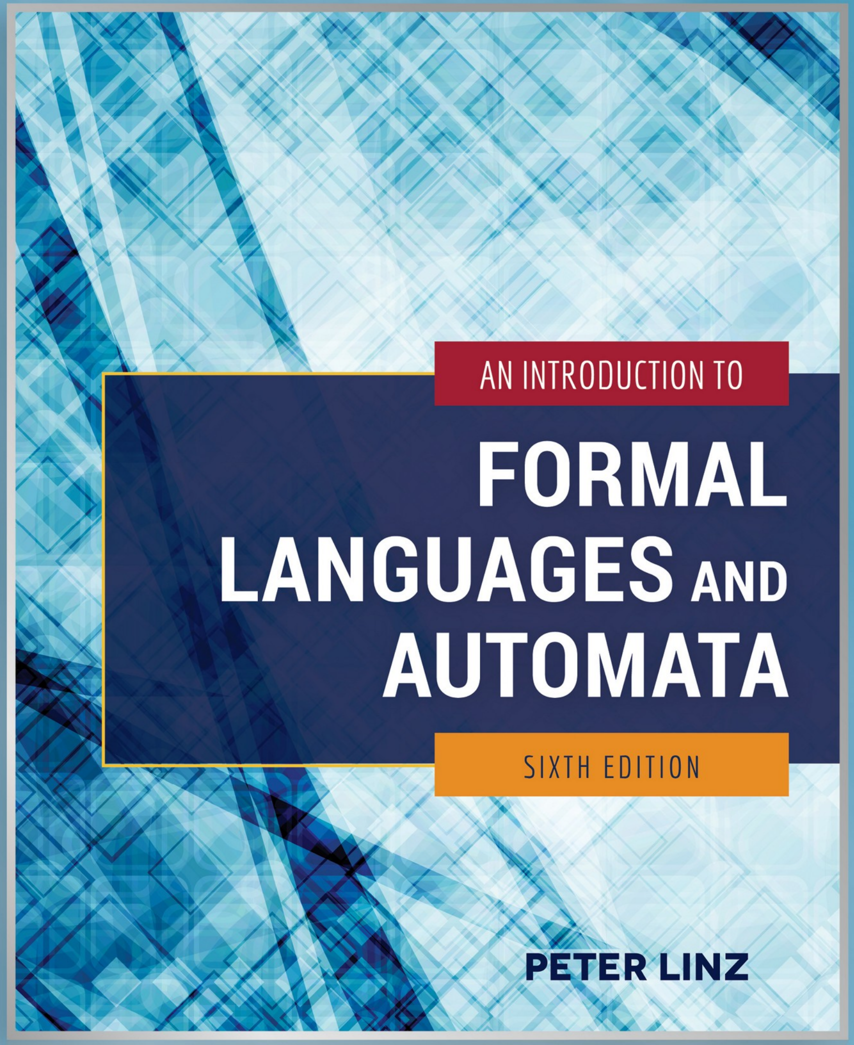
Chapter 9

TURING MACHINES



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

The Standard Turing Machine

- A standard Turing machine has unlimited storage in the form of a tape consisting of an infinite number of cells, with each cell storing one symbol
- The read-write head can travel in both directions, processing one symbol per move
- A deterministic control function causes the machine to change states and possibly overwrite the tape contents
- Input string is surrounded by blanks, so the input alphabet is considered a proper subset of the tape alphabet

Diagram of a Standard Turing Machine

In a standard Turing machine, the tape acts as the input, output, and storage medium.

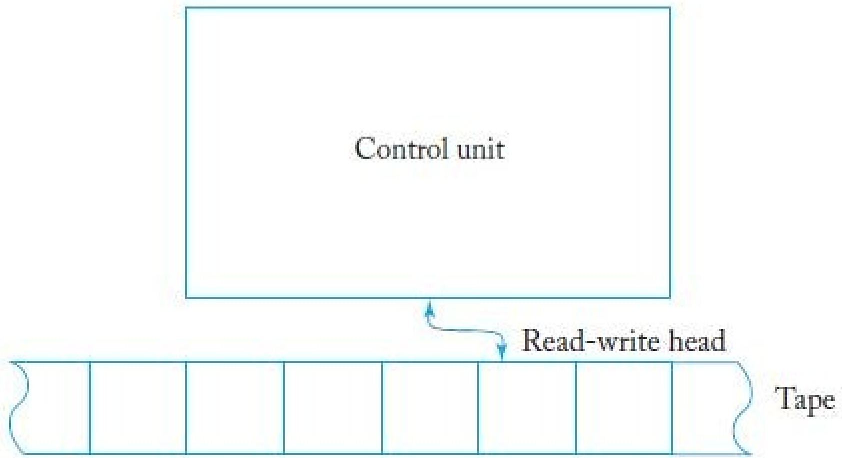


FIGURE 9.1

Definition of a Turing Machine

- A *Turing Machine* is defined by:
  - A finite set of internal states Q
  - An input alphabet Σ
  - A tape alphabet Γ
  - A transition function δ
  - A special symbol • from Γ called the blank
  - An initial state q_0
  - A set of final states F
- Input to the transition function 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

Sample Turing Machine Transition

- Example 9.1 presents 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

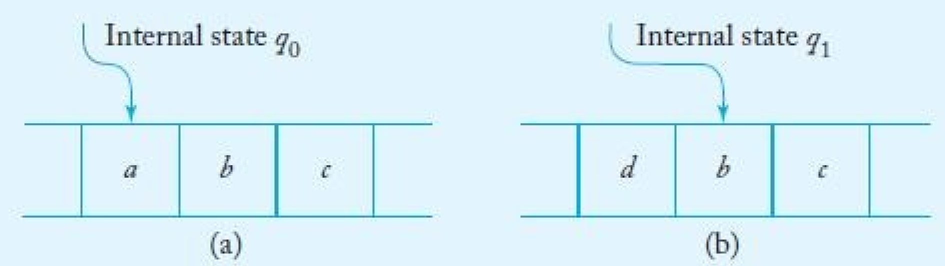


FIGURE 9.2 The situation (a) before the move and (b) after the move.

A Sample Turing Machine

- Example 9.2: Consider the Turing machine

  $Q = \{ q_0, q_1 \}$, $\Sigma = \{ a, b \}$, $\Gamma = \{ a, b, \bullet \}$, $F = \{ q_1 \}$

  with initial state q_0 and transition function given by:

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

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

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

- 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)

Tracing the Operation of a Turing Machine

Figure 9.3 shows several stages of the operation of the Turing Machine in Example 9.2 as it processes a tape with initial contents aa

FIGURE 9.3 A sequence of moves.

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

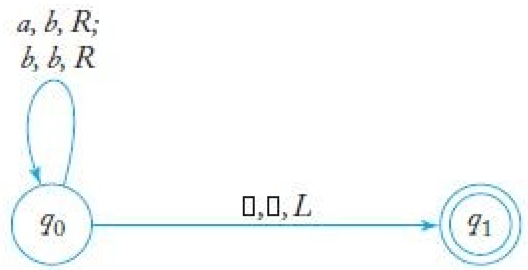


FIGURE 9.4

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

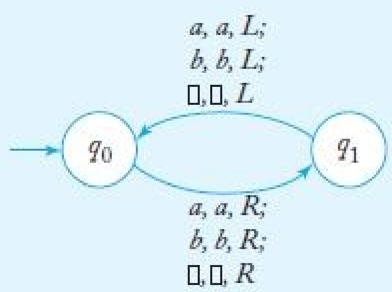


FIGURE 9.5

The Language Accepted by a Turing Machine

- 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

Turing Machines as Transducers

- 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

A Sample Turing Machine Transducer

- Given two positive integers x and y in unary notation, separated by a single zero, the Turing machine below computes the function 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

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

$\delta(q_1, 1) = (q_1, 1, R)$ $\delta(q_1, \bullet) = (q_2, \bullet, L)$

$\delta(q_2, 1) = (q_3, 0, L)$ $\delta(q_3, 1) = (q_3, 1, L)$

$\delta(q_3, \bullet) = (q_4, \bullet, R)$

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

Combining Turing Machines

- By combining Turing Machines that perform simple tasks, complex algorithms can be implemented
- For example, assume the existence of a machine to compare two numbers (comparer), one to add two numbers (adder), and one to erase the input (eraser)
- Figure 9.8 shows the diagram of a Turing Machine that computes the function $f(x, y) = x + y$ (if $x \geq y$), 0 (if $x < y$)

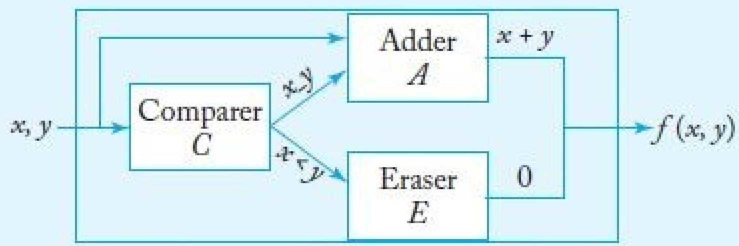


FIGURE 9.8

Turing's Thesis

- How powerful are Turing machines?
- *Turing's Thesis* contends that any computation carried out by mechanical means can be performed by some Turing machine
- An acceptance of Turing's Thesis leads to a definition of an algorithm:

  An *algorithm* for a function $f : D \rightarrow R$ is a Turing machine M, which given any $d \in D$ on its tape, eventually halts with the correct answer $f(d) \in R$ on its tape

Evidence Supporting Turing's Thesis

- Anything that can be done on any existing digital computer can also be done by a Turing machine
- 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
- Alternative models have been proposed for mechanical computation, but none of them is more powerful than the Turing machine model