CS 2336: Discrete Mathematics

Chapter 6

Languages: Finite State Machines

(Overview)

Instructor: Cheng-Hsin Hsu

Outline

6.1 Language: The Set Theory of Strings

6.2 Finite State Machines: A First Encounter

6.3 Finite State Machines: A Second Encounter

Powers of an Alphabet

- Alphabet Σ is a finite set of symbols
  - Conventionally, we do not list symbols that can be formed from other symbols!

- For a positive integer *n*, power of Σ is defined as:
  - $\Sigma^1 = \Sigma$
  - $\Sigma^{n+1} = \{xy | x \in \Sigma, y \in \Sigma^n\}$, where *xy* denotes the juxtaposition of *x* and *y*

- In general, $|\Sigma^n| = |\Sigma|^n$

Empty String and Words

- For an alphabet Σ, we let $\Sigma^0 = \{\lambda\}$, where λ is the empty string, which is the string contains no symbol from Σ

- Words (or sentences):
  - $\Sigma^+ = \bigcup_{n=1}^{\infty} \Sigma^n = \bigcup_{n \in \mathbb{Z}^+} \Sigma^n$
  - $\Sigma^* = \bigcup_{n=0}^{\infty} \Sigma^n$

- Ex 6.2: (a) $\Sigma = \{0, 1\}$, (b) $\Sigma = \{\beta, 0, 1, 2, \ldots, 9, + - \times, /, (,)\}$

Equal and Length

- Equal: If $w_1, w_2 \in \Sigma^+$, where $w_1 = x_1 x_2 \cdots x_m$ and $w_2 = y_1 y_2 \cdots y_m$, $w_1 = w_2$ if $m = n$ and $x_i = y_i$ for all i

- Let $w = x_1 x_2 \cdots x_n \in \Sigma^+$. We define the length of w to be n, and is denoted by $\| w \|$
  - $\| \lambda \| = 0$

Concatenation

- Concatenation: For $x = x_1x_2\cdots x_m$ and $y = y_1y_2\cdots y_n$, the concatenation of *x* and *y*, written as *xy*, is the string $x_1x_2\cdots x_my_1y_2\cdots y_n$
  - $\lambda x_1x_2\cdots x_m = x_1x_2\cdots x_m = x$
  - What is λ ?
  - $\lambda\lambda = \lambda$
- Power of *x*, $x^0 = \lambda, x^1 = x, x^2 = xx, x^3 = xx^2, \ldots$
  - $x^{n+1} = ?$

Prefix and Suffix

- For $x, y \in \Sigma^*, w = xy$
  - x is a prefix of w
  - x is a proper prefix of w, if y is not the empty string
  - y is a suffix of w
  - y is a proper suffix of w if x is not the empty string

- If $w=xyz$, then y is called a substring of w. If one of x and y is not the empty string, then y is a proper substring

Language

- For a given alphabet Σ, any subset of Σ^* is called a language over Σ
  - Including $\varnothing$, which is called empty language

- Ex 6.8: Give examples of language over $\Sigma = \{0, 1, 2\}$

Concatenation

- For two languages $A, B \subseteq \Sigma^*$, the concatenation of A and B, written as AB, is $\{ab | a \in A, b \in B\}$

- Note: We skip a few theorems in this section, interested readers are referred to the textbook

Outline

6.1 Language: The Set Theory of Strings

6.2 Finite State Machines: A First Encounter

6.3 Finite State Machines: A Second Encounter

A Vending Machine

Table 6.1

	t_0	t_1	t_2	t_3	t_4
State	(1) s_0	(4) s_1 (5¢)	(7) s_2 (10¢)	(10) s_3 (20¢)	(13) s_0
Input	(2) 5¢	(5) 5¢	(8) 10¢	(11) W	
Output	(3) Nothing	(6) Nothing	(9) Nothing	(12) P	

The numbers (1), (2), . . . , (12), (13) in this table indicate the order of events in the purchase of Mary Jo's package of peppermint chewing gum. For each input at time t_i, $0 \le i \le 3$, there is *at that time* a corresponding output and then a change in state. The new state at time t_{i+1} depends on both the input and the (present) state at time t_i.

Table 6.2

	t_0	t_1	t_2
State	(1) s_0	(4) s_3 (20¢)	(7) s_0
Input	(2) 25¢	(5) B	
Output	(3) 5¢ change	(6) S	

Finite State Machine

- A finite state machine is a five-tuple $M = (S, \mathscr{S}, \mathscr{O}, \nu, \omega)$
  - S: the set of internal states
  - $\mathscr{S}$: the input alphabet
  - $\mathscr{O}$: the output alphabet
  - $\nu : S \times \mathscr{S} \to S$: the next state function
  - $\omega : S \times \mathscr{S} \to \mathscr{O}$: the output function

State (Transition) Table

Table 6.3

	ν		ω	
	0	1	0	1
s_0	s_0	s_1	0	0
s_1	s_2	s_1	0	0
s_2	s_0	s_1	0	1

Table 6.4

State	s_0	$\nu(s_0, 1) = s_1$	$\nu(s_1, 0) = s_2$	$\nu(s_2, 1) = s_1$	$\nu(s_1, 0) = s_2$
Input	1	0	1	0	0
Output	$\omega(s_0, 1) = 0$	$\omega(s_1, 0) = 0$	$\omega(s_2, 1) = 1$	$\omega(s_1, 0) = 0$	

State Diagram

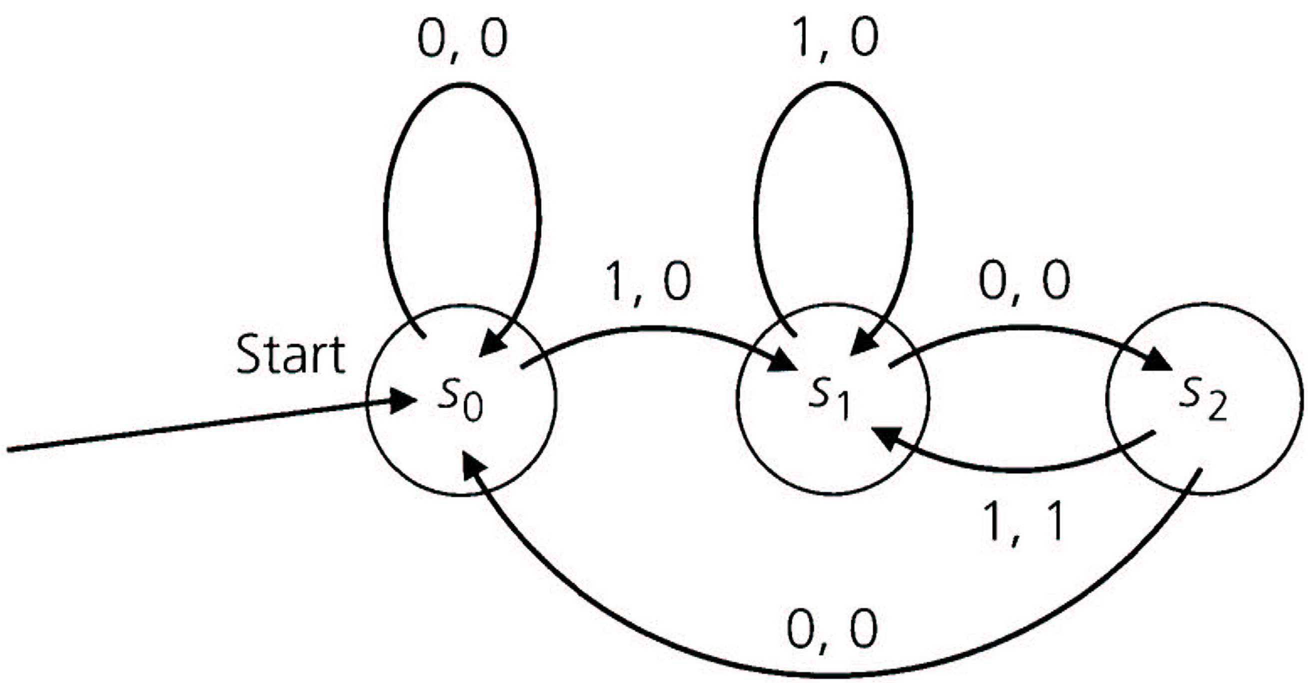

Figure 6.2

Outline

6.1 Language: The Set Theory of Strings

6.2 Finite State Machines: A First Encounter

6.3 Finite State Machines: A Second Encounter

Sequence Recognizer

- Let input and output alphabets be {*0*, *1*}. We want to construct a machine that recognize each occurrence of the sequence *111*

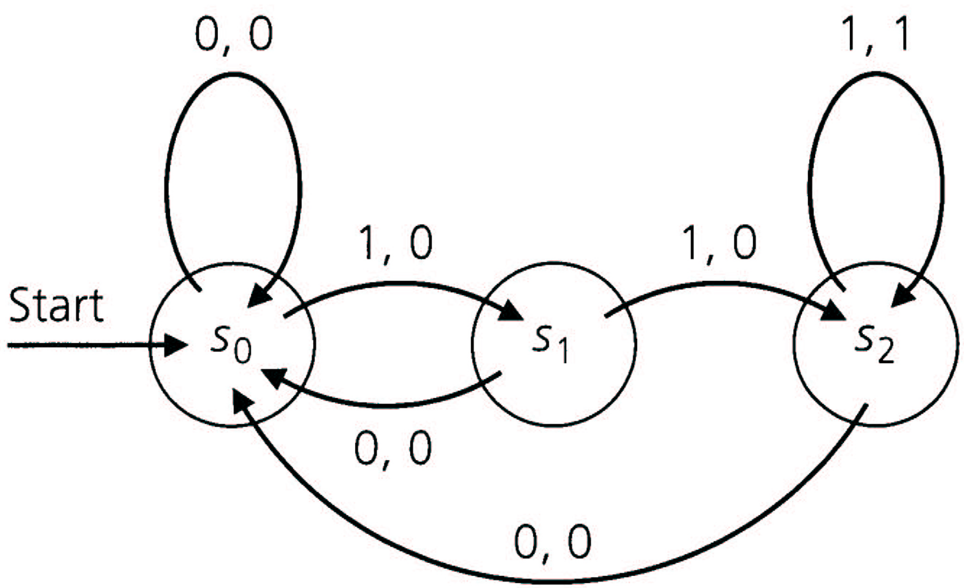

Figure 6.9

Sequence Recognizer (cont.)

- An equivalent state diagram

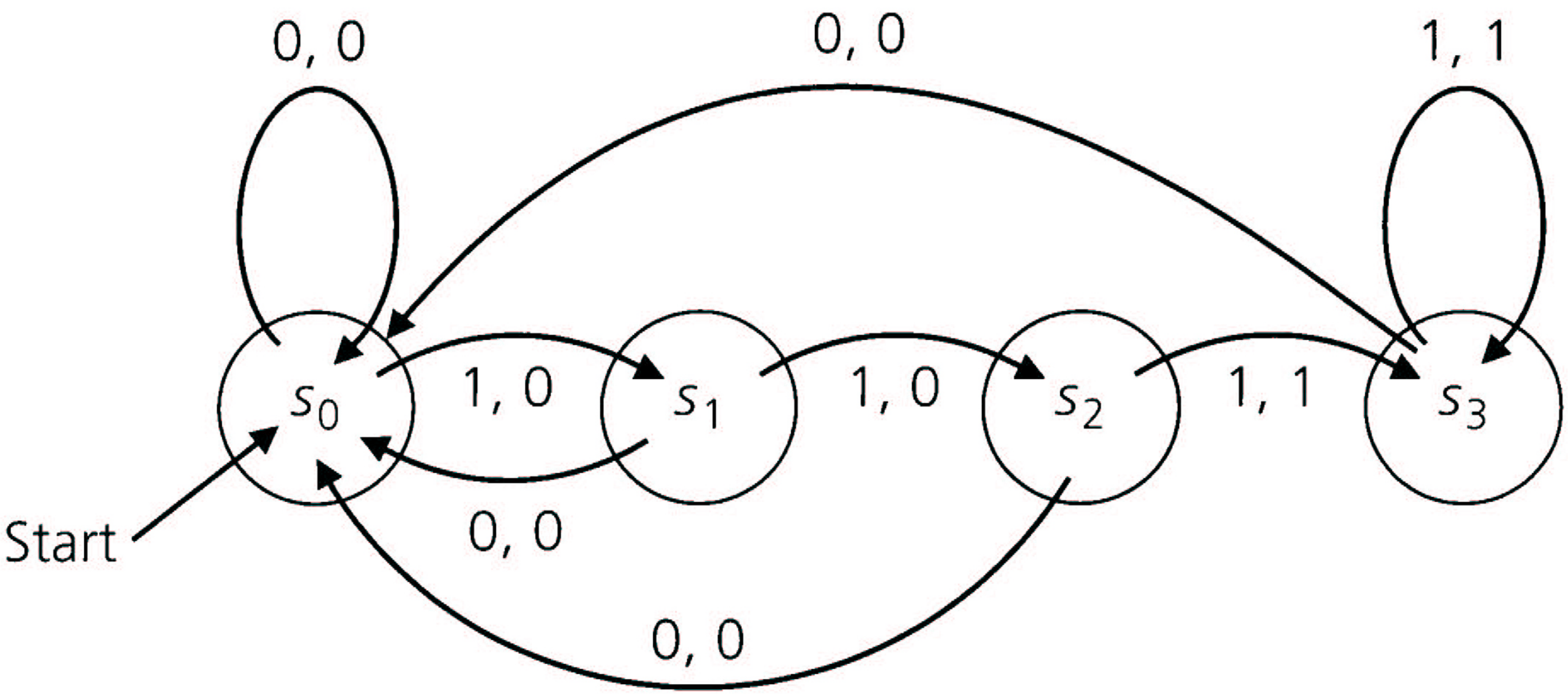

Figure 6.10

A Few Definitions

- Reachable: there is a string x so that $\nu(s_i, x) = s_j$
  - No state is reachable from s_3 except itself
- Transient: there is no string x so that $\nu(s_i, x) = s_i$
  - S_2 is the only transient state
- Sink: if $\nu(s_i, x) = s_i$ for all string x
  - S_3 is a sink
- Submachine and strongly connected

Illustrative Example

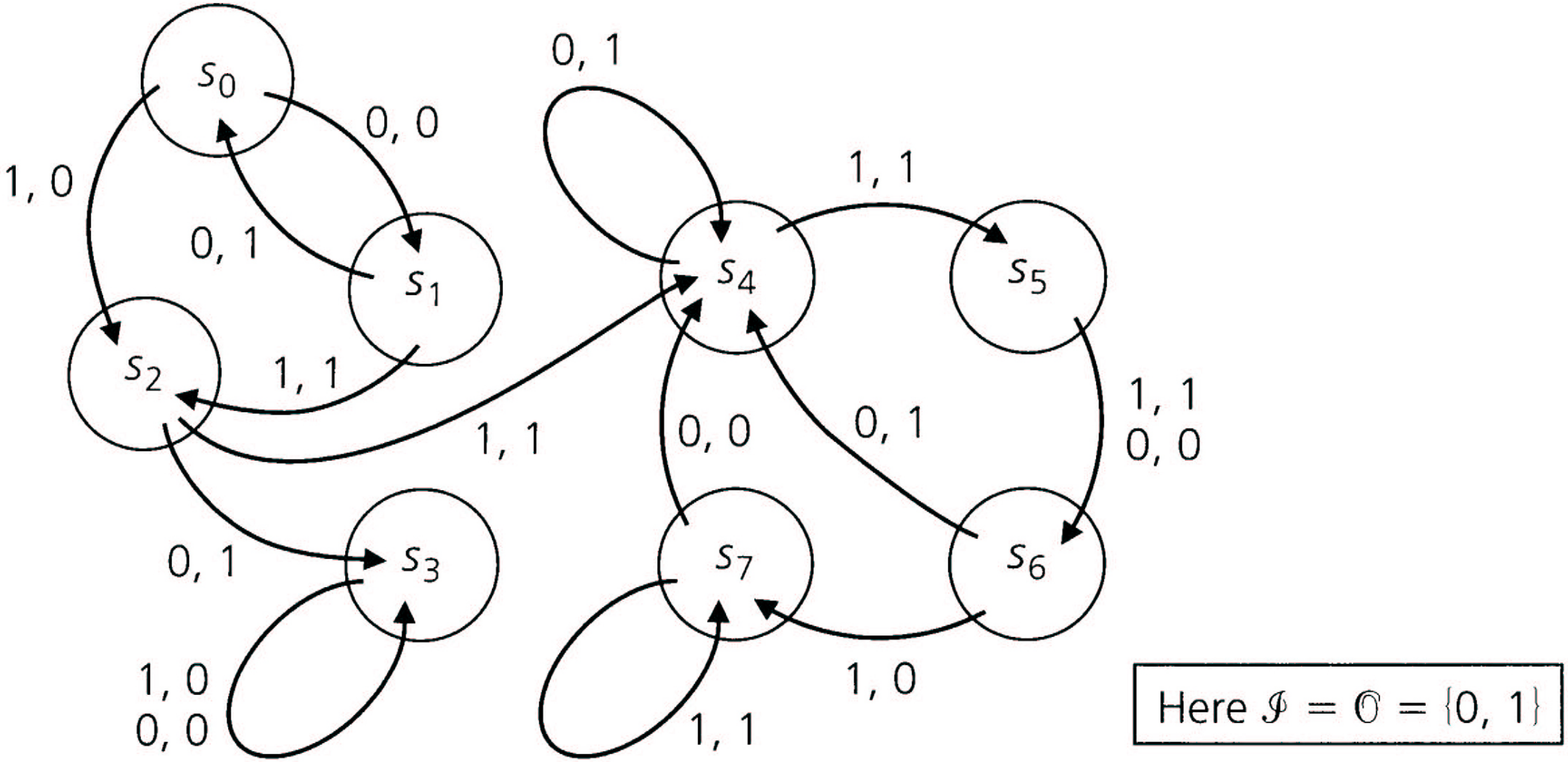

Figure 6.15