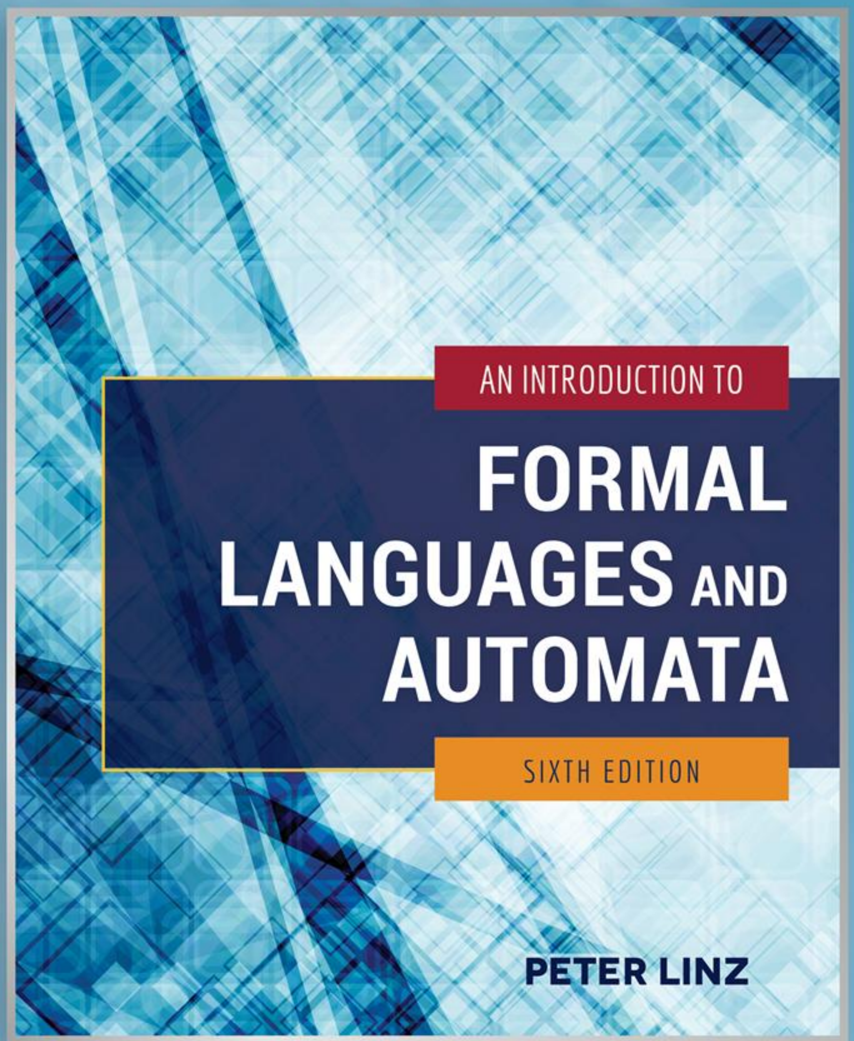
Chapter 10

OTHER MODELS OF TURING MACHINES



Learning Objectives

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

- Explain the concept of equivalence between classes of automata
- Describe how a Turing machine with a stay-option can be simulated by a standard Turing machine
- Describe how a standard Turing machine can be simulated by a machine with a semi-infinite tape
- Describe how off-line and multidimensional Turing machines can be simulated by standard Turing machines
- Construct two-tape Turing machines to accept simple languages
- Describe the operation of nondeterministic Turing machines and their relationship to deterministic Turing machines
- Describe the components of a universal Turing machine
- Describe the operation of linear bounded automata and their relationship to standard Turing machines

Equivalence of Classes of Automata

- Two automata are equivalent if they accept the same language
- Given two classes of automata C_1 and C_2, if for every automaton in C_1 there is an equivalent automaton in C_2, the class C_2 is at least as powerful as C_1
- If the class C_1 is at least as powerful as C_2, and the converse also holds, then the classes C_1 and C_2 are equivalent
- Equivalence can be established either through a constructive proof or by simulation

Turing Machines with a Stay-Option

- In a *Turing Machine with a Stay-Option*, the read-write head has the option to say in place after rewriting the cell content
- Theorem 10.1 states the class of Turing machines with a stay-option is equivalent to the class of standard Turing machines
- To show equivalence, we argue that any machine with a stay-option can be simulated by a standard Turing machine, since the stay-option can be accomplished by
  - A rule that rewrites the symbol and moves right, and
  - A rule that leaves the tape unchanged and moves left

Turing Machines with Semi-Infinite Tape

- As shown in Figure 10.2, a common variation of the standard Turing machine is one in which the tape is unbounded only in one direction
- A *Turing machine with semi-infinite tape* is otherwise identical to the standard model, except that no left move is possible when the read-write head is at the tape boundary

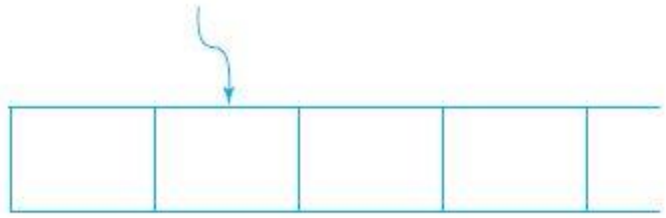


FIGURE 10.2

Equivalence of Standard Turing Machines and Semi-Infinite Tape Machines

- The classes are equivalent because, as shown in Figure 10.4, any standard Turing machine can be simulated by a machine with a semi-infinite tape
- The simulating machine has two tracks: the upper track contains the symbols to the right of an arbitrary reference point, while the lower track contains those to the left of the reference point in reverse order

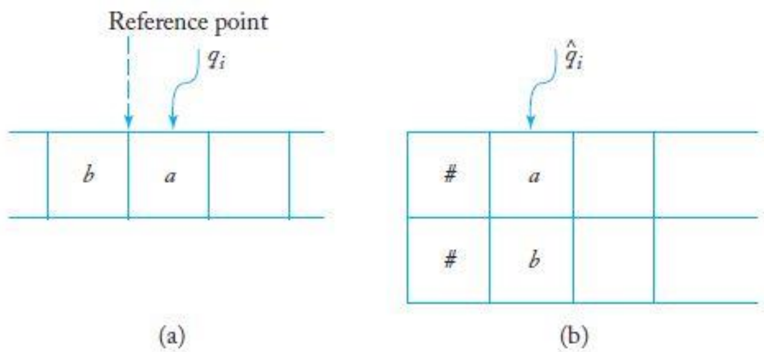


FIGURE 10.4 (a) Machine to be simulated. (b) Simulating machine.

The Off-Line Turing Machine

- As shown in Figure 10.6, an *off-line Turing machine* has a read-only input file in addition to the read-write tape
- Transitions are determined by both the current input symbol and the current tape symbol

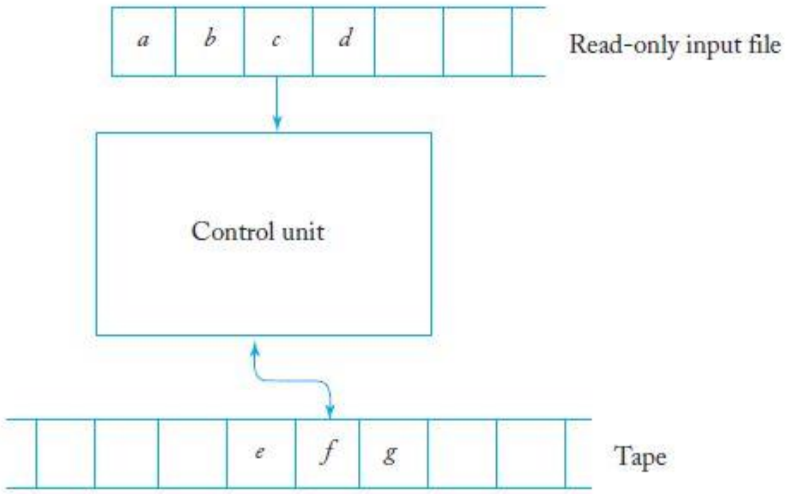


FIGURE 10.6

Equivalence of Standard Turing Machines and Off-Line Turing Machines

- The classes are equivalent because, as shown in Figure 10.7, a standard Turing machine with four tracks can simulate the computation of an off-line machine
- Two tracks are used to store the input file contents and current position, while the other two tracks store the contents and current position of the read-write tape

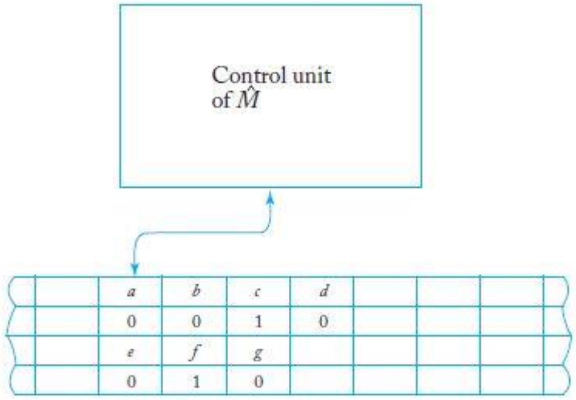


FIGURE 10.7

Multitape Turing Machines

- As shown in Figure 10.8, a *multitape Turing machine* has several tapes, each with its own independent read-write head
- A sample transition rule for a two-tape machine must consider the current symbols on both tapes:

$$\delta(q_0, a, e) = (q_1, x, y, L, R)$$

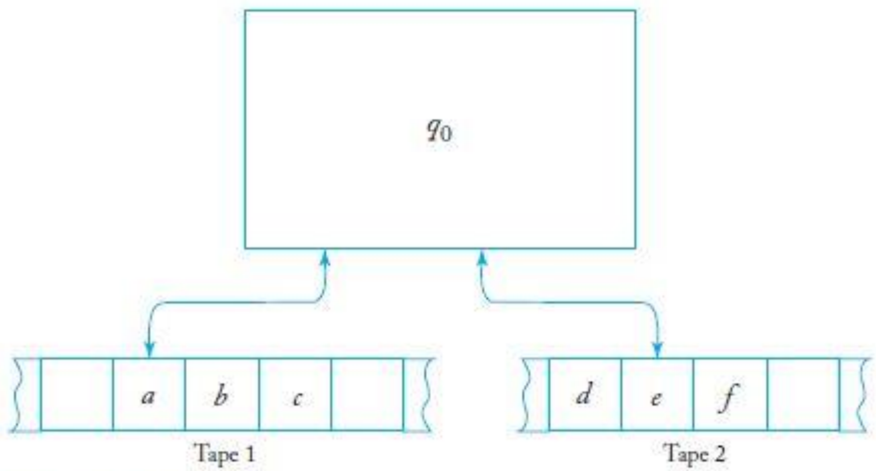


FIGURE 10.8

Equivalence of Standard Turing Machines and Multitape Turing Machines

- The classes are equivalent because, as shown in Figure 10.11, a standard Turing machine with four tracks can simulate the computation of an off-line machine
- Two tracks are used to store the contents and current position of tape 1, while the other two tracks store the contents and current position of tape 2

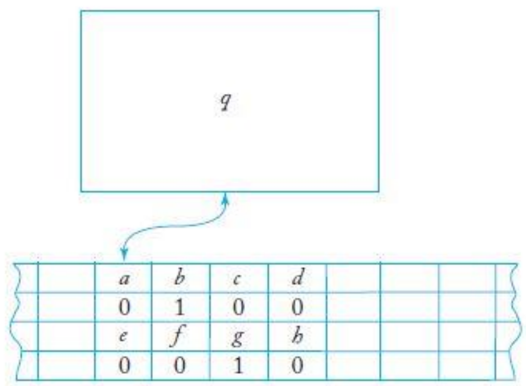


FIGURE 10.11

Multidimensional Turing Machines

- As shown in Figure 10.12, a m*ultidimensional Turing machine* has a tape that can extend infinitely in more than one dimension
- In the case of a two-dimensional machine, the transition function must specify movement along both dimensions

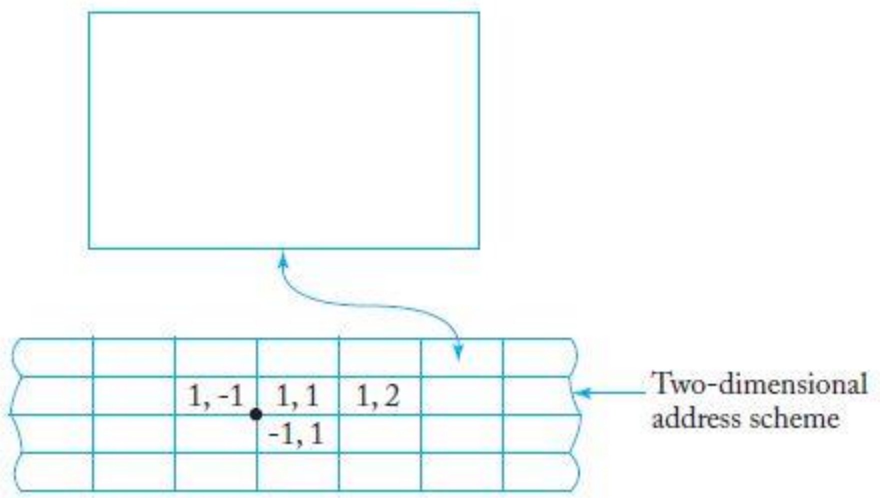


FIGURE 10.12

Equivalence of Standard Turing Machines and Multidimensional Turing Machines

- The classes are equivalent because, as shown in Figure 10.13, a standard Turing machine with two tracks can simulate the computation of a two-dimensional machine
- In the simulating machine, one track is used to store the cell contents and the other one to keep the associated address

	a				*b*						
	1	#	2	#	1	0	#	-	3	#	

FIGURE 10.13

Nondeterministic Turing Machines

- A *nondeterministic Turing machine* is one with potentially many transition choices for a given (state, symbol) combination
- Example 10.2 presents a sample transition rule for a nondeterministic machine:

$$\delta(q_0, a) = \{(q_1, b, R), (q_2, c, L)\}$$

- Since multiple transitions may be applied at each step, the machine may have multiple active simultaneous threads, any of which may accept the input string when the thread halts
- For every nondeterministic Turing machine, there is an equivalent deterministic machine that can simulate its operation

A Universal Turing Machine

- A *universal Turing machine* is a reprogrammable Turing machine which, given as input the description of a Turing machine M and a string w, can simulate the computation of M on w
- A universal Turing machine has the structure of a multitape machine, as shown in Figure 10.16

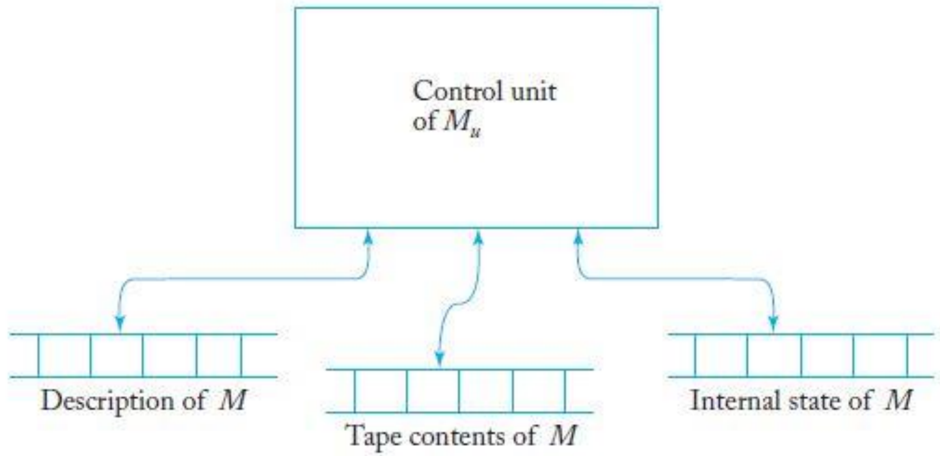


FIGURE 10.16

Linear Bounded Automata

- The power of a standard Turing machine can be restricted by limiting the area of the tape that can be used
- A *linear bounded automaton* is a Turing machine that restricts the usable part of the tape to exactly the cells used by the input
- Input can be considered as bracketed by two special symbols or markers which can be neither overwritten nor skipped by the read-write head
- Linear bounded automata are assumed to be nondeterministic and accept languages in the same manner as other Turing machine accepters

Languages Accepted by Linear Bounded Automata

- It can be shown that any context-free language can be accepted by a linear bounded automaton
- In addition, linear bounded automata can be designed to accept languages which are not context-free, such as

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

- While it is difficult to come up with a concrete and explicitly defined language to use as an example, linear bounded automata are not as powerful as standard Turing machines