \geq y$, and that will halt in a non-final state q_n if $x < y$.

$$q_0 w(x) 0 w(y) \overset{*}{\vdash} q_y \mathrm{w(x)0w(y)}, \qquad if\ x \geq y,$$
$$q_0 w(x) 0 w(y) \overset{*}{\vdash} q_n \mathrm{w(x)0w(y)}, \qquad if\ x < y.$$

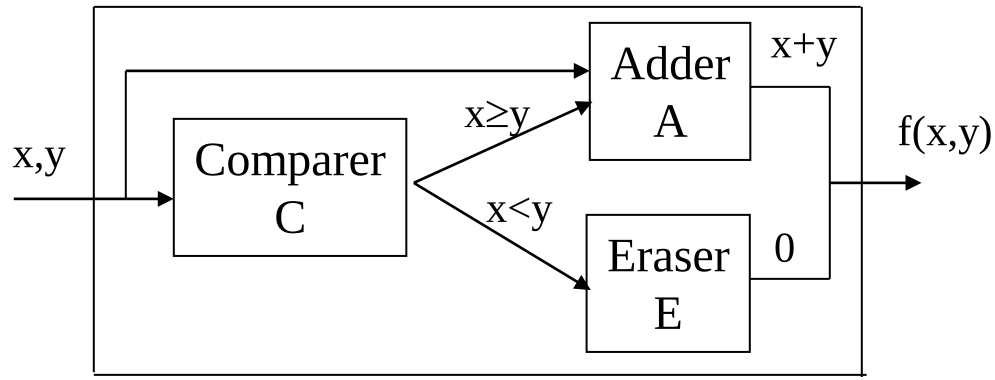
Example 9.11: Let x and y be two positive integers represented in unary notation. Construct a Turing machine that will halt in a final state q_y if $x \geq y$, and that will halt in a non-final state q_n if $x < y$.

$$q_0 111011 \overset{*}{\vdash} q_y 111011, \qquad x = 3 \; and \; y = 2,$$
$$q_0 110111 \overset{*}{\vdash} q_n 110111, \qquad x = 2 \; and \; y = 3.$$

Combining Turing Machines for Complicated Tasks

Example 9.12: Design a Turing machine that computes the function

$$f(x,y) = x + y, \quad if\ x \geq y,$$
$$= 0, \quad if\ x < y.$$



$$q_{C,0}w(x)0w(y) \overset{*}{\vdash} q_{A,0}\mathrm{w(x)0w(y)}, \qquad if\ x \geq y,$$
$$q_{C,0}w(x)0w(y) \overset{*}{\vdash} q_{E,0}\mathrm{w(x)0w(y)}, \qquad if\ x < y.$$

$$q_{A,0}\mathrm{w(x)0w(y)} \overset{*}{\vdash} q_{A,f}\mathrm{w(x)w(y)0}$$
$$q_{E,0}\mathrm{w(x)0w(y)} \overset{*}{\vdash} q_{E,f}\,0$$

Combining Turing Machines for Complicated Tasks (Cont.)

Example 9.13: Consider the instruction: *If a then* q_j *else* q_k.

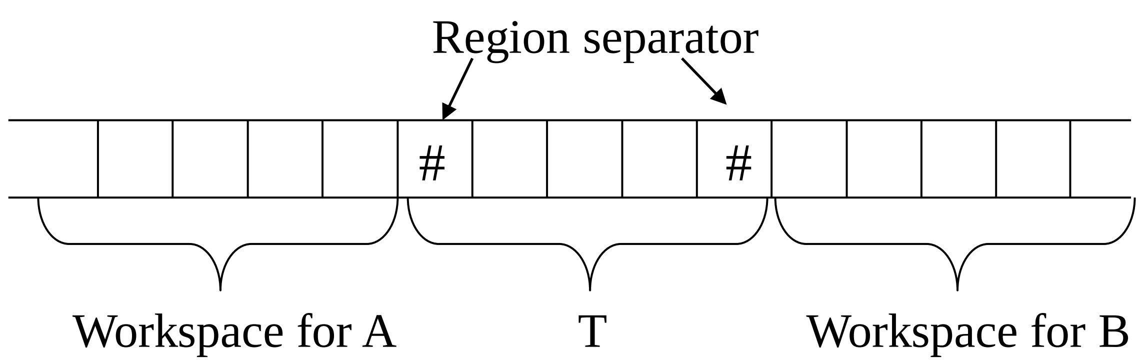
$\delta(q_i,a) = (q_{j0},a,R)$ for all $q_i \in Q$,

$\delta(q_i,b) = (q_{k0},b,R)$ for all $q_i \in Q$ and all $b \in \Gamma-\{a\}$,

$\delta(q_{j0},c) = (q_j,c,L)$ for all $c \in \Gamma$,

$\delta(q_{k0},a) = (q_k,c,L)$ for all $c \in \Gamma$.

Consider subprogram



Turing's Thesis

Turing thesis (a hypothesis): Any computation that can be carried out by mechanical means can be performed by some Turing machine.

A computation is mechanical if and only if it can be performed by some Turing machine.

1. Anything that can be done on any existing digital computer can also be done by a Turing machine.
2. No one has yet been able to suggest a problem, solvable by what we intuitively consider an algorithm, for which a Turing machine program cannot be written.
3. Alternative models have been proposed for mechanical computation, but none of them are more powerful than the Turing machine model.

Turing's Thesis (Cont.)

An acceptance of Turing's Thesis leads to a definition of an algorithm:

Definition 9.3: An algorithm for a function f: D→R is a Turing machine M, which given as input any d ∈ D on its tape, eventually halts with the correct answer f(d) on its tape. Specially, we can require that

$$q_0 d \vdash_M^* q_f f(d), \quad q_f \in F$$

for all d ∈ D