

# **MATEMÁTICAS DISCRETA Y COMBINATORIA**

100  
2330

## **Una introducción con aplicaciones**

**TERCERA EDICIÓN**

### **RALPH P. GRIMALDI**

Rose-Hulman Institute of Technology



**Addison-Wesley Iberoamericana**

Argentina • Chile • Colombia • España • Estados Unidos  
México • Perú • Puerto Rico • Venezuela

# 6

---

## Lenguajes: Máquinas de estados finitos

**E**n esta era de computadores y telecomunicaciones, nos enfrentamos día con día a situaciones de entrada y salida. Por ejemplo, al comprar un refresco en una máquina expendedora, damos como *entrada* ciertas monedas y después oprimimos un botón para obtener la *salida* esperada, es decir, el refresco. La primera moneda que damos como entrada pone la máquina en movimiento. Aunque generalmente no nos preocupamos acerca de lo que ocurre dentro de la máquina (a menos que se descomponga y tengamos una pérdida), convendría notar que, de alguna forma, la máquina lleva un registro de las monedas depositadas, hasta que se introduce el importe correcto. Sólo entonces, y no antes, la máquina deja salir el refresco esperado. En consecuencia, para que el vendedor tenga la ganancia esperada por cada refresco, la máquina debe *recordar internamente*, conforme se va insertando cada moneda, la suma de dinero depositado.

Un computador es otro ejemplo de un dispositivo de entrada y salida. En este caso, la entrada es por lo general algún tipo de información y la salida es el resultado obtenido después de procesar esta información. La forma en que se procesa la entrada depende de la labor interna del computador, el cual debe tener la capacidad de recordar la información anterior cuando trabaja con nueva información.

Con los conceptos desarrollados antes acerca de los conjuntos y las funciones, en este capítulo estudiaremos un modelo abstracto llamado *máquina de estados finitos*, o *circuito secuencial*. Estos circuitos son uno de los dos tipos básicos de circuitos de control que se encuentran en los computadores digitales. (El otro tipo es el *circuito combinatorio* o *red de puertas*, que veremos en el capítulo 15.) También se encuentran en otros sistemas, como la máquina expendedora, o en los controles de los ascensores o los sistemas de semáforos.

Como su nombre lo indica, una máquina de estados finitos tiene un número finito de estados internos, en que la máquina recuerda cierta información cuando está en un estado particular. Sin embargo, antes de pasar a este concepto necesitamos material de la teoría de conjuntos para hablar de la entrada válida para dicha máquina.

## 6.1

### Lenguaje: La teoría de conjuntos de las cadenas

Las sucesiones de símbolos, o caracteres, tienen un papel clave en el procesamiento de información por parte de un computador. Como los programas de computación pueden representarse en términos de sucesiones finitas de caracteres, se necesita alguna forma algebraica para manejar tales sucesiones finitas, o *cadenas*.

En esta sección utilizaremos  $\Sigma$  para designar un conjunto de símbolos *finito* distinto del vacío, llamado colectivamente *alfabeto*. Por ejemplo,  $\Sigma = \{0, 1\}$  o  $\Sigma = \{a, b, c, d, e\}$ .

En cualquier alfabeto  $\Sigma$ , no enumeramos los elementos que puedan formarse mediante otros elementos de  $\Sigma$  por yuxtaposición (es decir, si  $a, b \in \Sigma$ , entonces la cadena  $ab$  es la yuxtaposición de los símbolos  $a$  y  $b$ ). Como resultado de esta convención, haremos a un lado los alfabetos como  $\Sigma = \{0, 1, 2, 11, 12\}$  y  $\Sigma = \{a, b, c, ba, aa\}$ . (Además, este convenio nos ayudará en la definición 6.5, cuando hablaremos de la longitud de una cadena.)

Usando un alfabeto  $\Sigma$  como punto de partida podemos construir cadenas con base en los símbolos de  $\Sigma$ , mediante la siguiente idea.

#### Definición 6.1

Si  $\Sigma$  es un alfabeto y  $n \in \mathbb{Z}^+$ , definimos las *potencias de  $\Sigma$*  recursivamente de la siguiente manera:

- 1)  $\Sigma^1 = \Sigma$ , y
- 2)  $\Sigma^{n+1} = \{xy \mid x \in \Sigma, y \in \Sigma^n\}$ , donde  $xy$  denota la yuxtaposición de  $x$  y  $y$ .

#### Ejemplo 6.1

Sea  $\Sigma$  cualquier alfabeto.

Si  $n = 2$ , entonces  $\Sigma^2 = \{xy \mid x, y \in \Sigma\}$ . Por ejemplo, si  $\Sigma = \{0, 1\}$ , encontraremos que  $\Sigma^2 = \{00, 01, 10, 11\}$ .

Cuando  $n = 3$ , los elementos de  $\Sigma^3$  tienen la forma  $uvv$ , donde  $u \in \Sigma$  e  $v \in \Sigma^2$ . Pero como conocemos la forma de los elementos de  $\Sigma^2$ , también podemos considerar que las cadenas en  $\Sigma^3$  son sucesiones de la forma  $uxy$ , donde  $u, x, y \in \Sigma$ . Como un ejemplo para este caso, supongamos que  $\Sigma = \{a, b, c, d, e\}$ . Entonces  $\Sigma^3$  contiene  $5^3 = 125$  cadenas de tres símbolos, entre ellos *aaa*, *acb*, *ace*, *cdd* y *eda*.

En general, para todo  $n \in \mathbb{Z}^+$  encontramos que  $|\Sigma^n| = |\Sigma|^n$ , ya que estamos trabajando con disposiciones (de tamaño  $n$ ) en las que está permitido repetir cualquiera de los  $|\Sigma|$  objetos.

Ahora que hemos examinado  $\Sigma^n$  para  $n \in \mathbb{Z}^+$ , veremos una potencia más de  $\Sigma$ .

#### Definición 6.2

Para cualquier alfabeto  $\Sigma$  definimos  $\Sigma^0 = \{\lambda\}$ , donde  $\lambda$  denota la *cadena vacía*, es decir, la cadena que no consta de *ningún* símbolo tomado de  $\Sigma$ .

El símbolo  $\lambda$  nunca es un elemento de nuestro alfabeto  $\Sigma$ , y no debemos confundirlo con el espacio en blanco que se encuentra en muchos alfabetos.

Sin embargo, aunque  $\lambda \notin \Sigma$ , ocurre que  $\emptyset \subseteq \Sigma$ , por lo que en este caso es necesario ser cuidadosos. Observamos que

- 1)  $\{\lambda\} \not\subseteq \Sigma$  ya que  $\lambda \notin \Sigma$ , y
- 2)  $\{\lambda\} \neq \emptyset$  ya que  $|\{\lambda\}| = 1 \neq 0 = |\emptyset|$ .

Para hablar colectivamente de los conjuntos  $\Sigma^0, \Sigma^1, \Sigma^2, \dots$ , presentaremos la siguiente notación de las uniones de tales conjuntos.

**Definición 6.3** Si  $\Sigma$  es cualquier alfabeto, entonces

- a)  $\Sigma^+ = \bigcup_{n=1}^{\infty} \Sigma^n = \bigcup_{n \in \mathbb{Z}^+} \Sigma^n$ , y
- b)  $\Sigma^* = \bigcup_{n=0}^{\infty} \Sigma^n$ .

Vemos que la única diferencia entre los conjuntos  $\Sigma^+$  y  $\Sigma^*$  es la presencia del elemento  $\lambda$ , ya que  $\lambda \in \Sigma^n$  sólo cuando  $n = 0$ . Además,  $\Sigma^* = \Sigma^+ \cup \Sigma^0$ .

Además de usar el término *cadena*, también nos referiremos a los elementos de  $\Sigma^*$  o  $\Sigma^+$  como *palabras* y a veces como *enunciados*. Para  $\Sigma = \{0, 1, 2\}$ , encontramos palabras como 0, 01, 102 y 1112 tanto en  $\Sigma^+$  como en  $\Sigma^*$ .

Por último, observemos que aunque los conjuntos  $\Sigma^+$  y  $\Sigma^*$  son *infinitos*, los elementos de estos conjuntos son cadenas *finitas* de símbolos.

### Ejemplo 6.2

Para  $\Sigma = \{0, 1\}$ , el conjunto  $\Sigma^*$  consta de todas las cadenas finitas de ceros y unos junto con la cadena vacía. Para un  $n$  razonablemente pequeño, sí podríamos enumerar todas las cadenas de  $\Sigma^n$ .

Si  $\Sigma = \{\beta, 0, 1, 2, \dots, 9, +, -, \times, /, (, )\}$ , donde  $\beta$  denota el espacio en blanco, es más difícil describir  $\Sigma^*$ ; para  $n > 2$ ,  $\Sigma^n$  tiene demasiadas cadenas para enumerarlas. En  $\Sigma^*$  encontramos expresiones aritméticas que nos son familiares, como  $(7 + 5)/(2 \times (3 - 10))$ , así como disparates de tipo  $+)(7/\times + 3/($ .

Ahora nos enfrentamos a una situación conocida. Como en el caso de las proposiciones (Cap. 2), los conjuntos (Cap. 3) y las funciones (Cap. 5), una vez más necesitamos poder decidir cuándo debe considerarse que dos objetos bajo estudio, en este caso cadenas, son iguales. Analizaremos este asunto a continuación.

### Definición 6.4

Si  $w_1, w_2 \in \Sigma^+$ , entonces podemos escribir  $w_1 = x_1, x_2 \dots x_m$  y  $w_2 = y_1, y_2 \dots y_n$ , para  $m, n \in \mathbb{Z}^+$ , y  $x_1, x_2 \dots x_m, y_1, y_2 \dots y_n \in \Sigma$ . Decimos que las cadenas  $w_1$  y  $w_2$  son *iguales*, y escribimos  $w_1 = w_2$ , si  $m = n$ , y  $x_i = y_i$  para todo  $1 \leq i \leq m$ .

Se sigue de esta definición que dos cadenas de  $\Sigma^+$  son *iguales* sólo cuando cada una está formada por el mismo número de símbolos de  $\Sigma$  y los símbolos correspondientes en las dos cadenas coinciden idénticamente.

También necesitamos el número de símbolos de una cadena para definir otra de sus propiedades.

**Definición 6.5**

Si  $w \in \Sigma^+$ , entonces  $w = x_1 x_2 \dots x_n$ , donde  $x_i \in \Sigma$  para todo  $1 \leq i \leq n$ . Definimos la *longitud* de  $w$ , que se denota con  $\|w\|$ , como el valor  $n$ . Para el caso de  $\lambda$ , tenemos que  $\|\lambda\| = 0$ .

Como resultado de la definición 6.5, encontramos que para cualquier alfabeto  $\Sigma$ , si  $w \in \Sigma^*$  y  $\|w\| \geq 1$ , entonces  $w \in \Sigma^+$ , y viceversa. Además, para cualquier  $y \in \Sigma^*$ ,  $\|y\| = 1$  si y sólo si  $y \in \Sigma$ . Si  $\Sigma$  contiene el símbolo  $\beta$  (para el espacio en blanco), también ocurre que  $\|\beta\| = 1$ .

Si utilizamos un alfabeto particular, como  $\Sigma = \{0, 1, 2\}$ , y examinamos los elementos  $x = 01$ ,  $y = 212$  y  $z = 01212$  (de  $\Sigma^*$ ), encontramos que

$$\|z\| = \|01212\| = 5 = 2 + 3 = \|01\| + \|212\| = \|x\| + \|y\|.$$

Para continuar nuestro estudio de las propiedades de las cadenas y los alfabetos, necesitamos ampliar la idea de yuxtaposición.

**Definición 6.6**

Sean  $x, y \in \Sigma^*$  con  $x = x_1 x_2 \dots x_m$  y  $y = y_1 y_2 \dots y_n$ , tales que cada  $x_i$ , para  $1 \leq i \leq m$ , y cada  $y_j$ , para  $1 \leq j \leq n$ , está en  $\Sigma$ . La *concatenación* de  $x$  y  $y$ , que escribimos como  $xy$ , es la cadena  $x_1 x_2 \dots x_m y_1 y_2 \dots y_n$ .

La *concatenación* de  $x$  y  $\lambda$  es

$$x\lambda = x_1 x_2 \dots x_m \lambda = x_1 x_2 \dots x_m = x,$$

y la *concatenación* de  $\lambda$  y  $x$  es

$$\lambda x = \lambda x_1 x_2 \dots x_m = x_1 x_2 \dots x_m = x.$$

Finalmente, la *concatenación* de  $\lambda$  y  $\lambda$  es  $\lambda\lambda = \lambda$ .

Arriba hemos definido una operación binaria cerrada en  $\Sigma^*$  (y  $\Sigma^+$ ). Esta operación es asociativa pero no comutativa (a menos que  $|\Sigma| = 1$ ), y como  $x\lambda = \lambda x = x$  para cualquier  $x \in \Sigma^*$ , el elemento  $\lambda \in \Sigma^*$  es el neutro para la operación de concatenación. Las ideas de las últimas dos definiciones (la longitud de una cadena y la operación de concatenación) están relacionadas con el resultado

$$\|xy\| = \|x\| + \|y\|, \text{ para cualesquiera } x, y \in \Sigma^*,$$

de donde obtenemos el caso especial

$$\|x\| = \|x\| + 0 = \|x\| + \|\lambda\| = \|x\lambda\| (= \|\lambda x\|).$$

Por último, para cualquier  $z \in S$ , tenemos que  $\|z\| = \|\lambda z\| = \|\lambda z\| = 1$ , mientras que  $\|zz\| = 2$ .

La operación binaria cerrada de concatenación nos permite plantear ahora otra definición recursiva. Ya definimos las potencias de un alfabeto  $\Sigma$ . Ahora examinaremos las potencias de las cadenas.

**Definición 6.7**

Para cualquier  $x \in \Sigma^*$ , definimos las *potencias* de  $x$  como  $x^0 = \lambda$ ,  $x^1 = x$ ,  $x^2 = xx$ ,  $x^3 = xx^2$ ,  $\dots$ ,  $x^{n+1} = xx^n$ ,  $\dots$ , donde  $n \in \mathbb{N}$ .

Esta definición es otro ejemplo de definición recursiva de una entidad matemática: la entidad matemática que buscamos en este momento se obtiene a partir de entidades (comparables) previamente obtenidas. Además, la definición nos proporciona un camino para trabajar con una concatenación de  $n$ -partes (como una potencia  $(n + 1)$ ) de una cadena consigo misma e incluye el caso especial en que la cadena es sólo un símbolo.

**Ejemplo 6.3**

Si  $\Sigma = \{0, 1\}$  y  $x = 01$ , entonces  $x^0 = \lambda$ ,  $x^1 = 01$ ,  $x^2 = 0101$  y  $x^3 = 010101$ . Para cualquier  $n > 0$ ,  $x^n$  consta de una cadena de  $n$  ceros y  $n$  unos, donde el primer símbolo es 0 y los símbolos posteriores se alternan. En este caso,  $\|x^2\| = 4 = 2\|x\|$ ,  $\|x^3\| = 6 = 3\|x\|$ , y para todo  $n \in \mathbb{N}$ ,  $\|x^n\| = n\|x\|$ .

Ahora estamos listos para enfrentar el tema principal de esta sección, el concepto de lenguaje. Sin embargo, antes de hacerlo necesitamos saber más acerca de otras tres ideas. Estas ideas se refieren a subsecciones especiales de cadenas.

**Definición 6.8**

Si  $x, y \in \Sigma^*$  y  $w = xy$ , entonces la cadena  $x$  es un *prefijo* de  $w$ , y si  $y \neq \lambda$ , entonces  $x$  es un *prefijo propio*. De manera similar, la cadena  $y$  es un *sufijo* de  $w$  y es un *sufijo propio* cuando  $x \neq \lambda$ .

**Ejemplo 6.4**

Sea  $\Sigma = \{a, b, c\}$ , y consideremos la cadena  $w = abbcc$ . Entonces cada una de las cadenas  $\lambda$ ,  $a$ ,  $ab$ ,  $abb$ ,  $abbc$ , y  $abbcc$  es un prefijo de  $w$ , y, excepto la misma  $abbcc$ , cada una es un prefijo propio. Por otro lado, cada una de las cadenas  $\lambda$ ,  $c$ ,  $cc$ ,  $bcc$ ,  $bbcc$  y  $abbcc$  es un sufijo de  $w$ , y las primeras cinco cadenas son sufijos propios.

En general, para cualquier alfabeto  $\Sigma$ , si  $n \in \mathbb{Z}^+$  y  $x_i \in \Sigma$  para todo  $1 \leq i \leq n$ , entonces cada una de las cadenas  $\lambda$ ,  $x_1$ ,  $x_1x_2$ ,  $x_1x_2x_3$ , ..., y  $x_1x_2x_3\dots x_n$  es un prefijo de la cadena  $x = x_1x_2x_3\dots x_n$ . Por otro lado,  $\lambda$ ,  $x_n$ ,  $x_{n-1}x_n$ ,  $x_{n-2}x_{n-1}x_n$ , ..., y  $x_1x_2x_3\dots x_n$  son sufijos de esta cadena. Así,  $x$  tiene  $n + 1$  prefijos,  $n$  de los cuales son propios; la situación es la misma para los sufijos.

**Ejemplo 6.5**

Si  $\|x\| = 5$ ,  $\|y\| = 4$  y  $w = xy$ , entonces  $w$  tiene a  $x$  como un prefijo propio y a  $y$  como un sufijo propio. En total,  $w$  tiene nueve prefijos propios y nueve sufijos propios, puesto que  $\lambda$  es un prefijo propio y un sufijo propio para cada cadena de  $\Sigma^*$ . En este caso,  $xy$  es un prefijo y un sufijo, pero en ningún caso es propio.

**Ejemplo 6.6**

Para un alfabeto dado  $\Sigma$ , sean  $w, a, b, c, d \in \Sigma^*$ . Si  $w = ab = cd$ , entonces

- $a$  es un prefijo de  $c$ , o  $c$  es un prefijo de  $a$ ; y,
- $b$  es un sufijo de  $d$ , o  $d$  es un sufijo de  $b$ .

**Definición 6.9**

Si  $x, y, z \in \Sigma^*$  y  $w = xyz$ , entonces  $y$  es una *subcadena* de  $w$ . Cuando  $x$  o  $z$  es diferente de  $\lambda$  (de modo que  $y$  es diferente de  $w$ ),  $y$  es una *subcadena propia*.

**Ejemplo 6.7**

Para  $\Sigma = \{0, 1\}$ , sea  $w = 00101110 \in \Sigma^*$ . Las siguientes son subcadenas de  $w$ :

- 1) 1011: Esto surge solamente de una forma: cuando  $w = xyz$ , con  $x = 00$ ,  $y = 1011$  y  $z = 10$ .
- 2) 10: Este ejemplo surge de dos formas:
  - a)  $w = xyz$  donde  $x = 00$ ,  $y = 10$  y  $z = 1110$ , y
  - b)  $w = xyz$  para  $x = 001011$ ,  $y = 10$  y  $z = \lambda$ .

En el caso (b) la subcadena también es un sufijo (propio) de  $w$ .

Ahora que estamos familiarizados con las definiciones necesarias, es hora de pensar en el concepto de lenguaje.

Cuando consideramos el alfabeto común, incluyendo el espacio en blanco, muchas cadenas como *qxo*, *el wxx rojo atzl*, y *aeytl* no representan palabras o partes de enunciados del español, aunque sean elementos de  $\Sigma^*$ . En consecuencia, para considerar solamente las palabras y expresiones que tienen sentido en el idioma español, nos concentraremos en un subconjunto de  $\Sigma^*$ . Esto nos lleva a la siguiente generalización.

**Definición 6.10**

Para un alfabeto dado  $\Sigma$ , cualquier subconjunto de  $\Sigma^*$  es un *lenguaje* sobre  $\Sigma$ . Esto incluye el subconjunto  $\emptyset$ , al que llamaremos *lenguaje vacío*.

**Ejemplo 6.8**

Si  $\Sigma = \{0, 1\}$ , los conjuntos  $A = \{0, 01, 001\}$  y  $B = \{0, 01, 001, 0001, \dots\}$  son ejemplos de lenguajes sobre  $\Sigma$ .

**Ejemplo 6.9**

Si  $\Sigma$  es el alfabeto de 26 letras, 10 dígitos y los símbolos especiales utilizados en una implementación dada de Pascal, la colección de programas ejecutables para esa implementación constituye un lenguaje. En la misma situación, cada programa ejecutable podría ser considerado un lenguaje, como podría serlo también un conjunto finito particular de tales programas.

Ya que los lenguajes son conjuntos, podemos formar la unión, intersección y diferencia simétrica de dos lenguajes. Sin embargo, para poder trabajar, es más útil una extensión de la operación binaria cerrada definida (en la Def. 6.6) para cadenas.

**Definición 6.11**

Para un alfabeto  $\Sigma$  y los lenguajes  $A, B \subseteq \Sigma^*$ , la *concatenación* de  $A$  y  $B$ , denotada con  $AB$ , es  $\{ab \mid a \in A, b \in B\}$ .

Podríamos comparar la concatenación con el producto cartesiano; veríamos que, al igual que  $A \times B \neq B \times A$  en general, también  $AB \neq BA$  en general. Para  $A, B$  finitos, teníamos que  $|A \times B| = |B \times A|$ , pero aquí puede ocurrir que  $|AB| \neq |BA|$  incluso para lenguajes finitos.

**Ejemplo 6.10**

Sean  $\Sigma = \{x, y, z\}$ , y sean  $A, B$  los lenguajes finitos  $A = \{x, xy, z\}$ ,  $B = \{\lambda, y\}$ . Entonces  $AB = \{x, xy, z, xyy, zy\}$  y  $BA = \{x, xy, z, yx, yxy, yz\}$ , por lo que

- 1)  $|AB| = 5 \neq 6 = |BA|$ , y
- 2)  $|AB| = 5 \neq 6 = 3 \cdot 2 = |A||B|$ .

Las diferencias surgen debido a que hay dos formas de representar  $xy$ : (1)  $xy$  para  $x \in A, y \in B$  y (2)  $xy\lambda$  donde  $xy \in A$  y  $\lambda \in B$ . (El concepto de unicidad de la representación es algo que no podemos dar por hecho. Aunque no se cumple en este caso, es la clave del éxito de muchas ideas matemáticas, como lo vimos en el teorema fundamental de la aritmética (Teorema 4.11) y como lo veremos de nuevo en el capítulo 15.)

El ejemplo anterior sugiere que para lenguajes finitos  $A$  y  $B$ ,  $|AB| \leq |A||B|$ . Podemos mostrar que esto es cierto en general.

El siguiente teorema analiza algunas propiedades satisfechas por la concatenación de lenguajes.

**TEOREMA 6.1**

Para un alfabeto  $\Sigma$ , sean  $A, B, C \subseteq \Sigma^*$ . Entonces

- |                                       |                                       |
|---------------------------------------|---------------------------------------|
| a) $A\{\lambda\} = \{\lambda\}A = A$  | b) $(AB)C = A(BC)$                    |
| c) $A(B \cup C) = AB \cup AC$         | d) $(B \cup C)A = BA \cup CA$         |
| e) $A(B \cap C) \subseteq AB \cap AC$ | f) $(B \cap C)A \subseteq BA \cap CA$ |

**Demostración:** Nos ocuparemos de las partes (d) y (f) y dejaremos las demás para el lector.

(d) Como tratamos de demostrar que dos conjuntos son iguales, de nuevo utilizaremos la idea de igualdad de conjuntos de la definición 3.2. Si partimos de un elemento  $x \in \Sigma^*$ , tenemos que  $x \in (B \cup C)A \Rightarrow x = yz$  para  $y \in B \cup C$  y  $z \in A \Rightarrow (x = yz \text{ para } y \in B, z \in A) \text{ o } (x = yz \text{ para } y \in C, z \in A) \Rightarrow x \in BA \text{ o } x \in CA$ , por lo cual  $(B \cup C)A \subseteq BA \cup CA$ . Recíprocamente, si  $x \in BA \cup CA \Rightarrow x \in BA \text{ o } x \in CA \Rightarrow x = ba_1$  donde  $b \in B$  y  $a_1 \in A$ , o  $x = ca_2$  donde  $c \in C$  y  $a_2 \in A$ . Supongamos que  $x = ba_1$  para  $b \in B, a_1 \in A$ . Como  $B \subseteq B \cup C$ , tenemos que  $x = ba_1$ , donde  $b \in B \cup C$  y  $a_1 \in A$ . Entonces  $x \in (B \cup C)A$ , por lo que  $BA \cup CA \subseteq (B \cup C)A$ . (El argumento es similar si  $x = ca_2$ .) Una vez establecidas ambas inclusiones, se sigue que  $(B \cup C)A = BA \cup CA$ .

(f) Para  $x \in \Sigma^*$ , vemos que  $x \in (B \cap C)A \Rightarrow x = yz$  donde  $y \in B \cap C$  y  $z \in A \Rightarrow (x = yz \text{ para } y \in B \text{ y } z \in A) \text{ y } (x = yz \text{ para } y \in C \text{ y } z \in A) \Rightarrow x \in BA \text{ y } x \in CA \Rightarrow x \in BA \cap CA$ , por lo que  $(B \cap C)A \subseteq BA \cap CA$ .

Si  $\Sigma = \{x, y, z\}$ , sean  $B = \{x, xx, y\}$ ,  $C = \{y, xy\}$  y  $A = \{y, yy\}$ . Entonces  $xyy \in BA \cap CA$ , pero  $xyy \notin (B \cap C)A$ . En consecuencia,  $(B \cap C)A \subset BA \cap CA$ .

Análogamente a los conceptos de  $\Sigma^n$ ,  $\Sigma^*$ ,  $\Sigma^+$ , damos las siguientes definiciones para un lenguaje arbitrario  $A \subseteq \Sigma^*$ .

**Definición 6.12** Para cualquier lenguaje  $A \subseteq \Sigma^*$  podemos construir otros lenguajes de la manera siguiente:

- $A^0 = \{\lambda\}$ ,  $A^1 = A$  y para cualquier  $n \in \mathbb{Z}^+$ ,  $A^{n+1} = \{ab \mid a \in A, b \in A^n\}$
- $A^+ = \bigcup_{n \in \mathbb{Z}^+} A^n$ , la *clausura positiva* de  $A$
- $A^* = A^+ \cup \{\lambda\}$ . El lenguaje  $A^*$  es la *clausura de Kleene* de  $A$ , en honor del lógico norteamericano Stephen Cole Kleene (1909– ).

**Ejemplo 6.11**

Si  $\Sigma = \{x, y, z\}$  y  $A = \{x\}$ , entonces (1)  $A^0 = \{\lambda\}$ ; (2)  $A^n = \{x^n\}$  para  $n \in \mathbb{N}$ ; (3)  $A^+ = \{x^n \mid n \geq 1\}$  y (4)  $A^* = \{x^n \mid n \geq 0\}$ .

**Ejemplo 6.12**

Sea  $\Sigma = \{x, y\}$ .

- Si  $A = \{xx, xy, yx, yy\} = \Sigma^2$ , entonces  $A^*$  es el lenguaje de todas las cadenas  $w \in \Sigma^*$  tales que la longitud de  $w$  es par.
- Con  $A$  como en la parte (a) y  $B = \{x, y\}$ , el lenguaje  $BA^*$  contiene todas las cadenas de  $\Sigma^*$  de longitud impar y también vemos que, en este caso,  $BA^* = A^*B$  y que  $\Sigma^* = A^* \cup BA^*$ .
- El lenguaje  $\{x\}\{x, y\}^*$  (la concatenación de los lenguajes  $\{x\}$  y  $\{x, y\}^*$ ) contiene cada cadena de  $\Sigma^*$  para la cual  $x$  es un prefijo. El lenguaje que contiene todas las cadenas de  $\Sigma^*$  para las cuales  $yy$  es un sufijo puede definirse como  $\{x, y\}^*\{yy\}$ . Cada cadena del lenguaje  $\{x, y\}^*\{xx\}\{x, y\}^*$  tiene a  $xx$  como una subcadena.
- Cualquier cadena del lenguaje  $\{x\}^*\{y\}^*$  consta de un número finito (tal vez igual a 0) de  $x$  seguido por un número finito (que también puede ser cero) de  $y$ . Y aunque  $\{x\}^*\{y\}^* \subseteq \{x, y\}^*$ , la cadena  $w = xyx$  está en  $\{x, y\}^*$  pero no en  $\{x\}^*\{y\}^*$ . Por lo tanto,  $\{x\}^*\{y\}^* \subset \{x, y\}^*$ .

**Ejemplo 6.13**

En el álgebra de los números reales, si  $a, b \in \mathbb{R}$  y  $a, b > 0$ , entonces  $a^2 = b^2 \Rightarrow a = b$ . Sin embargo, en el caso de los lenguajes, si  $\Sigma = \{x, y\}$ ,  $A = \{\lambda, x, x^3, x^4, \dots\} = \{x^n \mid n \geq 0\} - \{x^2\}$  y  $B = \{x^n \mid n \geq 0\}$ , entonces  $A^2 = B^2 (= B)$ , pero  $A \neq B$ . (Nota: Nunca ocurre que  $\lambda \in \Sigma$ , pero es posible que  $\lambda \in A \subseteq \Sigma^*$ .)

Continuaremos esta sección con un lema y un segundo teorema que tratan acerca de las propiedades de los lenguajes. El lema proporciona otra situación en que utilizamos el principio de inducción matemática.

**EMA 6.1**

Sea  $\Sigma$  un alfabeto, con lenguajes  $A, B \subseteq \Sigma^*$ . Si  $A \subseteq B$ , entonces  $A^n \subseteq B^n$  para todo  $n \in \mathbb{Z}^+$ .

**Demostración:** Consideremos la proposición abierta  $S(n): A \subseteq B \Rightarrow A^n \subseteq B^n$ . Como  $A^1 = A \subseteq B = B^1$ , vemos que  $S(1)$  es la proposición:  $A \subseteq B \Rightarrow A \subseteq B$ , por lo que el resultado es cierto en este primer caso. Si suponemos que  $S(k)$  es verdadero, tenemos que  $A \subseteq B \Rightarrow A^k \subseteq B^k$ . De la hipótesis de  $S(k+1)$  sabemos que  $A \subseteq B$  y de  $S_k$  tenemos  $A^k \subseteq B^k$  (por *Modus Ponens*). Consideremos ahora una cadena  $x$  de  $A^{k+1}$ . Entonces, de la definición 6.12, sabemos que  $x = x_1x_k$ , donde  $x_1 \in A$ ,  $x_k \in A^k$ . Como  $A \subseteq B$  y  $A^k \subseteq B^k$  (por la hipótesis de inducción), tenemos que  $x_1 \in B$  y  $x_k \in B^k$ . En consecuencia,  $x = x_1x_k \in BB^k = B^{k+1}$ . Por lo tanto,  $S(k+1)$  es verdadera y, por el principio de inducción finita, se sigue que si  $A \subseteq B$ , entonces  $A^n \subseteq B^n$  para todo  $n \in \mathbb{Z}^+$ .

**TEOREMA 6.2**

Para un alfabeto  $\Sigma$  y los lenguajes  $A, B \subseteq \Sigma^*$ ,

- a)  $A \subseteq AB^*$
- b)  $A \subseteq B^*A$
- c)  $A \subseteq B \Rightarrow A^+ \subseteq B^+$
- d)  $A \subseteq B \Rightarrow A^* \subseteq B^*$
- e)  $AA^* = A^*A = A^+$
- f)  $A^*A^* = A^* = (A^*)^* = (A^*)^+ = (A^+)^*$
- g)  $(A \cup B)^* = (A^* \cup B^*)^* = (A^*B^*)^*$

**Demostración:** Haremos las demostraciones de las partes (c) y (g).

(c) Sea  $A \subseteq B$  y  $x \in A^+$ . Entonces  $x \in A^+ \Rightarrow x \in A^n$ , para algún  $n \in \mathbb{Z}^+$ . Del lema 6.1 se sigue entonces que  $x \in B^n \subseteq B^+$  con lo que hemos mostrado que  $A^+ \subseteq B^+$ .

(g)  $[(A \cup B)^* = (A^* \cup B^*)^*]$ . Sabemos que  $A \subseteq A^*$ ,  $B \subseteq B^* \Rightarrow (A \cup B) \subseteq (A^* \cup B^*) \Rightarrow (A \cup B)^* \subseteq (A^* \cup B^*)^*$  (por la parte d). A la inversa, también vemos que  $A, B \subseteq A \cup B \Rightarrow A^*, B^* \subseteq (A \cup B)^*$  (por la parte d)  $\Rightarrow (A^* \cup B^*) \subseteq (A \cup B)^* \Rightarrow (A^* \cup B^*)^* \subseteq (A \cup B)^*$  (por las partes d y f). De ambas inclusiones se sigue que  $(A \cup B)^* = (A^* \cup B^*)^*$ .

$[(A^* \cup B^*)^* = (A^*B^*)^*]$ . Primero vemos que  $A^*, B^* \subseteq A^*B^*$  (por las partes a y b)  $\Rightarrow (A^* \cup B^*) \subseteq A^*B^* \Rightarrow (A^* \cup B^*)^* \subseteq (A^*B^*)^*$  (por la parte d). Recíprocamente, si  $xy \in A^*B^*$ , donde  $x \in A^*$  y  $y \in B^*$ , entonces  $x, y \in A^* \cup B^*$ , por lo que  $xy \in (A^* \cup B^*)^*$  y  $A^*B^* \subseteq (A^* \cup B^*)^*$ . Usando nuevamente las partes (d) y (f) para mostrar que  $(A^*B^*)^* \subseteq (A^* \cup B^*)^*$ , de donde se sigue el resultado.

Para cerrar esta primera sección analizaremos la idea de un conjunto definido en forma recursiva (dada en la sección 4.2), como lo demuestran los siguientes cuatro ejemplos.

**Ejemplo 6.14**

Para el alfabeto  $\Sigma = \{0, 1\}$  consideremos el lenguaje  $A \subseteq \Sigma^*$ , donde cada palabra de  $A$  contiene solamente una aparición del símbolo 0. Entonces  $A$  es un conjunto infinito y entre las palabras de  $A$  podemos encontrar 0, 01, 10, 01111, 11110111 y 1111111110. También existe una infinidad de palabras de  $\Sigma^*$  que no están en  $A$  (como 1, 11, 00, 000, 010 y 0111111110). Podemos definir este lenguaje  $A$  de manera recursiva, como sigue:

- 1) Nuestro paso inicial indica que  $0 \in A$ ; y
- 2) Para el proceso recursivo queremos incluir en  $A$  las palabras  $1x$  y  $x1$ , para cualquier palabra  $x \in A$ .

Con esta definición, el siguiente análisis muestra que la palabra 1011 está en  $A$ .

De la parte (1) de nuestra definición, sabemos que  $0 \in A$ . Entonces, si aplicamos la parte (2) de nuestra definición tres veces, vemos que:

- i)  $01 \in A$ , puesto que  $0 \in A$ ;
  - ii)  $011 \in A$ , ya que  $01 \in A$ ; y
  - iii) como  $011 \in A$ , tenemos que  $1011 \in A$ .
- 

**Ejemplo 6.15**

Para  $\Sigma = \{(),\}$ , el alfabeto que contiene a los paréntesis izquierdo y derecho, queremos examinar el lenguaje  $A \subseteq \Sigma^*$  que consta de todas las cadenas no vacías de paréntesis que son gramaticalmente correctas como expresiones algebraicas. Entonces tenemos, por ejemplo, las tres cadenas  $(( ))$ ,  $(((( ))))$ ; y  $(()()()$  de este lenguaje, pero no tenemos cadenas como  $(( ))()$ ;  $(())()$ ; o  $)()$  $(( ))$ ). Vemos que para que una cadena  $x$  (distinta de  $\lambda$ ) esté en  $A$ , entonces

- i) debemos tener el mismo número de paréntesis izquierdos que derechos; y
- ii) el número de paréntesis izquierdos (siempre) debe ser mayor o igual que el número de paréntesis derechos, al examinar cada uno de los paréntesis en  $x$ , leyéndolos de izquierda a derecha.

El lenguaje  $A$  puede definirse de manera recursiva como sigue:

- 1)  $()$  está en  $A$ ; y
- 2) Para cualesquiera  $x, y \in A$ , tenemos (i)  $xy \in A$  y (ii)  $(x) \in A$ .

[Como mencionamos antes del ejemplo 4.19, también tenemos una restricción implícita, que ninguna otra cadena de paréntesis está en  $A$ , a menos que se obtenga mediante los pasos (1) y (2) anteriores.]

Con esta definición recursiva mostraremos la forma de establecer que la cadena  $(( )) \in \Sigma^*$  está en el lenguaje  $A$ .

**Pasos**

- 1)  $()$  está en  $A$ .
- 2)  $()()$  está en  $A$ .
- 3)  $(( ))$  está en  $A$ .

**Razones**

- |   |
|---|
| Parte (1) de la definición recursiva    |
| Paso (1) y parte (2i) de la definición  |
| Paso (2) y parte (2ii) de la definición |

**Ejemplo 6.16**

Para  $\Sigma = \{0, 1\}$ , la siguiente es una definición recursiva del lenguaje  $A \subseteq \Sigma^*$  tal que  $x \in A$  si el número de ceros de  $x$  es igual al número de unos.

- 1)  $\lambda \in A$ ; y
- 2) si  $x \in A$ , entonces cada uno de los siguientes también está en  $A$ :

- |           |           |            |
|-----------|-----------|------------|
| i) $01x$  | ii) $x01$ | iii) $0x1$ |
| iv) $10x$ | v) $x10$  | vi) $1x0$  |

(Y ninguna otra cadena de ceros y unos está en  $A$ .)

---

**Ejemplo 6.17**

Dado un alfabeto  $\Sigma$ , consideremos la cadena  $x = x_1 x_2 x_3 \dots x_{n-1} x_n \in \Sigma^*$ , donde  $x_i \in \Sigma$  para cada  $1 \leq i \leq n$  y  $n \in \mathbb{Z}^+$ . El *inverso* de  $x$ , que se denota  $x^R$ , es la cadena que se obtiene de  $x$  leyendo los símbolos (en  $x$ ) de derecha a izquierda; es decir,  $x^R = x_n x_{n-1} \dots x_3 x_2 x_1$ . Por ejemplo, si  $\Sigma = \{0, 1\}$  y  $x = 01101$ , entonces  $x^R = 10110$  y para  $w = 101101$  tenemos que  $w^R = 101101 = w$ . En general, podemos definir el inverso de una cadena (de  $\Sigma^*$ ) de manera recursiva como sigue:

- 1)  $\lambda^R = \lambda$ ; y
- 2) Para cualquier  $n \in \mathbb{N}$ , si  $x \in \Sigma^{n+1}$ , entonces escribimos  $x = zy$  donde  $z \in \Sigma$  y  $y \in \Sigma^n$ , y definimos  $x^R = (zy)^R = (y^R)z$ .

Con esta definición recursiva podemos demostrar que si  $\Sigma$  es cualquier alfabeto y  $x_1, x_2 \in \Sigma^*$ , entonces  $(x_1 x_2)^R = x_2^R x_1^R$ .

**Demostración:** La demostración es por inducción matemática sobre el valor de  $\|x_1\|$ . Si  $\|x_1\| = 0$ , entonces,  $x_1 = \lambda$  y  $(x_1 x_2)^R = (\lambda x_2)^R = x_2^R = x_2^R \lambda = x_2^R \lambda^R = x_2^R x_1^R$  ya que  $\lambda^R = \lambda$  según la parte (1) de la definición recursiva. En consecuencia, el resultado es cierto en este primer caso, el cual constituye el paso inicial. Para el paso inductivo supondremos que el resultado es verdadero para cualesquiera  $y, x_2 \in \Sigma^*$ , donde  $\|y\| = k$  para algún  $k \in \mathbb{N}$ . Consideremos ahora lo que ocurre con  $x_1, x_2 \in \Sigma^*$ , donde  $x_1 = zy_1$ , con  $\|z\| = 1$  y  $\|y_1\| = k$ . Aquí vemos que  $(x_1 x_2)^R = (zy_1 x_2)^R = (y_1 x_2)^R z$  [de la parte (2) de la definición recursiva]  $= x_2^R y_1^R z$  (de la hipótesis de inducción)  $= x_2^R (zy_1)^R$  [de nuevo por la parte (2) de la definición recursiva]  $= x_2^R x_1^R$ . Por lo tanto, el resultado es cierto para todo  $x_1, x_2 \in \Sigma^*$  por el principio de inducción matemática.

**EJERCICIOS 6.1**

1. Sea  $\Sigma = \{a, b, c, d, e\}$ . (a) ¿Cuánto valen  $|\Sigma^2|$  y  $|\Sigma^3|$ ? (b) ¿Cuántas cadenas de  $\Sigma^*$  tienen una longitud de al menos cinco?
2. Para  $\Sigma = \{w, x, y, z\}$ , determine el número de cadenas en  $\Sigma^*$  de longitud cinco (a) que comienzan con  $w$ ; (b) con precisamente dos  $w$ ; (c) sin  $w$ ; (d) con un número par de  $w$ .
3. Si  $x \in \Sigma^*$  y  $\|x^3\| = 36$ , ¿cuánto vale  $\|x\|$ ?
4. Sea  $\Sigma = \{\beta, x, y, z\}$ , donde  $\beta$  denota un espacio en blanco, de modo que  $x\beta \neq x$ ,  $\beta\beta \neq \beta$  y  $x\beta y \neq xy$ , pero  $x\lambda y = xy$ . Calcule lo siguiente:
  - a)  $\|\lambda\|$
  - b)  $\|\lambda\lambda\|$
  - c)  $\|\beta\|$
  - d)  $\|\beta\beta\|$
  - e)  $\|\beta^3\|$
  - f)  $\|x\beta\beta y\|$
  - g)  $\|\beta\lambda\|$
  - h)  $\|\lambda^{10}\|$
5. Sea  $\Sigma = \{v, w, x, y, z\}$  y  $A = \bigcup_{n=1}^6 \Sigma^n$ . ¿Cuántas cadenas en  $A$  tienen a  $xy$  como prefijo propio?
6. Sea  $\Sigma$  un alfabeto. Sea  $x_i \in \Sigma$  para  $1 \leq i \leq 100$  (donde  $x_i \neq x_j$  para cualquier  $1 \leq i < j \leq 100$ ). ¿Cuántas subcadenas no vacías existen para la cadena  $s = x_1 x_2 \dots x_{100}$ ?
7. Para el alfabeto  $\Sigma = \{0, 1\}$ , sean  $A, B, C \subseteq \Sigma^*$  los siguientes lenguajes:
  - a)  $A \cap B$
  - b)  $A - B$
  - c)  $A \Delta B$
  - d)  $A \cap C$
  - e)  $B \cap C$
  - f)  $B \cup C$
  - g)  $\overline{(A \cap C)}$
  - h)  $\overline{A} \cap \overline{C}$
  - i)  $\overline{A} \cap B$

$$\begin{aligned} A &= \{0, 1, 00, 11, 000, 111, 0000, 1111\}, \\ B &= \{w \in \Sigma^* \mid 2 \leq \|w\| \}, \\ C &= \{w \in \Sigma^* \mid 2 \geq \|w\| \}. \end{aligned}$$

Determine los siguientes subconjuntos (lenguajes) de  $\Sigma^*$ .

- a)  $A \cap B$
- b)  $A - B$
- c)  $A \Delta B$
- d)  $A \cap C$
- e)  $B \cap C$
- f)  $B \cup C$
- g)  $\overline{(A \cap C)}$
- h)  $\overline{A} \cap \overline{C}$
- i)  $\overline{A} \cap B$

8. Sean  $A = \{10, 11\}$ ,  $B = \{00, 1\}$  lenguajes para el alfabeto  $\Sigma = \{0, 1\}$ . Determine lo siguiente:  
 (a)  $AB$ ; (b)  $BA$ ; (c)  $A^3$ ; (d)  $B^2$ .
9. Si  $A, B, C$  y  $D$  son lenguajes sobre  $\Sigma$ , demuestre que (a)  $(A \subseteq B \wedge C \subseteq D) \Rightarrow AC \subseteq BD$ ; y  
 (b)  $A \emptyset = \emptyset A = \emptyset$ .
10. Para  $\Sigma = \{x, y, z\}$ , sean  $A, B \subseteq \Sigma^*$  dadas por  $A = \{xy\}$  y  $B = \{\lambda, x\}$ . Determine (a)  $AB$ ; (b)  $BA$ ; (c)  $B^3$ ; (d)  $B^+$ ; (e)  $A^*$ .
11. Dado un alfabeto  $\Sigma$ , ¿existe algún lenguaje  $A \subseteq \Sigma^*$  tal que  $A^* = A$ ?
12. Para  $\Sigma = \{0, 1\}$ , determine si la cadena 00010 está en cada uno de los siguientes lenguajes (tomados de  $\Sigma^*$ ).  
 a)  $\{0, 1\}^*$   
 b)  $\{000, 101\}\{10, 11\}$   
 c)  $\{00\}\{0\}^*\{10\}$   
 d)  $\{000\}^*\{1\}^*\{0\}$   
 e)  $\{00\}^*\{10\}^*$   
 f)  $\{0\}^*\{1\}^*\{0\}^*$
13. Para  $\Sigma = \{0, 1\}$ , describa las cadenas en  $A^*$  para cada uno de los siguientes lenguajes  $A \subseteq \Sigma^*$ :  
 a)  $\{01\}$   
 b)  $\{000\}$   
 c)  $\{0, 010\}$   
 d)  $\{1, 10\}$
14. Para  $\Sigma = \{0, 1\}$ , determine todos los posibles lenguajes  $A, B \subseteq \Sigma^*$  tales que  $AB = \{01, 000, 0101, 0111, 01000, 010111\}$ .
15. Dado un lenguaje no vacío  $A \subseteq \Sigma^*$ , demuestre que si  $A^2 = A$ , entonces  $\lambda \in A$ .
16. Para un alfabeto dado  $\Sigma$ , sea  $a \in \Sigma$ , con  $a$  fijo. Defina las funciones  $p_a, s_a, r: \Sigma^* \rightarrow \Sigma^*$  y la función  $d: \Sigma^* \rightarrow \Sigma^*$  como sigue:
  - La función prefijo (con  $a$ ):  $p_a(x) = ax, x \in \Sigma^*$ .
  - La función sufijo (con  $a$ ):  $s_a(x) = xa, x \in \Sigma^*$ .
  - La función de inversión:  $r(\lambda) = \lambda$ ; para  $x \in \Sigma^*$ , si  $x = x_1 x_2 \dots x_{n-1} x_n$ , donde  $x_i \in \Sigma$ , para todo  $1 \leq i \leq n$ , entonces  $r(x) = x_n x_{n-1} \dots x_2 x_1 = x^R$  (como se definió en el ejemplo 6.17).
  - La función de eliminación frontal: para cada  $x \in \Sigma^*$ , si  $x = x_1 x_2 x_3 \dots x_n$ , entonces  $d(x) = x_2 x_3 \dots x_n$ .
 a) ¿Cuáles de estas cuatro funciones son uno a uno?  
 b) ¿Cuáles de estas cuatro funciones son sobre? Si una función no es sobre, determine su imagen.  
 c) ¿Es alguna de estas funciones invertible? En tal caso, determine su función inversa.  
 d) Sea  $\Sigma = \{a, e, i, o, u\}$ . ¿Cuántas palabras  $x \in \Sigma^4$  satisfacen  $r(x) = x$ ? ¿Cuántas en  $\Sigma^5$ ? ¿Cuántas en  $\Sigma^n$ , donde  $n \in \mathbb{N}$ ?  
 e) Para  $x \in \Sigma^*$ , determine  $(d \circ p_a)(x)$  y  $(r \circ d \circ r \circ s_a)(x)$ .  
 f) Si  $\Sigma = \{a, e, i, o, u\}$  y  $B = \{ae, ai, ao, oo, eio, eiou\} \subseteq \Sigma^*$ , determine  $r^{-1}(B), p_a^{-1}(B), s_a^{-1}(B)$ , y  $|d^{-1}(B)|$ .
17. Si  $A (\neq \emptyset)$  es un lenguaje y  $A^2 = A$ , demuestre que  $A = A^*$ .
18. Proporcione las demostraciones de las partes restantes de los teoremas 6.1 y 6.2.
19. Demuestre que para cualesquier lenguajes finitos  $A, B \subseteq \Sigma^*$   $|AB| \leq |A||B|$ .
20. Si  $A, B$  son lenguajes sobre  $\Sigma$  y  $A \subseteq B^*$ , demuestre que  $A^* \subseteq B^*$ .
21. Para un alfabeto dado  $\Sigma$ , sea  $I$  un conjunto de índices, donde para cada  $i \in I$ ,  $B_i \subseteq \Sigma^*$ . Si  $A \subseteq \Sigma^*$ , demuestre que (a)  $A(\bigcup_{i \in I} B_i) = \bigcup_{i \in I} AB_i$  y (b)  $(\bigcup_{i \in I} B_i)A = \bigcup_{i \in I} B_i A$ . [Estos resultados generalizan las partes (c) y (d) del teorema 6.1.]
22. Para  $\Sigma = \{x, y\}$ , use los lenguajes finitos de  $\Sigma^*$  (como en el ejemplo 6.12), junto con las operaciones de conjuntos, para describir el conjunto de cadenas de  $\Sigma^*$  tales que (a) contienen exactamente una ocurrencia de  $x$ ; (b) contienen exactamente dos ocurrencias de  $x$ ; (c) comienzan con  $x$ ; (d) terminan con  $yxy$ ; (e) comienzan con  $x$  o terminan con  $yxy$  o ambas; (f) comienzan con  $x$  o terminan con  $yxy$  pero no ambas.
23. Sea  $\Sigma$  el alfabeto  $\{0, 1\}$  y sea  $A \subseteq \Sigma^*$  el lenguaje definido recursivamente de la manera siguiente:  
 1) Los símbolos 0, 1 están en  $A$ ; ésta es la base de nuestra definición; y  
 2) Para cualquier palabra  $x \in A$ , la palabra  $0x1$  también está en  $A$ ; esto constituye el proceso recursivo.

- (a) Determine cuatro palabras diferentes de  $A$ ; dos de longitud 3 y dos de longitud 5.  
 (b) Use la definición recursiva dada para mostrar que 0001111 está en  $A$ .  
 (c) Explique por qué 00001111 no está en  $A$ .
24. Para el alfabeto  $\Sigma = \{0, 1\}$ , sea  $A \subseteq \Sigma^*$  el lenguaje formado por todas las palabras que *no* contienen la subcadena 10. Entonces, las palabras como  $\lambda, 0, 00, 000, 0000, 01, 001, 0011, 0111, 1, 11, 111$  están en  $A$ , pero ninguna de las palabras 10, 101, 100, 010, 0010 o 11110 están en este lenguaje.
- a) Dé una definición recursiva del lenguaje  $A$ .  
 b) Use la definición de la parte (a) para determinar si 00111 está en  $A$ .
25. Use la definición recursiva del ejemplo 6.15 para verificar que cada una de las siguientes cadenas está en el lenguaje  $A$  de ese ejemplo.
- a) (( ))( )      b) (( ))( )( )      c) ( )( )( )
26. Proporcione una definición recursiva de cada uno de los siguientes lenguajes  $A \subseteq \Sigma^*$ , donde  $\Sigma = \{0, 1\}$ .
- a)  $x \in A$  si (y sólo si) el número de ceros de  $x$  es par.  
 b)  $x \in A$  si (y sólo si) sólo aparece un cero en  $x$ .  
 c)  $x \in A$  si (y sólo si) todos los unos de  $x$  preceden a todos los ceros.
27. Para cualquier alfabeto  $\Sigma$ , una cadena  $x$  de  $\Sigma^*$  es un *palíndromo* si  $x = x^R$  (es decir,  $x$  es igual a su inverso). Si  $A \subseteq \Sigma^*$ , donde  $A = \{x \in \Sigma^* \mid x = x^R\}$ , ¿cómo podemos definir el lenguaje  $A$  de manera recursiva?

## 6.2

### Máquinas de estados finitos: Un primer encuentro

Regresemos ahora a la máquina expendedora mencionada al comienzo de este capítulo y analicémosla en las siguientes circunstancias.

En cierta oficina, una máquina expendedora tiene dos tipos de refrescos en lata: cola (C) y cerveza de raíz (RB). El costo de una lata es de 20 centavos. La máquina acepta monedas de cinco, diez y veinticinco centavos y devuelve el cambio necesario. Un día, María decidió comprar una lata de cerveza de raíz. Fue a la máquina expendedora, insertó dos monedas de cinco centavos y una de diez centavos, en ese orden, y oprimió el botón blanco, marcado con W. El resultado: salió su lata de cerveza de raíz. (Para obtener una lata de cola se presiona el botón negro, designado por B.)

Podemos representar lo que hizo María para hacer su compra, como aparece en la tabla 6.1;  $t_0$  es el instante inicial, cuando ella inserta su primera moneda de cinco centavos, y  $t_1, t_2, t_3, t_4$  son los instantes posteriores, con  $t_1 < t_2 < t_3 < t_4$ .

**Tabla 6.1**

	$t_0$	$t_1$	$t_2$	$t_3$	$t_4$
Estado	(1) $s_0$	(4) $s_1$ (5¢)	(7) $s_2$ (10¢)	(10) $s_3$ (20¢)	(13) $s_0$
Entrada	(2) 5¢	(5) 5¢	(8) 10¢	(11) W	
Salida	(3) Nada	(6) Nada	(9) Nada	(12) RB	

Los números (1), (2), ..., (12), (13) de esta tabla indican el orden de sucesos en la compra de la cerveza de raíz de María. Para cada entrada en el instante  $t_i$ ,  $0 \leq i \leq 3$ , existe *en ese instante* una salida correspondiente y luego un cambio de estado. El siguiente estado  $t_{i+1}$  depende de la entrada y el estado (presente) en el instante  $t_i$ .

La máquina está en un estado de espera en el estado  $s_0$ . Espera que un cliente comience a insertar monedas con un total de 20 centavos o más y oprima un botón para obtener un refresco. Si en cualquier momento el total de monedas insertadas excede los 20 centavos, la máquina proporciona el cambio necesario (antes de que el cliente oprima el botón para obtener su refresco).

En el instante  $t_0$ , María inserta en la máquina su primera entrada, 5 centavos. No recibe nada en ese momento, pero en el tiempo siguiente  $t_1$ , la máquina está en el estado  $s_1$ , donde ésta *recuerda* su total de 5 centavos y espera una segunda entrada (de 5 centavos en el instante  $t_1$ ). Nuevamente, la máquina (en el instante  $t_1$ ) sigue sin proporcionar una salida, pero al siguiente instante,  $t_2$ , está en el estado  $s_2$ , recordando un total de diez centavos, que es igual a 5 centavos (recordados en el estado  $s_1$ ) más cinco centavos (insertados en el instante  $t_1$ ). Al introducir la moneda de diez centavos (en el instante  $t_2$ ) como siguiente entrada a la máquina, María todavía no recibe el refresco ya que la máquina no “sabe” el tipo que ella prefiere, pero ahora “sabe” ( $t_3$ ) que ha insertado el total necesario de 20 centavos = 10 centavos (recordados en el estado  $s_2$ ) más 10 centavos (insertados en el instante  $t_2$ ). Por último, María oprime el botón blanco y en el instante  $t_4$  la máquina entrega la salida (su lata de cerveza de raíz) y después regresa, en el instante  $t_4$ , al punto de partida  $s_0$ , justo a tiempo para que Rodrigo, el amigo de María, deposite una moneda de 25 centavos, reciba una moneda de 5 centavos como cambio, oprima el botón negro y obtenga la lata de refresco de cola que desea. La compra hecha por Rodrigo se analiza en la tabla 6.2.

Tabla 6.2

	$t_0$	$t_1$	$t_2$
<b>Estado</b>	(1) $s_0$	(4) $s_3$ (20¢)	(7) $s_0$
<b>Entrada</b>	(2) 25¢	(5) B	
<b>Salida</b>	(3) cambio de 5¢	(6) C	

Lo que ha ocurrido en el caso de esta máquina puede abstraerse para ayudar en el análisis de ciertos aspectos de los computadores digitales y los sistemas de comunicación telefónica.

Las principales características de esta máquina son las siguientes:

- 1) La máquina sólo puede estar en uno de una *cantidad finita de estados* en un instante dado. Estos estados son los *estados internos* de la máquina y, en un instante dado, la memoria total disponible de la máquina es el conocimiento del estado interno en el que se encuentra en ese instante.
- 2) La máquina aceptará como *entrada* sólo un número finito de símbolos, que se conocen en conjunto como el *alfabeto de entrada*  $\mathcal{E}$ . En el ejemplo de la máquina expendedora, el alfabeto de entrada es {moneda de 5 centavos, moneda de 10 centavos, moneda de 25 centavos, blanco, negro}, y cada elemento se reconoce por su estado interno.
- 3) Mediante cada combinación de entradas y estados internos se determina una *salida* y un *estado siguiente*. El conjunto finito de todas las salidas posibles constituye el *alfabeto de salida*  $\mathcal{O}$  de la máquina.
- 4) Suponemos que los procesos secuenciales de la máquina están *sincronizados* por pulsos de reloj separados y distintos y que la máquina opera de manera *determinista*, ya que la salida queda completamente determinada por la entrada total proporcionada y el estado inicial de la máquina.

Estas observaciones conducen a la siguiente definición.

**Definición 6.13**

Una *máquina de estados finitos* es una 5-upla  $M = (S, \mathcal{I}, \mathcal{O}, v, \omega)$ , donde  $S$  es el conjunto de estados internos de  $M$ ;  $\mathcal{I}$  es el alfabeto de entrada de  $M$ ;  $\mathcal{O}$  es el alfabeto de salida de  $M$ ;  $v: S \times \mathcal{I} \rightarrow S$  es la *función siguiente estado*; y  $\omega: S \times \mathcal{I} \rightarrow \mathcal{O}$  es la *función de salida*.

Según la notación de esta definición, si la máquina está en el estado  $s$  en el instante  $t_i$  e introducimos  $x$  en ese momento, entonces la salida en el instante  $t_i$  es  $\omega(s, x)$ . Esta salida es seguida por una transición de la máquina, en el instante  $t_{i+1}$ , al siguiente estado interno, dado por  $v(s, x)$ .

Suponemos que cuando una máquina de estados finitos recibe su primera entrada, estamos en el instante  $t_0 = 0$  y la máquina está en un estado de inicio ya determinado, denotado con  $s_0$ . Nuestro desarrollo se concentrará principalmente en la salida y las transiciones de estado que ocurren de manera secuencial, con poca o ninguna referencia a la sucesión de pulsos de reloj en los instantes  $t_0, t_1, t_2, \dots$

Puesto que los conjuntos  $S, \mathcal{I}$  y  $\mathcal{O}$  son finitos, es posible representar  $v$  y  $\omega$ , para una máquina de estados finitos dada, por medio de una tabla que enumera  $v(s, x)$  y  $\omega(s, x)$  para todo  $s \in S$  y todo  $x \in \mathcal{I}$ , y que se conoce como la *tabla de estados* o *tabla de transición* de la máquina dada. Una segunda representación de la máquina se hace por medio de un *diagrama de estados*.

Mostraremos la tabla de estados y el diagrama de estados en los siguientes ejemplos.

**Ejemplo 6.18**

Consideremos la máquina de estados finitos  $M = (S, \mathcal{I}, \mathcal{O}, v, \omega)$ , donde  $S = \{s_0, s_1, s_2\}$ ,  $\mathcal{I} = \{\emptyset\} = \{0, 1\}$  y  $v, \omega$  aparecen en la *tabla de estados* de la tabla 6.3. La primera columna de la tabla enumera los *estados (presentes)* de la máquina. Las entradas de la segunda fila son los elementos del alfabeto de entrada  $\mathcal{I}$ , que se enumeran una vez bajo  $v$  y otra vez bajo  $\omega$ . Los seis números de las últimas dos columnas (y las últimas tres filas) son elementos del alfabeto de salida  $\mathcal{O}$ .

**Tabla 6.3**

	$v$		$\omega$	
	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

Por ejemplo, para calcular  $v(s_1, 1)$ , encontramos  $s_1$  en la columna de estados presentes y procedemos en horizontal sobre  $s_1$ , hasta estar debajo de la entrada 1 en la sección de las tablas de  $v$ . Esta entrada da  $v(s_1, 1) = s_1$ . De la misma forma vemos que  $\omega(s_1, 1) = 0$ .

Si  $s_0$  indica el estado inicial y la entrada proporcionada a  $M$  es la cadena 1010, entonces la salida es 0010, como lo demuestra la tabla 6.4. En este caso, la máquina se queda en el

**Tabla 6.4**

<b>Estado</b>	$s_0$	$v(s_0, 1) = s_1$	$v(s_1, 0) = s_2$	$v(s_2, 1) = s_1$	$v(s_1, 0) = s_2$
<b>Entrada</b>	1	0	1	0	0
<b>Salida</b>	$\omega(s_0, 1) = 0$	$\omega(s_1, 0) = 0$	$\omega(s_2, 1) = 1$	$\omega(s_1, 0) = 0$	

estado  $s_2$ , de modo que si tuviéramos otra cadena de entrada, proporcionaríamos el primer carácter de esa cadena, 0 en ese caso, en el estado  $s_2$  a menos que se *vuelva a arrancar* la máquina para iniciar de nuevo en  $s_0$ .

Puesto que estamos interesados principalmente en la salida y no en la sucesión de estados de transición, la misma máquina puede representarse por medio de un *diagrama de estados*. En este caso podemos obtener la cadena de salida sin enumerar los estados de transición. En tal diagrama, cada estado interno  $s$  se representa mediante un círculo con  $s$  dentro de él. Para los estados  $s_i$  y  $s_j$ , si  $v(s_i, x) = s_j$  para  $x \in \mathcal{G}$ , y  $\omega(s_i, x) = y$  para  $y \in \mathcal{O}$ , representamos esto en el diagrama de estados trazando una *arista dirigida* (o *arco*) del círculo para  $s_i$  al círculo para  $s_j$  y rotulamos el arco con la entrada  $x$  y la salida  $y$ , como se muestra en la figura 6.1.

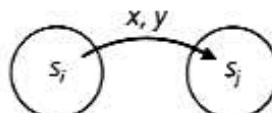


Figura 6.1

Con estos convenios, el diagrama de estados de la máquina  $M$  de la tabla 6.3 aparece en la figura 6.2. Aunque la tabla es más compacta, el diagrama nos permite seguir una cadena de entrada a través de cada uno de los estados de transición que determina, eligiendo cada uno de los símbolos correspondientes de salida antes de cada transición. Si la cadena de entrada es 00110101, entonces, partiendo del estado  $s_0$ , la primera entrada de 0 produce una salida de 0 y nos lleva de nuevo a  $s_0$ . La siguiente entrada de 0 produce el mismo resultado, pero para la tercera entrada, 1, la salida es 0 y estamos ahora en el estado  $s_1$ . Si seguimos de esta forma, llegamos a la cadena de salida 00000101 y terminamos en el estado  $s_1$ . (Observemos que la cadena de entrada 00110101 es un elemento de  $\mathcal{G}^*$ , la clausura de Kleene de  $\mathcal{G}$  y que la cadena de salida está en  $\mathcal{O}^*$ , la clausura de Kleene de  $\mathcal{O}$ .)

Si partimos de  $s_0$ , ¿cuál es la salida para la cadena de entrada 1100101101?

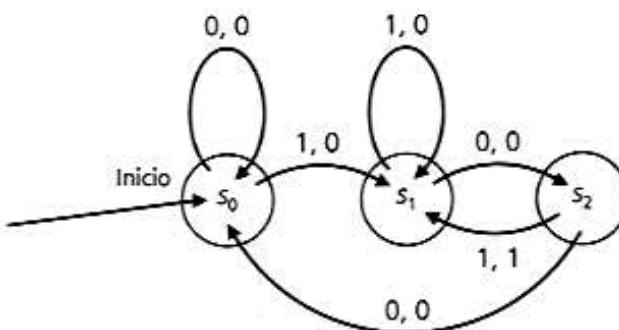


Figura 6.2

**Ejemplo 6.19**

Para la máquina expendedora ya descrita en esta sección, tenemos la tabla de estados de la tabla 6.5, donde

- 1)  $S = \{s_0, s_1, s_2, s_3, s_4\}$ , de modo que en el estado  $s_k$ ,  $0 \leq k \leq 4$ , la máquina recuerda retener  $5k$  centavos.

Tabla 6.5

	$\nu$					$\omega$				
	5¢	10¢	25¢	B	W	5¢	10¢	25¢	B	W
$s_0$	$s_1$	$s_2$	$s_4$	$s_0$	$s_0$	$n$	$n$	5¢	$n$	$n$
$s_1$	$s_2$	$s_3$	$s_4$	$s_1$	$s_1$	$n$	$n$	10¢	$n$	$n$
$s_2$	$s_3$	$s_4$	$s_4$	$s_2$	$s_2$	$n$	$n$	15¢	$n$	$n$
$s_3$	$s_4$	$s_4$	$s_4$	$s_3$	$s_3$	$n$	5¢	20¢	$n$	$n$
$s_4$	$s_4$	$s_4$	$s_4$	$s_0$	$s_0$	5¢	10¢	25¢	C	RB

- 2)  $\mathcal{F} = \{5 \text{ centavos}, 10 \text{ centavos}, 25 \text{ centavos}, B, W\}$ , donde B denota el botón negro que se presiona para obtener el refresco de cola, y W el botón blanco para la cerveza de raíz.
- 3)  $\mathcal{O} = \{n(\text{nada}), RB(\text{cerveza de raíz}), C(\text{cola}), 5 \text{ centavos}, 10 \text{ centavos}, 15 \text{ centavos}, 20 \text{ centavos}, 25 \text{ centavos}\}$ .

Como observamos en el análisis anterior al ejemplo 6.19, para una máquina de estados finitos general  $M = \{S, \mathcal{F}, \mathcal{O}, \nu, \omega\}$ , la entrada puede pensarse como un elemento de  $\mathcal{F}^*$ , y la salida como uno de  $\mathcal{O}^*$ . En consecuencia, podemos extender los dominios de  $\nu$  y  $\omega$  de  $S \times \mathcal{F}$  a  $S \times \mathcal{F}^*$ . Para  $\omega$  ampliamos el codominio a  $\mathcal{O}^*$ , recordando que, de ser necesario, tanto  $\mathcal{F}^*$  como  $\mathcal{O}^*$  contienen la cadena vacía  $\lambda$ . Con estas extensiones, si  $x_1 x_2 \dots x_k \in \mathcal{F}^*$ , para  $k \in \mathbb{Z}^+$ , y si partimos de cualquier estado  $s_1 \in S$ , tenemos

$$\begin{aligned} \nu(s_1, x_1) &= s_2 \\ \nu(s_1, x_1 x_2) &= \nu(\nu(s_1, x_1), x_2) = \nu(s_2, x_2) = s_3 \\ \nu(s_1, x_1 x_2 x_3) &= \nu(\underbrace{\nu(s_1, x_1)}_{s_2}, x_2, x_3) = \nu(s_3, x_3) = s_4 \\ &\quad \vdots \\ \nu(s_1, x_1 x_2 \dots x_k) &= \nu(s_k, x_k) = s_{k+1}, \quad y \\ \omega(s_1, x_1) &= y_1 \\ \omega(s_1, x_1 x_2) &= \omega(s_1, x_1) \omega(\nu(s_1, x_1), x_2) = \omega(s_1, x_1) \omega(s_2, x_2) = y_1 y_2 \\ \omega(s_1, x_1 x_2 x_3) &= \omega(s_1, x_1) \omega(s_2, x_2) \omega(s_3, x_3) = y_1 y_2 y_3 \\ &\quad \vdots \\ \omega(s_1, x_1 x_2 \dots x_k) &= \omega(s_1, x_1) \omega(s_2, x_2) \dots \omega(s_k, x_k) = y_1 y_2 \dots y_k \in \mathcal{O}^* \end{aligned}$$

Además,  $\nu(s_1, \lambda) = s_1$  para cualquier  $s_1 \in S$ .

(Usaremos estas extensiones de nuevo en el capítulo 7.)

Cerraremos esta sección con un ejemplo pertinente en las ciencias de la computación.

**Ejemplo 6.20** Sean  $x = x_5 x_4 x_3 x_2 x_1 = 00111$  y  $y = y_5 y_4 y_3 y_2 y_1 = 01101$  números binarios, donde  $x_1$  y  $y_1$  son los bits menos significativos. Los ceros restantes en  $x$  y  $y$  aparecen para que las dos cadenas  $x$  y  $y$  sean de igual longitud y garanticen el espacio suficiente para completar la suma.

† El estado  $s_2$  es determinado por  $s_1$  y  $x_1$ . No es solamente el segundo de una lista predeterminada de estados.

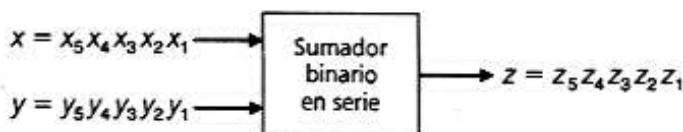


Figura 6.3

Un *sumador binario en serie* es una máquina de estados finitos que podemos usar para obtener  $x + y$ . El diagrama de la figura 6.3 ilustra esto, donde  $z = z_5z_4z_3z_2z_1$  tiene el dígito menos significativo en el bit  $z_1$ .

En la suma  $z = x + y$ , tenemos

$$\begin{array}{r}
 x = 0 \ 0 \ 1 \ 1 \ 1 \\
 + y = + 0 \ 1 \ 1 \ 0 \ 1 \\
 \hline
 z = 1 \ 0 \ 1 \ 0 \ 0
 \end{array}$$

tercera suma                          primera suma

Observamos que para la primera suma  $x_1 = y_1 = 1$  y  $z_1 = 0$ , mientras que para la tercera suma tenemos  $x_3 = y_3 = 1$  y  $z_3 = 1$  por *acarreo* de la suma de  $x_2$  y  $y_2$  (y el *acarreo* de  $x_1 + y_1$ ). En consecuencia, cada salida depende de la suma de dos entradas y de la habilidad para *recordar* un acarreo de 0 o 1, lo que es crucial cuando el acarreo es 1.

El sumador binario en serie se modela mediante una máquina de estados finitos  $M = (S, \mathcal{G}, \mathcal{O}, \nu, \omega)$  como sigue. El conjunto  $S = \{s_0, s_1\}$ , donde  $s_i$  indica un acarreo de  $i$ ;  $\mathcal{G} = \{00, 01, 10, 11\}$ ; así, existe un par de entradas, dependiendo de si buscamos  $0 + 0$ ,  $0 + 1$ ,  $1 + 0$  o  $1 + 1$ , respectivamente; y  $\mathcal{O} = \{0, 1\}$ . Las funciones  $\nu$  y  $\omega$  están dadas en la tabla de estados (Tabla 6.6) y el diagrama de estados (Fig. 6.4).

Tabla 6.6

	$\nu$				$\omega$			
	00	01	10	11	00	01	10	11
$s_0$	$s_0$	$s_0$	$s_0$	$s_1$	0	1	1	0
$s_1$	$s_0$	$s_1$	$s_1$	$s_1$	1	0	0	1

En la tabla 6.6 encontramos, por ejemplo, que  $\nu(s_1, 01) = s_1$  y  $\omega(s_1, 01) = 0$ , ya que  $s_1$  indica un acarreo de 1 desde la suma de los bits anteriores. La entrada 01 indica que estamos sumando 0 y 1 (y acarreando un 1). De aquí que la suma sea 10 y  $\omega(s_1, 01) = 0$  para el 0 de 10. El acarreo se recadena otra vez en  $s_1 = \nu(s_1, 01)$ .

Del diagrama de estados (Fig. 6.4) vemos que el estado inicial debe ser  $s_0$  ya que no hay acarreo anterior a la suma de los bits menos significativos.

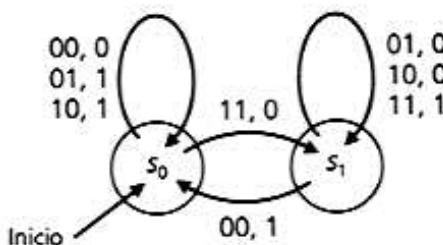


Figura 6.4

Los diagramas de estados de las figuras 6.2 y 6.4 son ejemplos de *grafos dirigidos etiquetados*. Veremos más de la teoría de grafos en este libro, ya que esta teoría no sólo tiene aplicaciones en las ciencias de la computación y la ingeniería eléctrica, sino también en la teoría de códigos (códigos de prefijo) y optimización (redes de transporte).

## EJERCICIOS 6.2

- Con la máquina de estados finitos del ejemplo 6.18, determine la salida para cada una de las siguientes entradas  $x \in \mathcal{G}^*$ , así como el último estado interno en el proceso de transición. (Suponga que siempre partimos del estado  $s_0$ )
  - $x = 1010101$
  - $x = 1001001$
  - $x = 101001000$
- Para la máquina de estados finitos del ejemplo 6.18, una cadena de entrada  $x$  produce la cadena de salida 00101, si partimos del estado  $s_0$ . Determine  $x$ .
- Sea  $M = (S, \mathcal{G}, \mathcal{O}, v, \omega)$  una máquina de estados finitos donde  $S = \{s_0, s_1, s_2, s_3\}$ ,  $\mathcal{G} = \{a, b, c\}$ ,  $\mathcal{O} = \{0, 1\}$ , y  $v$  y  $\omega$  están determinados por la tabla 6.7.

Tabla 6.7

	$v$			$\omega$		
	$a$	$b$	$c$	$a$	$b$	$c$
$s_0$	$s_0$	$s_3$	$s_2$	0	1	1
$s_1$	$s_1$	$s_1$	$s_3$	0	0	1
$s_2$	$s_1$	$s_1$	$s_3$	1	1	0
$s_3$	$s_2$	$s_3$	$s_0$	1	0	1

- Si partimos de  $s_0$ , ¿cuál es la salida para la cadena de entrada  $abbccc$ ?
- Trace el diagrama de estados para esta máquina de estados finitos.
- Proporcione la tabla de estados para la máquina expendedora del ejemplo 6.19, si el costo de cada lata de cola o de cerveza de raíz se incrementa a 25 centavos.
- Una máquina de estados finitos  $M = (S, \mathcal{G}, \mathcal{O}, v, \omega)$  tiene  $\mathcal{G} = \mathcal{O} = \{0, 1\}$  y se determina mediante el diagrama de estados que se muestra en la figura 6.5.

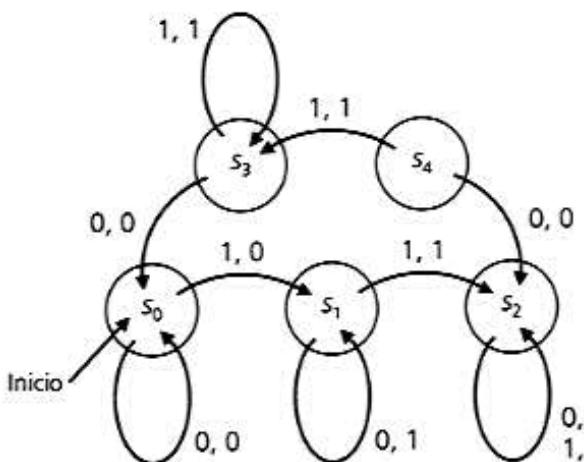


Figura 6.5

- Determine la cadena de salida para la cadena de entrada 110111, comenzando en  $s_0$ . ¿Cuál es el último estado de transición?
- Responda la parte (a) para la misma cadena pero tomando  $s_1$  como el estado inicial. ¿Qué sucede cuando  $s_2$  y  $s_3$  son los estados iniciales?

- c) Encuentre la tabla de estados para esta máquina.
- d) ¿Desde qué estado debemos comenzar para que la cadena de entrada 10010 produzca la salida 10000?
- e) Determine una cadena de entrada  $x \in \{0, 1\}^*$  de longitud mínima, tal que  $v(s_4, x) = s_1$ . ¿Es  $x$  único?
6. La máquina  $M$  tiene  $\mathcal{G} = \{0, 1\} = \mathcal{C}$  y se determina mediante el diagrama de estados de la figura 6.6.

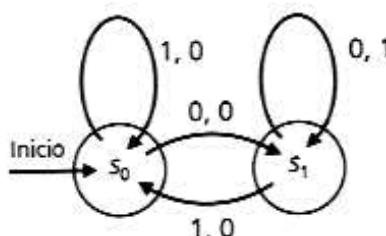


Figura 6.6

- a) Describa con palabras lo que hace esta máquina de estados finitos.
- b) ¿Qué debe recordar el estado  $s_1$ ?
- c) Encuentre dos lenguajes  $A, B \subseteq \{0, 1\}^*$  tales que para cada  $x \in AB$ ,  $\omega(s_0, x)$  tiene a 1 como un sufijo.
7. a) Si  $S$ ,  $\mathcal{G}$  y  $\mathcal{C}$  son conjuntos finitos, con  $|S| = 3$ ,  $|\mathcal{G}| = 5$ , y  $|\mathcal{C}| = 2$ , determine
- $|S \times \mathcal{G}|$ ;
  - el número de funciones  $v: S \times \mathcal{G} \rightarrow S$ ; y,
  - el número de funciones  $\omega: S \times \mathcal{C} \rightarrow \mathcal{C}$ .
- b) Para  $S$ ,  $\mathcal{G}$ , y  $\mathcal{C}$  de la parte (a), ¿cuántas máquinas de estados finitos determinan?
8. Sea  $M = (S, \mathcal{G}, \mathcal{C}, v, \omega)$  una máquina de estados finitos con  $\mathcal{G} = \mathcal{C} = \{0, 1\}$  y  $S, v$  y  $\omega$  determinados por el diagrama de estados de la figura 6.7.

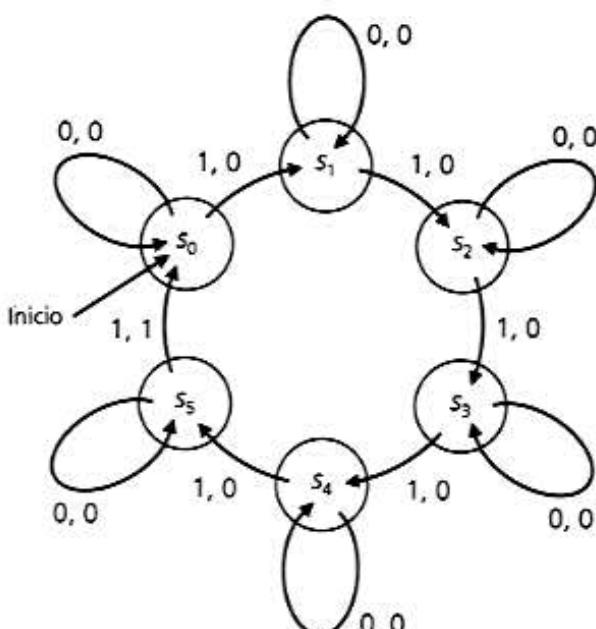


Figura 6.7

- a) Encuentre la salida para la cadena de entrada  $x = 0110111011$ .
- b) Dé la tabla de transición para esta máquina de estados finitos.
- c) Si partimos del estado  $s_0$  y la salida para la cadena de entrada  $x$  es 0000001, determine todas las posibilidades para  $x$ .
- d) Describa con palabras lo que hace esta máquina de estados finitos.

9. a) Encuentre la tabla de estados para la máquina de estados finitos de la figura 6.8, donde  $\mathcal{G} = \{0, 1\}$ .

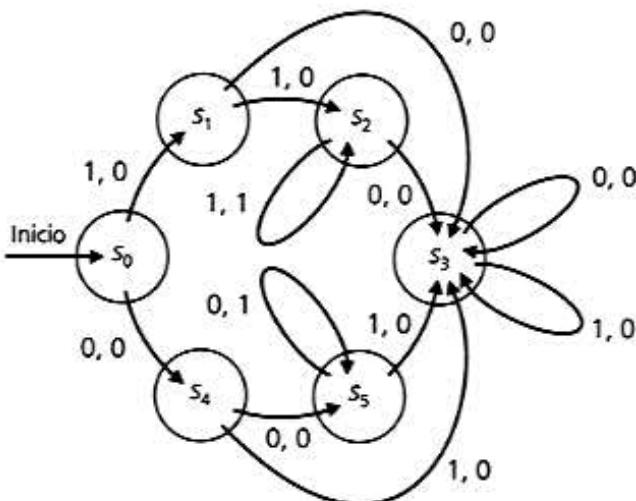


Figura 6.8

- b) Sea  $x \in \mathcal{G}^*$  con  $\|x\| = 4$ . Si 1 es un sufijo de  $\omega(s_0, x)$ , ¿cuáles son las posibilidades para la cadena  $x$ ?  
 c) Sea  $A \subseteq \{0, 1\}^*$  el lenguaje tal que  $\omega(s_0, x)$  tiene 1 como un sufijo para todo  $x$  de  $A$ . Determine  $A$ .  
 d) Encuentre el lenguaje  $A \subseteq \{0, 1\}^*$  tal que  $\omega(s_0, x)$  tiene a 111 como un sufijo para todo  $x$  de  $A$ .

### 6.3

#### Máquinas de estados finitos: Un segundo encuentro

Habiendo visto algunos ejemplos de máquinas de estados finitos, pasemos al estudio de otras máquinas que son importantes para el diseño del hardware de computadores. Un tipo importante de máquina es el *reconocedor secuencial*.

##### Ejemplo 6.21

En este caso,  $\mathcal{G} = \mathcal{O} = \{0, 1\}$  y queremos construir una máquina que reconozca cada aparición de la secuencia 111 al encontrarla en cualquier cadena de entrada  $x \in \mathcal{G}^*$ . Por ejemplo, si  $x = 111010111$ , entonces la salida correspondiente debe ser 001000011, donde un 1 en la posición  $i$ -ésima de la salida indica que aparece un 1 en las posiciones  $i$ ,  $i - 1$ , e  $i - 2$  de  $x$ . En este caso, las secuencias 111 pueden solaparse, por lo que podemos pensar que algunos caracteres en la cadena de entrada son caracteres de más de una terna de unos.

Si  $s_0$  denota el estado inicial, vemos que debemos tener un estado que recuerde 1 (el posible comienzo de 111) y un estado que recuerde 11. Además, cada vez que nuestro símbolo de entrada sea 0, regresamos a  $s_0$  para comenzar de nuevo la búsqueda de tres unos consecutivos.

En la figura 6.9,  $s_1$  recuerda un único 1, y  $s_2$  recuerda la cadena 11. Si se llega a  $s_2$ , entonces un tercer “1” indica la presencia de la terna en la cadena de entrada, y la salida 1 *reconoce* esta presencia. Pero también este tercer “1” significa que tenemos los primeros dos unos de otra posible terna que sigue dentro de la cadena (como sucede en 11101011).

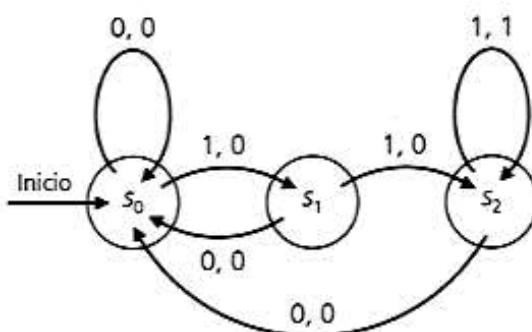


Figura 6.9

"1" 1). Así, después de reconocer la presencia de 111 con una salida de 1, volvemos al estado  $s_2$  para recordar las dos entradas de 1 "1".

Si nos interesa reconocer todas las cadenas que terminen en 111, entonces para cualquier  $x \in \{0, 1\}^*$ , la máquina reconocerá tal secuencia con una salida final de 1. Esta máquina es entonces un reconocedor del lenguaje  $A = \{0, 1\}^*\{111\}$ .

En la figura 6.10 aparece otra máquina de estados finitos que reconoce la misma terna 111. Las máquinas de estados finitos representadas por los diagramas de estados de las figuras 6.9 y 6.10 realizan la misma tarea y se dice que son *equivalentes*. El diagrama de estados de la figura 6.10 tiene un estado más que el de la figura 6.9, pero por el momento no nos interesa obtener una máquina de estados finitos con un número mínimo de estados. En el capítulo 7 desarrollaremos una técnica que toma una máquina de estados finitos dada  $M$  y encuentra otra máquina equivalente a ésta, con el número más pequeño de estados internos necesarios.

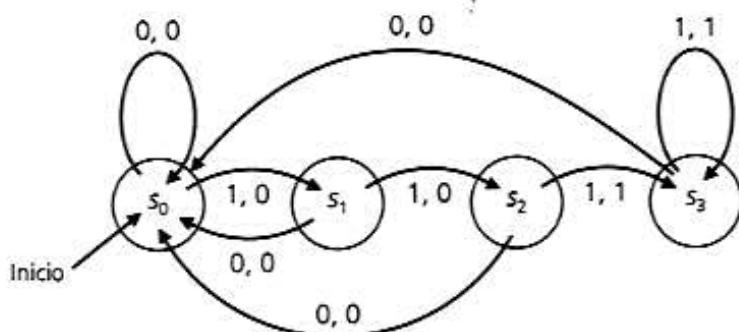


Figura 6.10

El siguiente ejemplo es un poco más selectivo.

### Ejemplo 6.22

Ahora no sólo queremos reconocer la presencia de 111 sino también queremos reconocer únicamente aquellas ocurrencias que terminen en una posición múltiplo de tres. En consecuencia, si  $\mathcal{A} = \emptyset = \{0, 1\}$  y  $x \in \mathcal{A}^*$ , donde  $x = 1110111$ , entonces queremos que  $\omega(s_0, x) = 0010000$ , y no 0010001. Además, para  $x \in \mathcal{A}^*$ , donde  $x = 111100111$ , la salida  $\omega(s_0, x)$  es 001000001, y no 001100001, ya que, considerando la longitud, no se permite el solapamiento de las sucesiones 111.

Otra vez comenzamos en  $s_0$  (Fig. 6.11), pero ahora  $s_1$  debe recordar un primer 1 sólo si éste ocurre en  $x$  en la posición 1, 4, 7, ... Si la entrada en  $s_0$  es 0, simplemente no podemos regresar a  $s_0$  como en el ejemplo 6.21. Debemos recordar que este 0 es el *primero* de los tres símbolos que no nos interesan. De aquí que de  $s_0$  vamos a  $s_3$  y luego a  $s_4$ , procesando

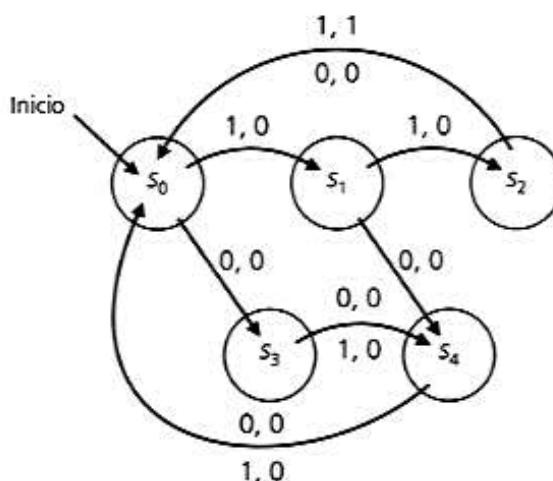


Figura 6.11

cualquier terna de la forma  $0yz$  donde 0 aparece en  $x$  en la posición  $3k + 1$ ,  $k \geq 0$ . El mismo tipo de situación sucede en  $s_1$  cuando la entrada es 0. Por último, en  $s_2$ , la sucesión de 111 se reconoce con una salida de 1, si ésta ocurre. Entonces la máquina regresa a  $s_0$  para dar paso al siguiente símbolo de entrada de la cadena.

**Ejemplo 6.23**

La figura 6.12 muestra los diagramas de estados para una máquina de estados finitos que reconocerá la ocurrencia de una secuencia de 0101 en una cadena de entrada  $x \in \{0, 1\}^*$ , donde  $\emptyset = \{0, 1\}$ . La máquina de la figura 6.12(a) reconoce con una salida de 1 cada ocurrencia de 0101 en una cadena de entrada, sin importar dónde ocurra. En la figura 6.12(b) la máquina reconoce con una salida de 1 solamente aquellos prefijos de  $x$  cuya longitud es un múltiplo de cuatro y cuyo final es 0101. (Por lo tanto, en este caso no se permite el solapamiento.) En consecuencia, para  $x = 01010100101$ ,  $\omega(s_0, x) = 00010100001$  para (a), mientras que para (b),  $\omega(s_0, x) = 00010000000$ .

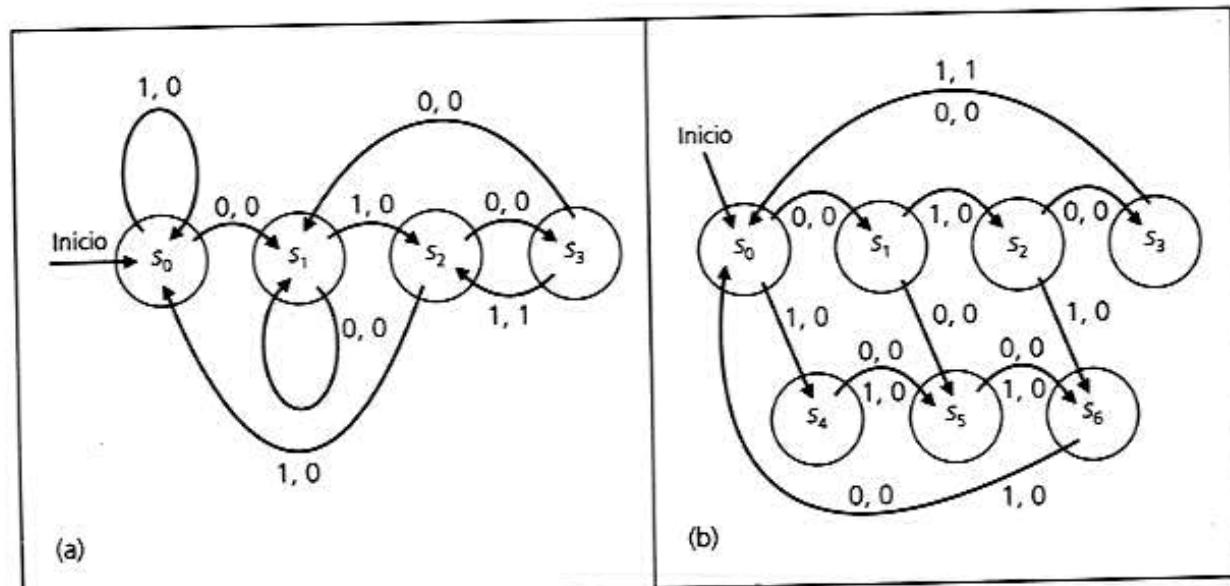


Figura 6.12

Ahora que ya hemos analizado algunas máquinas de estados finitos que reconocen sucesiones, es el momento de examinar un conjunto de sucesiones que *no pueden* ser reconocidas por cualquier máquina de estados finitos. Este ejemplo nos da otra oportunidad para aplicar el principio del palomar.

**Ejemplo 6.24**

Sean  $\mathcal{F} = \emptyset = \{0, 1\}$ . ¿Podremos construir una máquina de estados finitos que reconozca con precisión las cadenas del lenguaje  $A = \{01, 0011, 000111, \dots\} = \{0^i 1^i \mid i \in \mathbb{Z}^+\}$ ? Si lo logramos y  $s_0$  denota el estado inicial, esperaríamos que  $\omega(s_0, 01) = 01$ ,  $\omega(s_0, 0011) = 0011$ , y, en general,  $\omega(s_0, 0^i 1^i) = 0^i 1^i$ , para cualquier  $i \in \mathbb{Z}^+$ . [Nota: En este caso, por ejemplo, queremos  $\omega(s_0, 0011) = 0011$ , donde el primer 1 es la salida para reconocer la subcadena 01 y el segundo 1 es para reconocer la cadena 0011.]

Suponga que existe una máquina de estados finitos  $M = (S, \mathcal{F}, \emptyset, v, \omega)$  que puede reconocer precisamente las cadenas de  $A$ . Sea  $s_0 \in S$ , donde  $s_0$  es el estado inicial, y sea  $|S| = n \geq 1$ . Consideremos ahora la cadena  $0^{n+1} 1^{n+1}$  del lenguaje  $A$ . Si nuestra máquina de estado  $M$  opera correctamente, entonces  $\omega(s_0, 0^{n+1} 1^{n+1}) = 0^{n+1} 1^{n+1}$ . Por lo tanto, en la tabla 6.8 vemos la forma en que esta máquina de estados finitos procesará los  $n + 1$  ceros, comenzando en el estado  $s_0$ , para después continuar con los  $n$  estados  $s_1 = v(s_0, 0), s_2 = v(s_1, 0), \dots, s_n = v(s_{n-1}, 0)$ . Como  $|S| = n$ , aplicando el principio del palomar a los  $n + 1$  estados  $s_0, s_1, s_2, \dots, s_{n-1}, s_n$ , nos damos cuenta de que hay dos estados  $s_i$  y  $s_j$  donde  $i < j$  pero  $s_i = s_j$ .

**Tabla 6.8**

Estado	$s_0$	$s_1$	$s_2$	...	$s_{n-1}$	$s_n$	$s_{n+1}$	...	$s_{2n}$	$s_{2n+1}$
Entrada	0	0	0	...	0	0	1	...	1	1
Salida	0	0	0	...	0	0	1	...	1	1

En la tabla 6.9 vemos cómo la eliminación de las  $j - i$  columnas (para los estados  $s_{i+1}, \dots, s_j$ ) da como resultado la tabla 6.10. Esta tabla nos muestra que la máquina de estados finitos  $M$  reconoce la cadena  $x = 0^{(n+1)-(j-i)} 1^{n+1}$ , donde  $n + 1 - (j - i) < n + 1$ . Por desgracia,  $x \notin A$ , de modo que  $M$  reconoce una cadena que supuestamente *no* debe reconocerse. Esto

**Tabla 6.9**

Estado	$s_0$	$s_1$	$s_2$	...	$s_i$	$s_{i+1}$	...	$s_j$	$s_{j+1}$	...	$s_n$	$s_{n+1}$	...	$s_{2n}$	$s_{2n+1}$
Entrada	0	0	0	...	0	0	...	0	0	...	0	1	...	1	1
Salida	0	0	0	...	0	0	...	0	0	...	0	1	...	1	1

**Tabla 6.10**

Estado	$s_0$	$s_1$	$s_2$	...	$s_i$	$s_{j+1}$	...	$s_n$	$s_{n+1}$	...	$s_{2n}$	$s_{2n+1}$
Entrada	0	0	0	...	0	0	...	0	1	...	1	1
Salida	0	0	0	...	0	0	...	0	1	...	1	1

demuestra que no podemos construir una máquina de estados finitos que reconozca precisamente las cadenas del lenguaje  $A = \{0^i 1^i \mid i \in \mathbb{Z}^+\}$ .

Una clase de máquinas de estados finitos importante en el diseño de dispositivos digitales es la de las *máquinas de retraso de  $k$ -unidades*, donde  $k \in \mathbb{Z}^*$ . Para  $k = 1$ , queremos construir una máquina  $M$  tal que si  $x = x_1 x_2 \dots x_{m-1} x_m$ , entonces para el estado inicial  $s_0$ ,  $\omega(s_0, x) = 0x_1 x_2 \dots x_{m-1}$ , de tal manera que la salida es la entrada retardada una unidad de tiempo (pulso de reloj). [El uso de 0 como un primer símbolo en  $\omega(s_0, x)$  es convencional.]

### Ejemplo 6.25

Sean  $\mathcal{I} = \mathcal{O} = \{0, 1\}$ . Con el estado inicial  $s_0$ ,  $\omega(s_0, x) = 0$  para  $x = 0$  o  $1$  ya que la primera salida es 0; los estados  $s_1$  y  $s_2$  (en la Fig. 6.13) recuerdan una entrada anterior de 0 y 1, respectivamente. En la figura, podemos etiquetar, por ejemplo, el arco que va de  $s_1$  a  $s_2$  con 1, 0, ya que con una entrada de 1 necesitamos ir a  $s_2$ , donde se recuerdan las entradas de 1 en el momento  $t_i$  para que se puedan convertir en salidas de 1 en el momento  $t_{i+1}$ . El 0 en la etiqueta 1, 0 es la salida, ya que el inicio en  $s_1$  indica que la entrada anterior fue 0, que se convierte en la salida actual. Las etiquetas de los otros arcos se obtienen con el mismo tipo de razonamiento.

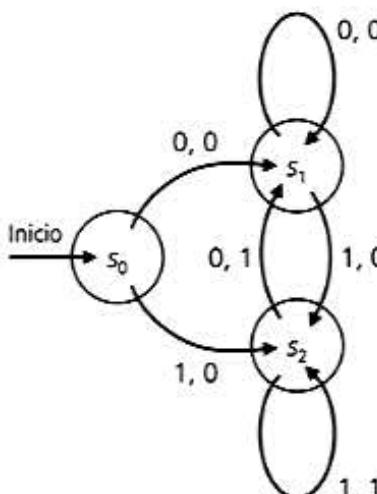


Figura 6.13

### Ejemplo 6.26

Si observamos la estructura para el retraso de una unidad, extendemos nuestras ideas a la máquina de retraso de dos unidades de la figura 6.14. Si  $x \in \mathcal{I}^*$ , sea  $x = x_1 x_2 \dots x_m$  donde  $m > 2$ ; si  $s_0$  es el estado inicial, entonces  $\omega(s_0, x) = 00x_1 \dots x_{m-2}$ . Para los estados  $s_0, s_1, s_2$  la salida es 0 para todas las entradas posibles. Los estados  $s_3, s_4, s_5$  y  $s_6$  deben recordar las dos entradas anteriores 00, 01, 10 y 11, respectivamente. Para obtener los otros arcos del diagrama, debemos considerar uno de ellos y después utilizar razonamientos similares para los demás. En el arco de  $s_5$  a  $s_3$  de la figura 6.14(a), consideraremos la entrada 0. Como la entrada anterior de  $s_5$  de  $s_2$  es 0, debemos ir al estado que recuerda las dos entradas anteriores 00. Éste es el estado  $s_3$ . Si regresamos dos estados, de  $s_5$  a  $s_2$  a  $s_0$ , vemos que la entrada es 1 (de  $s_0$  a  $s_2$ ). Esto se convierte entonces en la salida (retardada dos unidades) del arco de  $s_5$  a  $s_3$ . La máquina completa se muestra en la parte (b) de la figura 6.14.

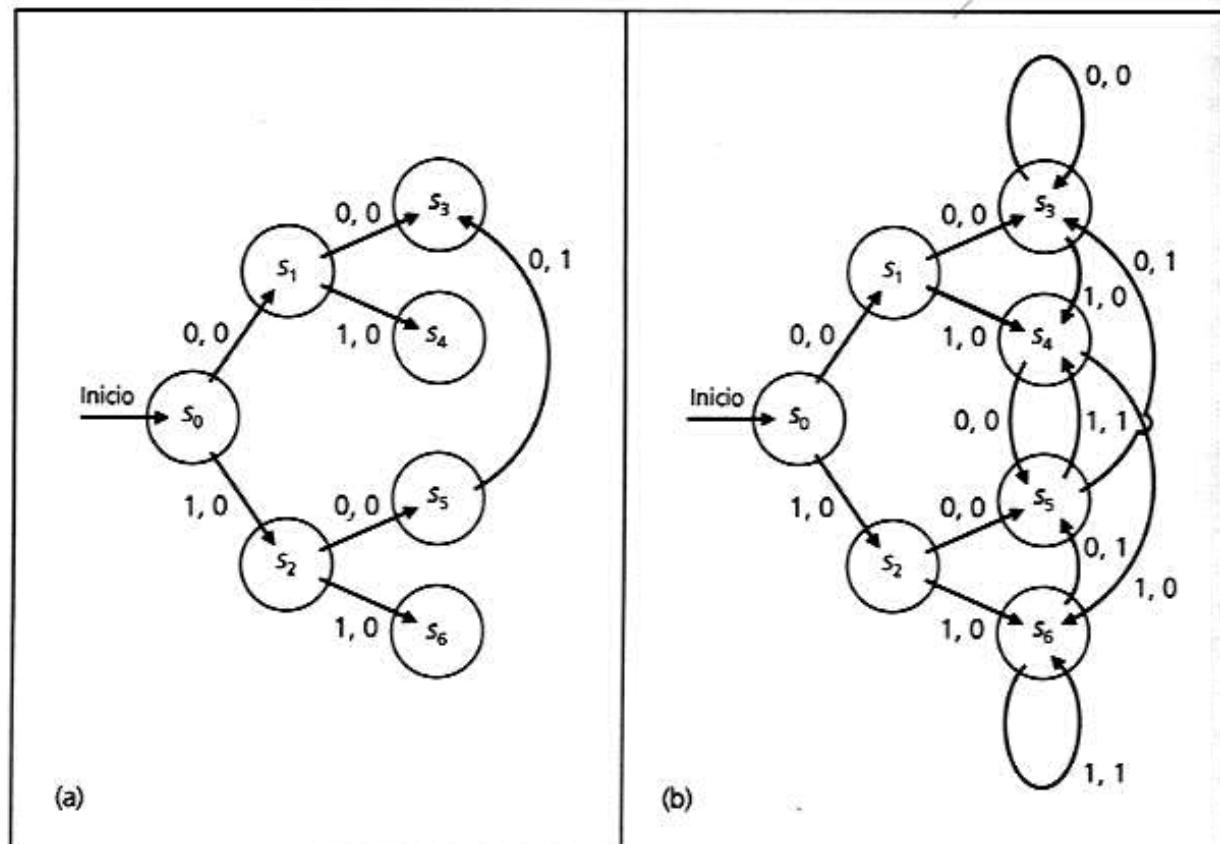


Figura 6.14

Regresemos ahora a algunas propiedades adicionales que surgen en el estudio de las máquinas de estados finitos. Usaremos la máquina de la figura 6.15 para los ejemplos de los términos definidos.

#### Definición 6.14

Sea  $M = (S, \mathcal{F}, \mathcal{O}, v, \omega)$  una máquina de estados finitos.

- Para  $s_i, s_j \in S$ , decimos que  $s_j$  se puede alcanzar desde  $s_i$  si  $s_i = s_j$  o si existe una cadena de entrada  $x \in \mathcal{F}^*$  tal que  $v(s_i, x) = s_j$ . (En la figura 6.15, podemos alcanzar el estado  $s_3$  desde  $s_0, s_1, s_2$  y  $s_3$  pero no desde  $s_4, s_5, s_6$ , o  $s_7$ . No podemos alcanzar ningún estado desde  $s_3$ , a excepción del propio  $s_3$ .)
- Un estado  $s \in S$  es *transitorio* si  $v(s, x) = s$  para  $x \in \mathcal{F}^*$  implica  $x = \lambda$ ; es decir, no existe  $x \in \mathcal{F}^+$  tal que  $v(s, x) = s$ . (Para la máquina de la figura 6.15,  $s_2$  es el único estado transitorio.)

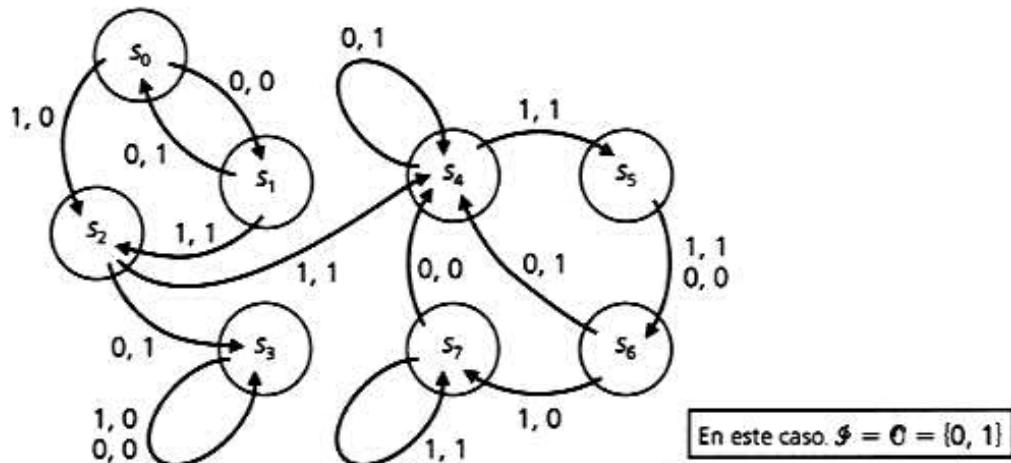


Figura 6.15

- c) Un estado  $s \in S$  es un *sumidero*, o *estado de sumidero*, si  $v(s, x) = s$ , para todo  $x \in \mathcal{G}^*$ . ( $s_3$  es el único sumidero de la figura 6.15.)
- d) Sea  $S_1 \subseteq S$ ,  $\mathcal{G}_1 \subseteq \mathcal{G}$ . Si  $v_1 = v|_{S_1 \times \mathcal{G}_1}: S_1 \times \mathcal{G}_1 \rightarrow S_1$  (es decir, la restricción de  $v$  a  $S_1 \times \mathcal{G}_1 \subseteq S \times \mathcal{G}$ ) tiene su imagen dentro de  $S_1$ , entonces con  $\omega_1 = \omega|_{S_1 \times \mathcal{G}_1}$ ,  $M_1 = (S_1, \mathcal{G}_1, \emptyset, v_1, \omega_1)$  es una *submáquina* de  $M$ . (Con  $S_1 = \{s_4, s_5, s_6, s_7\}$ , e  $\mathcal{G}_1 = \{0, 1\}$ , obtenemos una submáquina  $M_1$  de la máquina  $M$  de la figura 6.15.)
- e) Una máquina es *fuertemente conexa* si para cualesquiera estados  $s_i, s_j \in S$ , podemos alcanzar  $s_j$  desde  $s_i$ . (La máquina de la figura 6.15 no es fuertemente conexa, pero la submáquina  $M_1$  de la parte (d) tiene esta propiedad.)

Concluimos esta sección con un resultado que utiliza un diagrama de árbol.

### Definición 6.15

Para una máquina de estados finitos  $M$ , sean  $s_i, s_j$  dos estados distintos en  $S$ . La cadena de entrada más corta  $x \in \mathcal{G}^*$  es una *secuencia de transferencia* (o *transición*) desde  $s_i$  a  $s_j$  si

- a)  $v(s_i, x) = s_j$ , y  
 b)  $y \in \mathcal{G}^*$  con  $v(s_i, y) = s_j \Rightarrow \|y\| \geq \|x\|$ .

Puede haber más de una secuencia de este tipo para dos estados  $s_i, s_j$ .

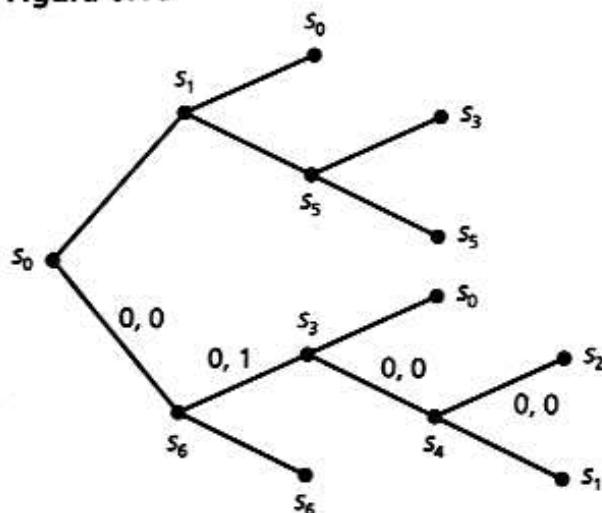
### Ejemplo 6.27

Encontraremos una secuencia de transferencia desde el estado  $s_0$  hasta el estado  $s_2$  para la máquina de estados finitos  $M$  dada por la tabla de estados de la tabla 6.11, donde  $\mathcal{G} = \emptyset = \{0, 1\}$ .

Tabla 6.11

	$v$		$\omega$	
	0	1	0	1
$s_0$	$s_6$	$s_1$	0	1
$s_1$	$s_5$	$s_0$	0	1
$s_2$	$s_1$	$s_2$	0	1
$s_3$	$s_4$	$s_0$	0	1
$s_4$	$s_2$	$s_1$	0	1
$s_5$	$s_3$	$s_5$	1	1
$s_6$	$s_3$	$s_6$	1	1

Figura 6.16



Al construir el diagrama de árbol de la figura 6.16, partimos del estado  $s_0$  y encontramos aquellos estados que podemos alcanzar desde  $s_0$  utilizando cadenas de longitud uno. En este caso encontramos  $s_1$  y  $s_6$ . Luego hacemos lo mismo con  $s_1$  y  $s_6$ , para encontrar, como resultado, los estados que podemos alcanzar desde  $s_0$  mediante cadenas de entrada de longitud dos. Si seguimos desarrollando el árbol de izquierda a derecha, llegamos a un vértice etiquetado con el estado deseado,  $s_2$ . Cada vez que alcanzamos un vértice etiquetado con un estado utilizado anteriormente, terminamos esa parte del desarrollo, pues ya no podemos alcanzar más estados nuevos. Después de llegar al estado que deseamos, regresamos a  $s_0$ .

y utilizamos la tabla de estados para etiquetar las ramas, como se muestra en la figura 6.16. Por lo tanto, para  $x = 0000$ ,  $v(s_0, x) = s_2$  con  $\omega(s_0, x) = 0100$ . (En este caso,  $x$  es único.)

### EJERCICIOS 6.3

- Sean  $\mathcal{S} = \emptyset = \{0, 1\}$ . (a) Construya un diagrama de estados para una máquina de estados finitos que reconozca cada ocurrencia de 0000 en una cadena  $x \in \mathcal{S}^*$ . (Se permite el solapamiento.) (b) Construya un diagrama de estados para una máquina de estados finitos que reconozca cada cadena  $x \in \mathcal{S}^*$  que termine en 0000 y que tenga una longitud de  $4k$ ,  $k \in \mathbb{Z}^+$ . (No se permite el solapamiento.)
- Resuelva el ejercicio 1 para cada una de las sucesiones 0110 y 1010.
- Construya un diagrama de estados para una máquina de estados finitos con  $\mathcal{S} = \emptyset = \{0, 1\}$  que reconozca todas las cadenas del lenguaje  $\{0, 1\}^*\{00\} \cup \{0, 1\}^*\{11\}$ .
- Para  $\mathcal{S} = \emptyset = \{0, 1\}$ , una cadena  $x \in \mathcal{S}^*$  tiene *paridad par* si contiene un número par de unos. Construya un diagrama de estados para una máquina de estados finitos que reconozca todas las cadenas no vacías de paridad par.
- La tabla 6.12 define  $v$  y  $\omega$  para una máquina de estados finitos  $M$  donde  $\mathcal{S} = \emptyset = \{0, 1\}$ .

Tabla 6.12

	$v$		$\omega$	
	0	1	0	1
$s_0$	$s_0$	$s_1$	0	0
$s_1$	$s_0$	$s_1$	1	1

- Trace el diagrama de estados para  $M$ .
- Determine la salida para las siguientes secuencias de entrada, comenzando en  $s_0$  en cada caso:
  - $x = 111$ ; (ii)  $x = 1010$ ; (iii)  $x = 00011$ .
- Describa con palabras lo que hace la máquina  $M$ .
- ¿Cómo se relaciona esta máquina con la de la figura 6.13?
- Muestre que no es posible construir una máquina de estados finitos que reconozca precisamente aquellas secuencias del lenguaje  $A = \{0^i, 1^j \mid i, j \in \mathbb{Z}^+, i > j\}$ . (El alfabeto de  $A$ , en este caso, es  $\Sigma = \{0, 1\}$ .)
- Para cada una de las máquinas de la tabla 6.13, determine los estados transitorios, los estados de sumidero, las submáquinas (donde  $\mathcal{S}_1 = \{0, 1\}$ ) y las submáquinas fuertemente conexas (donde  $\mathcal{S}_1 = \{0, 1\}$ ).

Tabla 6.13

	$v$		$\omega$	
	0	1	0	1
$s_0$	$s_4$	$s_1$	0	0
$s_1$	$s_4$	$s_2$	0	1
$s_2$	$s_3$	$s_5$	0	0
$s_3$	$s_2$	$s_5$	1	0
$s_4$	$s_4$	$s_4$	1	1
$s_5$	$s_2$	$s_3$	0	1

(a)

	$v$		$\omega$	
	0	1	0	1
$s_0$	$s_0$	$s_1$	1	0
$s_1$	$s_0$	$s_1$	0	1
$s_2$	$s_1$	$s_3$	0	0
$s_3$	$s_0$	$s_4$	0	0
$s_4$	$s_4$	$s_4$	1	1

(b)

	$v$		$\omega$	
	0	1	0	1
$s_0$	$s_1$	$s_2$	0	1
$s_1$	$s_0$	$s_2$	1	1
$s_2$	$s_2$	$s_3$	1	1
$s_3$	$s_6$	$s_4$	0	0
$s_4$	$s_5$	$s_5$	1	0
$s_5$	$s_3$	$s_4$	1	0
$s_6$	$s_6$	$s_6$	0	0

(c)

8. Determine una secuencia de transferencia del estado  $s_2$  al estado  $s_5$  en la máquina de estados finitos (c) del ejercicio 7. ¿Es esta secuencia única?

## 6.4

### Resumen y repaso histórico

En este capítulo vimos una introducción a la teoría de lenguajes y a una estructura discreta llamada *máquina de estados finitos*. Con nuestro desarrollo anterior de la teoría elemental de los conjuntos y las funciones finitas, pudimos conjuntar algunas nociones abstractas y modelar dispositivos digitales como los reconocedores y retardadores de secuencias. Un tratamiento similar de este material aparece en el capítulo 1 de L. Dornhoff y F. Hohn [3], así como en el capítulo 2 de D. F. Stanat y D. F. McAllister [14].

La máquina de estados finitos que hemos desarrollado se basa en el modelo planteado en 1955 por G. H. Mealy en [11], por lo que se conoce como la “máquina de Mealy”. El modelo se basa en conceptos anteriores que aparecen en la obra de D. A. Huffman [8] y E. F. Moore [12]. Para profundizar en el estudio de los primeros trabajos relacionados con diferentes aspectos y aplicaciones de la máquina de estados finitos, consulte el material editado por E. F. Moore [13]. En los capítulos 9 al 15 de Z. Kohavi [9] puede encontrarse más información acerca de la síntesis actual de tales máquinas y las consideraciones relativas a ellas en hardware, junto con un análisis amplio de muchas ideas relacionadas con el tema.

Para más detalles acerca de los lenguajes y su relación con las máquinas de estados finitos, hay que analizar el módulo UMAP de William J. Barnier [1], los capítulos 7 al 10 de J. L. Gersting [4] y los capítulos 7 y 8 de A. Gill [5]. Una amplia cobertura de estos temas (y otros relacionados con ellos) aparece en los textos de J. G. Brookshear [2], J. E. Hopcroft y J. D. Ullman [7], H. R. Lewis y C. H. Papadimitriou [10], y D. Wood [15].

Uno podría sorprenderse al saber que las ideas básicas de la teoría de autómatas se desarrollaron para resolver cuestiones más bien teóricas de los fundamentos de las matemáticas, como lo estableció en 1900 el matemático alemán David Hilbert (1862–1943). En 1935, el matemático y lógico inglés Alan Mathison Turing (1912–1954) se interesó en



David Hilbert (1862–1943)



**Alan Mathison Turing (1912–1954)**

Cortesía de The Granger Collection, Nueva York

el problema de decisión de Hilbert, que preguntaba si podría haber un método general aplicable a cualquier enunciado para determinar si éste era verdadero. El enfoque de Turing para la solución de este problema lo llevó a desarrollar lo que ahora se conoce como la *máquina de Turing*, el modelo más general de una máquina de cómputo. Con este modelo pudo establecer resultados teóricos muy profundos acerca de la forma en que tendrían que funcionar los computadores, antes de que fueran construidos realmente. Durante la Segunda Guerra Mundial, Turing trabajó en la oficina para asuntos externos de Bletchley Park, donde hizo un amplio uso del criptoanálisis de los mensajes nazis. Sus esfuerzos produjeron una máquina descifradora mecánica, *Enigma*, elemento muy importante que contribuyó a la caída del Tercer Reich. Después de la guerra y hasta su muerte, el interés de Turing acerca de la capacidad de las máquinas para pensar lo llevó a desempeñar un papel preponderante en el desarrollo de los computadores reales (no sólo teóricos). Para más detalles acerca de la vida de este importante académico, consultese la bibliografía de A. Hodges [6].

## BIBLIOGRAFÍA

1. Barnier, William J., "Finite-State Machines as Recognizers" (Módulo UMAP 671), *The UMAP Journal* 7, núm. 3 (1986), págs. 209–232.
2. Brookshear, J. Glenn, *Theory of Computation: Formal Languages, Automata, and Complexity*, Reading, Mass., Benjamin/Cummings, 1989.
3. Dornhoff, Larry L. y Franz E. Hohn, *Applied Modern Algebra*, Nueva York, Macmillan, 1978.
4. Gersting, Judith L., *Mathematical Structures for Computer Science*, San Francisco, W. H. Freeman, 1982.
5. Gill, Arthur, *Applied Algebra for the Computer Sciences*, Prentice-Hall Series in Automatic Computation, Englewood Cliffs, N.J., Prentice-Hall, 1976.
6. Hodges, Andrew, *Alan Turing: The Enigma*, Nueva York, Simon and Schuster, 1983.
7. Hopcroft, John E. y Jeffrey D. Ullman, *Introduction to Automata Theory, Languages and Computation*, Reading, Mass., Addison-Wesley, 1979.

8. Huffman, D. A., "The Synthesis of Sequential Switching Circuits", *Journal of the Franklin Institute* 257 (Marzo de 1954), págs. 161–190; (Abril de 1954), págs. 275–303. Reimpreso en Moore [13].
9. Kohavi, Zvi, *Switching and Finite Automata Theory*, 2<sup>a</sup> ed., Nueva York, Mc Graw-Hill, 1978.
10. Lewis, Harry R. y Christos H. Papadimitriou, *Elements of the Theory of Computation*, Englewood Cliffs, N.J., Prentice-Hall, 1981.
11. Mealy, G. H., "A Method for Synthesizing Sequential Circuits", *Bell System Technical Journal* 34 (Septiembre de 1955), págs. 1045–1079.
12. Moore, E. F., "Gedanken-experiments on Sequential Machines", *Automata Studies, Annals of Mathematical Studies*, núm. 34, págs. 129–153, Princeton, N.J., Princeton University Press, 1956.
13. Moore, E. F., editor, *Sequential Machines. Selected Papers*, Reading, Mass., Addison-Wesley, 1964.
14. Stanat, Donald F. y David F. McAllister, *Discrete Mathematics in Computer Science*, Englewood Cliffs, N.J., Prentice-Hall, 1977.
15. Wood, Derick, *Theory of Computation*, Nueva York, Wiley, 1987.

### EJERCICIOS COMPLEMENTARIOS

1. Sean  $\Sigma_1 = \{w, x, y\}$  y  $\Sigma_2 = \{x, y, z\}$  dos alfabetos. Si  $A_1 = \{x^i y^j | i, j \in \mathbb{Z}^*, j > i \geq 1\}$ ,  $A_2 = \{w^i y^j | i, j \in \mathbb{Z}^*, i > j \geq 1\}$ ,  $A_3 = \{w^i x^j y^k z^l | i, j, k, l \in \mathbb{Z}^*, i > j \geq 1, k > l \geq 1\}$ , y  $A_4 = \{z^i (wz)^j w^l | i, j, l \in \mathbb{Z}^*, i \geq 1, j \geq 2\}$ , determine si cada una de las siguientes proposiciones son verdaderas o falsas.

- $A_1$  es un lenguaje sobre  $\Sigma_1$ .
- $A_1$  es un lenguaje sobre  $\Sigma_2$ .
- $A_2$  es un lenguaje sobre  $\Sigma_1$ .
- $A_2$  es un lenguaje sobre  $\Sigma_2$ .
- $A_3$  es un lenguaje sobre  $\Sigma_1 \cup \Sigma_2$ .
- $A_1$  es un lenguaje sobre  $\Sigma_1 \cap \Sigma_2$ .
- $A_4$  es un lenguaje sobre  $\Sigma_1 \Delta \Sigma_2$ .
- $A_1 \cup A_2$  es un lenguaje sobre  $\Sigma_1$ .

2. Si  $A, B \subseteq \Sigma^*$ , ¿se cumple que  $A^* \subseteq B^* \Rightarrow A \subseteq B$ ?

3. Dé un ejemplo de lenguaje  $A$  sobre un alfabeto  $\Sigma$  tal que  $(A^2)^* \neq (A^*)^2$ .

4. Para un alfabeto dado  $\Sigma$  y un conjunto de índices  $I$ , sea  $B_i \subseteq \Sigma^*$  para cada  $i \in I$ . Si  $A \subseteq \Sigma^*$ , demuestre que (a)  $A(\bigcap_{i \in I} B_i) \subseteq \bigcap_{i \in I} AB_i$ ; y (b)  $(\bigcap_{i \in I} B_i)A \subseteq \bigcap_{i \in I} B_i A$ . [En este caso, por ejemplo,  $A(\bigcap_{i \in I} B_i)$  denota la concatenación de los lenguajes  $A$  y  $\bigcap_{i \in I} B_i$ .]

5. Sea  $M$  la máquina de estados finitos que se muestra en la figura 6.17. Para los estados  $s_i, s_j$ ,  $0 \leq i, j \leq 2$ , sea  $O_{ij}$  el conjunto de todas las cadenas de salida no vacías que  $M$  puede producir al ir del estado  $s_i$  al estado  $s_j$ . Si  $i = 2, j = 0$ , por ejemplo,  $O_{20} = \{0\} \{1, 00\}^*$ .

Encuentre  $O_{02}, O_{22}, O_{11}, O_{00}$ , y  $O_{10}$ .

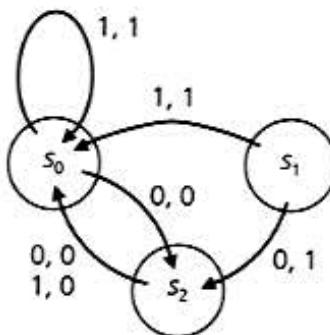


Figura 6.17

6. Sea  $M$  la máquina de estados finitos que se muestra en la figura 6.18.

- Encuentre la tabla de estados para esta máquina.
- Explique lo que hace esta máquina.
- ¿Cuántas cadenas de entrada distintas  $x$  cumplen que  $\|x\| = 8$  y  $v(s_0, x) = s_0$ ? ¿Cuántas cumplen  $\|x\| = 12$ ?

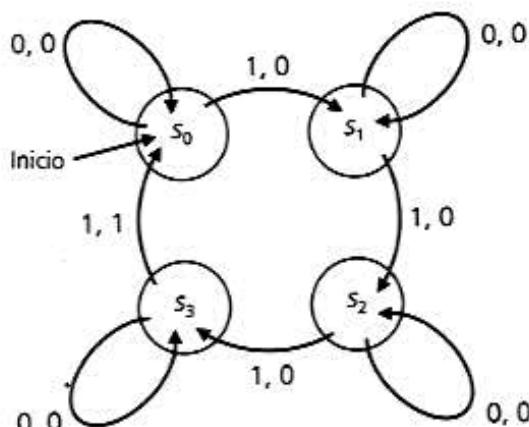


Figura 6.18

7. Sea  $M = (S, \mathcal{G}, \mathcal{O}, v, \omega)$  una máquina de estados finitos con  $|S| = n$ , y sea  $0 \in \mathcal{G}$ .

- Muestre que para cada cadena de entrada  $0000 \dots$ , la salida es finalmente periódica.
- ¿Cuál es el máximo número de ceros que podemos dar como entrada antes de que comience la salida periódica?
- ¿Cuál es la longitud del máximo periodo posible?

8. Trace el diagrama de estados para una máquina de estados finitos  $M = (S, \mathcal{G}, \mathcal{O}, v, \omega)$ , donde  $\mathcal{G} = \mathcal{O} = \{0, 1\}$ , si para cualquier  $x \in \mathcal{G}^*$ ,  $M$  pone un primer 1 cuando reconoce la subcadena 1111 y después pone un segundo 1 cuando reconoce la subcadena 0000, después de lo cual su salida es constante e igual a 0.

9. Sea  $\mathcal{G} = \mathcal{O} = \{0, 1\}$ . Construya un diagrama de estados para una máquina de estados finitos que invierta (de 0 a 1 o de 1 a 0) los símbolos que aparecen en la  $4^{\text{a}}$ , en la  $8^{\text{a}}$ , en la  $12^{\text{a}}$ , ..., posiciones de una cadena de entrada  $x \in \mathcal{G}^*$ . Por ejemplo, si  $s_0$  es el estado inicial, entonces  $\omega(s_0, 0000) = 0001$ ,  $\omega(s_0, 000111) = 000011$ , y  $\omega(s_0, 000000111) = 000100101$ .

10. Para  $\mathcal{G} = \mathcal{O} = \{0, 1\}$ , sea  $M$  la máquina de estados finitos dada en la tabla 6.14. Si el estado inicial de  $M$  no es  $s_1$ , encuentre una cadena de entrada  $x$  (de longitud mínima) tal que  $v(s_i, x) = s_1$ , para todo  $i = 2, 3, 4$ . (Por lo tanto,  $x$  lleva la máquina  $M$  al estado  $s_1$  independientemente del estado inicial.)

Tabla 6.14

	$v$		$\omega$	
	0	1	0	1
$s_1$	$s_4$	$s_3$	0	0
$s_2$	$s_2$	$s_4$	0	1
$s_3$	$s_1$	$s_2$	1	0
$s_4$	$s_1$	$s_4$	1	1

11. Muestre que no podemos construir una máquina de estados finitos que reconozca con precisión aquellas secuencias del lenguaje  $A = \{0^i 1^j \mid i, j \in \mathbb{Z}^+, i < j\}$ . (El alfabeto para  $A$  es  $\Sigma = \{0, 1\}$ .)

12. Si  $\mathcal{G} = \mathcal{O} = \{0, 1\}$ , sea  $M$  la máquina de estados finitos dada en la tabla 6.15. En este caso,  $s_0$  es el estado inicial. Sea  $A \subseteq \mathcal{G}^*$  el conjunto tal que  $x \in A$  si y sólo si el último símbolo en  $\omega(s_0, x)$  es 1. [Puede haber más de un 1 en la cadena de salida  $\omega(s_0, x)$ .] Construya una máquina de estados finitos donde el último símbolo de la cadena de salida sea 1 para todo  $y \in \mathcal{G}^* - A$ .

Tabla 6.15

	$v$		$\omega$	
	0	1	0	1
$s_0$	$s_1$	$s_2$	1	0
$s_1$	$s_2$	$s_1$	0	1
$s_2$	$s_2$	$s_3$	0	1
$s_3$	$s_1$	$s_0$	1	0

13. Sean  $\mathcal{G} = \mathcal{O} = \{0, 1\}$  para las dos máquinas de estados finitos  $M_1$  y  $M_2$  dadas en las tablas 6.16 y 6.17, respectivamente. El estado inicial para  $M_1$  es  $s_0$ , mientras que  $s_3$  es el estado inicial para  $M_2$ .

Tabla 6.16

	$v_1$		$\omega_1$	
	0	1	0	1
$s_0$	$s_0$	$s_1$	1	0
$s_1$	$s_1$	$s_2$	0	0
$s_2$	$s_2$	$s_0$	0	1

Tabla 6.17

	$v_2$		$\omega_2$	
	0	1	0	1
$s_3$	$s_3$	$s_4$	1	1
$s_4$	$s_4$	$s_3$	1	0

Conectamos estas máquinas como se muestra en la figura 6.19. Cada símbolo de salida de  $M_1$  se convierte en un símbolo de entrada para  $M_2$ . Por ejemplo, si damos 0 como entrada de  $M_1$ , entonces  $\omega_1(s_0, 0) = 1$  y  $v_1(s_0, 0) = s_0$ . Como resultado, damos como entrada 1 ( $= \omega_1(s_0, 0)$ ) a  $M_2$  para obtener  $\omega_2(s_3, 1) = 1$  y  $v_2(s_3, 1) = s_4$ .

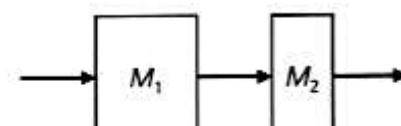


Figura 6.19

Construimos una máquina  $M = (S, \mathcal{G}, \mathcal{O}, v, \omega)$  que representa esta conexión de  $M_1$  y  $M_2$  como sigue:

$$\mathcal{G} = \mathcal{O} = \{0, 1\}.$$

$S = S_1 \times S_2$ , donde  $S_i$  es el conjunto de estados internos para  $M_i$ ,  $i = 1, 2$ .

$v: S \times \mathcal{G} \rightarrow S$ , donde

$v((s, t), x) = (v_1(s, x), v_2(t, \omega_1(s, x)))$ , para  $s \in S_1$ ,  $t \in S_2$ , y  $x \in \mathcal{G}$ .

$\omega: S \times \mathcal{G} \rightarrow \mathcal{O}$ , donde

$\omega((s, t), x) = \omega_2(t, \omega_1(s, x))$ , para  $s \in S_1$ ,  $t \in S_2$ , y  $x \in \mathcal{G}$ .

- a) Encuentre una tabla de estados para la máquina  $M$ .
- b) Determine la cadena de salida para la cadena de entrada 1101. ¿En qué estado está cada una de las máquinas  $M_1$  y  $M_2$  después de procesar esta cadena?

14. Aunque el diagrama de estados parece más conveniente que la tabla de estados cuando trabajamos con una máquina de estados finitos  $M = (S, \mathcal{I}, \mathcal{O}, \nu, \omega)$ , si las cadenas de entrada son cada vez más largas y los tamaños de  $S$ ,  $\mathcal{I}$  y  $\mathcal{O}$  se incrementan, la tabla de estados es útil para simular la máquina en un computador. La forma de bloque de la tabla sugiere el uso de una matriz o una tabla de dos dimensiones para almacenar  $\nu$ ,  $\omega$ . Utilice esta observación para es-

cribir un programa (o desarrolle un algoritmo) que simule la máquina de la tabla 6.18.

**Tabla 6.18**

	$\nu$		$\omega$	
	0	1	0	1
$s_1$	$s_2$	$s_1$	0	0
$s_2$	$s_3$	$s_1$	0	0
$s_3$	$s_3$	$s_1$	1	1

33. a)  $\tau(n) = (e_1 + 1)(e_2 + 1) \cdots (e_k + 1)$   
 b)  $k = 2: \tau(2) = \tau(3) = \tau(5) = 2$        $k = 3: \tau(2^2) = \tau(3^2) = \tau(5^2) = 3$   
 $k = 4: \tau(6) = \tau(8) = \tau(10) = 4$        $k = 5: \tau(2^4) = \tau(3^4) = \tau(5^4) = 5$   
 $k = 6: \tau(12) = \tau(18) = \tau(20) = 6$   
 c) Para todo  $k > 1$  y todos los primos  $p$ ,  $\tau^{-1}(p^{k-1}) = k$ .
35. a)  $S(8, 4)$       b)  $S(n, m)$
37. a) Sean  $m = 1$  y  $k = 1$ . Entonces, para cualquier  $n \geq k$ ,  $|f(n)| \leq 2 < 3 \leq |g(n)| = m|g(n)|$ , por lo que  $f \in O(g)$ .
39. Observemos primero que si  $\log_a n = r$ , entonces  $n = a^r$  y  $\log_b n = \log_b(a^r) = r \log_b a = (\log_b a)(\log_a n)$ . Ahora, sean  $m = (\log_b a)$  y  $k = 1$ . Entonces, para cualquier  $n \geq k$ ,  $|g(n)| = \log_b n = (\log_b a)(\log_a n) = m|f(n)|$ , por lo que  $g \in O(f)$ . Por último, si  $m = (\log_b a)^{-1} = \log_a b$  y  $k = 1$ , vemos que para cualquier  $n \geq k$ ,  $|f(n)| = \log_a n = (\log_b a)(\log_b n) = m|g(n)|$ . Por lo tanto,  $f \in O(g)$ .

## Capítulo 6

### Lenguajes: Máquinas de estados finitos

Sección 6.1—  
pág. 38

1. a) 25; 125      b) 3906      3. 12      5. 780
7. a)  $\{00, 11, 000, 111, 0000, 1111\}$       b)  $\{0, 1\}$   
 c)  $\Sigma^* - \{\lambda, 00, 11, 000, 111, 0000, 1111\}$       d)  $\{0, 1, 00, 11\}$   
 e)  $\{00, 01, 10, 11\} = \{w \mid \|w\| = 2\}$       f)  $\Sigma^*$   
 g)  $\Sigma^* - \{0, 1, 00, 11\} = \{\lambda, 01, 10\} \cup \{w \mid \|w\| \geq 3\}$   
 h)  $\Sigma^* - \{\lambda, 0, 1, 00, 01, 10, 11, 000, 111, 0000, 1111\}$   
 i)  $\Sigma^* - \{\lambda, 0, 1, 00, 11, 000, 111, 0000, 1111\}$
9. a)  $x \in AC \Rightarrow x = ac$  para algún  $a \in A$ ,  $c \in C \Rightarrow x \in BD$ , pues  $A \subseteq B$ ,  $C \subseteq D$   
 b) Si  $A \emptyset \neq \emptyset$ , sea  $x \in A \emptyset$ .  $x \in A \emptyset \Rightarrow x = yz$  para algún  $y \in A$ ,  $z \in \emptyset$ . Pero  $z \in \emptyset$  es imposible.  
 Entonces  $A \emptyset \neq \emptyset$ . [De manera similar,  $\emptyset A = \emptyset$ .]
11. La única forma en que esto puede ocurrir es que  $A = \{\lambda\}$ .
13. a) En este caso,  $A^*$  consta de todas las cadenas  $x$  de longitud par, donde si  $x \neq \lambda$ , entonces  $x$  comienza con 0 y termina con 1, y los símbolos (0 y 1) se alternan.  
 b) En este caso,  $A^*$  consta precisamente de aquellas cadenas formadas por  $3n$  ceros, para  $n \in \mathbb{N}$ .  
 c) Aquí, una cadena  $x \in A^*$  si (y sólo si)  
 (i)  $x$  es una cadena de  $n$  ceros, para  $n \in \mathbb{N}$ ; o  
 (ii)  $x$  es una cadena que comienza y termina en cero, y tiene al menos un 1 y al menos dos ceros entre dos unos cualesquier.
15. Sea  $\Sigma$  un alfabeto con  $\emptyset \neq A \subseteq \Sigma^*$ . Si  $|A| = 1$  y  $x \in A$ , entonces  $xx = x$ , pues  $A^2 = A$ . Pero  $\|xx\| = 2\|x\| = \|x\| \Rightarrow \|x\| = 0 \Rightarrow x \in \lambda$ . Si  $|A| > 1$ , sea  $x \in A$  tal que  $\|x\| > 0$  pero  $\|x\|$  es minimal. Entonces  $x \in A^2 \Rightarrow x = yz$ , para  $y, z \in A$ . Como  $\|x\| = \|y\| + \|z\|$ , si  $\|y\|, \|z\| > 0$ , entonces una de las cadenas  $y, z$  está en  $A$  con una longitud menor que  $\|x\|$ . En consecuencia, uno de los valores  $\|y\|$  o  $\|z\|$  es cero y  $\lambda \in A$ .
17. Si  $A = A^2$ , entonces se sigue por inducción matemática que  $A = A^n$  para todo  $n \in \mathbb{Z}^*$ . Por lo tanto,  $A = A^*$ . Por el ejercicio 15,  $A = A^2 \Rightarrow \lambda \in A$ . Por lo tanto,  $A = A^*$ .
19. Por la definición 6.11,  $AB = \{ab \mid a \in A, b \in B\}$  y como es posible tener  $a_1 b_1, a_2 b_2$  con  $a_1, a_2 \in A$ ,  $a_1 \neq a_2$  y  $b_1, b_2 \in B$ ,  $b_1 \neq b_2$  se sigue que  $|AB| \leq |A \times B| = |A||B|$ .
21. a)  $x \in A(\bigcup_{i \in I} B_i) \Leftrightarrow x = ab$  para algún  $a \in A$ ,  $b \in \bigcup_{i \in I} B_i \Leftrightarrow x = ab$ , donde  $a \in A$  y  $b \in B_i$ , para algún  $i \in I \Leftrightarrow x \in AB_i$  para algún  $i \in I \Leftrightarrow \bigcup_{i \in I} AB_i$ .
23. a) Las palabras 001 y 011 tienen longitud 3 y están en  $A$ . Las palabras 00011 y 00111 tienen longitud 5 y también están en  $A$ .  
 b) Del paso (1) sabemos que  $1 \in A$ . Entonces, aplicando el paso (2) tres veces, obtenemos  
 (i)  $1 \in A \Rightarrow 011 \in A$ ;

- (ii)  $011 \in A \Rightarrow 00111 \in A$ ; y  
 (iii)  $00111 \in A \Rightarrow 0001111 \in A$ .

c) Si  $00001111$  estuviera en  $A$ , entonces, del paso (2) vemos que esta palabra sería generada por  $000111$  (en  $A$ ). De la misma forma,  $000111$  en  $A \Rightarrow 0011$  está en  $A \Rightarrow 01$  está en  $A$ . Sin embargo, en  $A$  no hay palabras de longitud 2; de hecho, no hay palabras de longitud par en  $A$ .

## 25. a) Pasos

- 1)  $()$  está en  $A$ .  
 2)  $(( ))$  está en  $A$ .  
 3)  $(( ))()$  está en  $A$ .

## Razones

- Parte (1) de la definición recursiva  
 Paso (1) y parte (2-ii) de la definición  
 Pasos (1) y (2) y parte (2-i) de la definición

## b) Pasos

- 1)  $()$  está en  $A$ .  
 2)  $(( ))$  está en  $A$ .  
 3)  $(( ))()$  está en  $A$ .  
 4)  $(( ))()()$  está en  $A$ .

## Razones

- Parte (1) de la definición recursiva  
 Paso (1) y parte (2-ii) de la definición  
 Pasos (1) y (2) y parte (2-i) de la definición  
 Pasos (1) y (3) y parte (2-i) de la definición

27. (1)  $\lambda \in A$  y  $s \in A$  para todo  $s \in \Sigma$ ; y

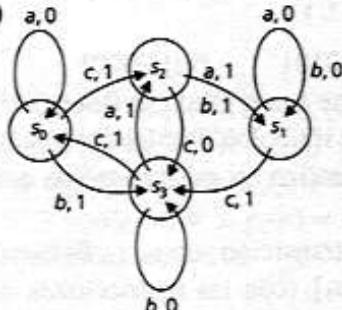
(2) Para cada  $x \in A$  y  $s \in \Sigma$ , la cadena  $sxs$  está también en  $A$ .

[Ninguna otra cadena de  $\Sigma^*$  está en  $A$ .]

Sección 6.2—  
pág. 333

1. a) 0010101;  $s_1$       b) 0000000;  $s_1$       c) 001000000;  $s_0$

3. a) 010110      b)



5. a) 010000;  $s_2$

b)  $(s_1) 100000; s_2$

$(s_2) 000000; s_2$

$(s_3) 110010; s_2$

	$v$		$w$	
	0	1	0	1
$s_0$	$s_0$	$s_1$	0	0
$s_1$	$s_1$	$s_2$	1	1
$s_2$	$s_2$	$s_2$	0	0
$s_3$	$s_0$	$s_3$	0	1
$s_4$	$s_2$	$s_3$	0	1

d)  $s_1$       e)  $x = 101$  (único)

7. a) (i) 15      (ii)  $3^{15}$       (iii)  $2^{15}$       b)  $6^{15}$

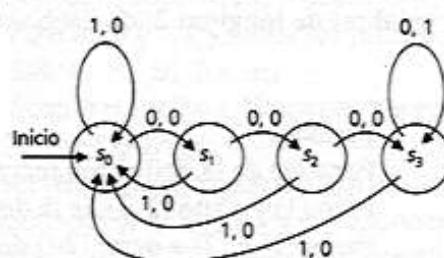
9. a)

	$v$		$w$	
	0	1	0	1
$s_0$	$s_4$	$s_1$	0	0
$s_1$	$s_3$	$s_2$	0	0
$s_2$	$s_3$	$s_2$	0	1
$s_3$	$s_3$	$s_3$	0	0
$s_4$	$s_5$	$s_3$	0	0
$s_5$	$s_5$	$s_3$	1	0

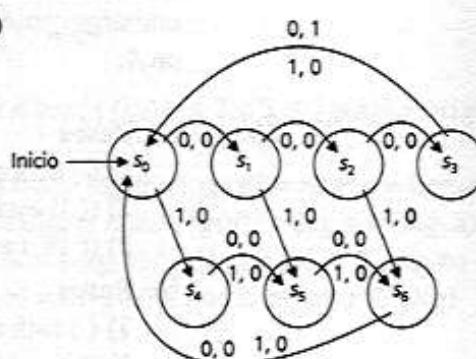
- b) Sólo hay dos posibilidades:  $x = 1111$  o  $x = 0000$ .  
 c)  $A = \{111\}\{1\}^* \cup \{000\}\{0\}^*$   
 d) En este caso,  $A = \{11111\}\{1\}^* \cup \{00000\}\{0\}^*$ .

Sección 6.3—  
pág. 342

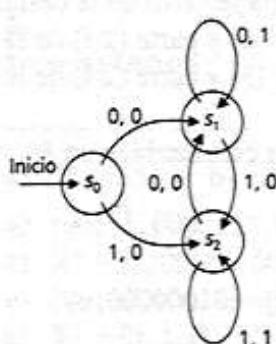
1. a)



b)



3.



5. b) (i) 011      (ii) 0101      (iii) 00001

c) La máquina tiene como salida un 0 seguido por los primeros  $n - 1$  símbolos de la cadena de entrada  $x$  con  $n$  símbolos. Por lo tanto, la máquina es un retraso unitario.

d) Esta máquina realiza las mismas tareas que la de la figura 6.13 (pero sólo tiene dos estados).

7. a) Los estados de transición son  $s_0, s_1$ . El estado  $s_4$  es un estado de sumidero.  $\{s_1, s_2, s_3, s_4, s_5\}$ ,  $\{s_4\}$  y  $\{s_2, s_3, s_5\}$  (con las restricciones correspondientes sobre la función dada  $v$ ) son submáquinas. Las submáquinas fuertemente conexas son  $\{s_4\}$  y  $\{s_2, s_3, s_5\}$ .  
 b) Los estados  $s_2, s_3$  son de transición. El único estado de sumidero es  $s_4$ . El conjunto  $\{s_0, s_1, s_3, s_4\}$  proporciona los estados de una submáquina;  $\{s_0, s_1\}$  y  $\{s_4\}$  son submáquinas fuertemente conexas.

Ejercicios  
complementarios—  
pág. 345

1. a) Verdadera    b) Verdadera    c) Verdadera    d) Falsa    e) Verdadera

f) Verdadera    g) Verdadera    h) Verdadera

3. Sean  $x \in \Sigma$  y  $A = \{x\}$ . Entonces  $A^2 = \{xx\}$  y  $(A^2)^* = \{\lambda, x^2, x^4, \dots\}$ .  
 $A^* = \{\lambda, x, x^2, x^3, \dots\}$  y  $(A^*)^2 = A^*$ , de modo que  $(A^2)^* \neq (A^*)^2$ .

5.  $C_{02} = \{1, 00\}^*\{0\}$

$C_{22} = \{0\}\{1, 00\}^*\{0\}$

$C_{11} = \emptyset$

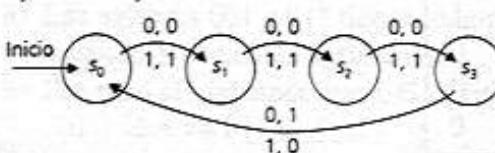
$C_{00} = \{1, 00\}^* - \{\lambda\}$

$C_{10} = \{1\}\{1, 00\}^* \cup \{10\}\{1, 00\}^*$

7. a) Por el principio del palomar, existe un primer estado  $s$  que se encuentra dos veces. Sea  $y$  la cadena de salida que resulta la primera vez que se encuentra  $s$ , hasta llegar a este estado por segunda vez. Entonces, a partir de ese punto, la salida es  $yyy\dots$

b)  $n$     c)  $n$

9.



11. Supongamos que podemos construir esa máquina y que tiene  $n$  estados, para  $n \in \mathbb{Z}^*$ . Para la cadena de entrada  $0^n 1^{n+1}$ , queremos que la salida sea  $0^{2n} 1$ . Sin embargo, al procesar los unos de esta cadena de entrada, obtenemos  $n + 1$  estados  $s_1, s_2, \dots, s_n, s_{n+1}$  de la función  $v$ . En consecuencia, por el principio del palomar, existen dos estados  $s_i, s_j$  tales que  $i < j$  pero  $s_i = s_j$ . Como resultado, si eliminamos los  $j - 1$  unos de los estados  $s_{i+1}, s_{i+2}, \dots, s_j$ , tenemos que la máquina reconoce la sucesión  $0^n 1^{n+1-(j-i)}$ , donde  $n + 1 - (j - i) \leq n$ . Sin embargo,  $0^n 1^{n+1-(j-i)} \notin A$ .

13. a)

	$\nu$		$\omega$	
	0	1	0	1
$(s_0, s_3)$	$(s_0, s_4)$	$(s_1, s_3)$	1	1
$(s_0, s_4)$	$(s_0, s_3)$	$(s_1, s_4)$	0	1
$(s_1, s_3)$	$(s_1, s_3)$	$(s_2, s_3)$	1	1
$(s_1, s_4)$	$(s_1, s_4)$	$(s_2, s_4)$	1	1
$(s_2, s_3)$	$(s_2, s_3)$	$(s_0, s_4)$	1	1
$(s_2, s_4)$	$(s_2, s_4)$	$(s_0, s_3)$	1	0

- b)  $\omega((s_0, s_3), 1101) = 1111$ ;  $M_1$  está en el estado  $s_0$  y  $M_2$  en el estado  $s_4$ .

## Capítulo 7

### Relaciones: La segunda vuelta

Sección 7.1—  
pág. 356

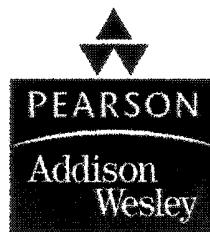
- a)  $\{(1, 1), (2, 2), (3, 3), (4, 4), (1, 2), (2, 1), (2, 3), (3, 2)\}$   
 b)  $\{(1, 1), (2, 2), (3, 3), (4, 4), (1, 2)\}$       c)  $\{(1, 1), (2, 2), (1, 2), (2, 1)\}$
- a) Sean  $f_1, f_2, f_3 \in \mathcal{F}$  con  $f_1(n) = n + 1$ ,  $f_2(n) = 5n$ , y  $f_3(n) = 4n + 1/n$ .  
 b) Sean  $g_1, g_2, g_3 \in \mathcal{F}$  con  $g_1(n) = 3$ ,  $g_2(n) = 1/n$  y  $g_3(n) = \operatorname{sen} n$ .
- a) Reflexiva, antisimétrica, transitiva  
 b) Transitiva  
 c) Reflexiva, simétrica, transitiva  
 d) Simétrica  
 e) Caso par: reflexiva, simétrica, transitiva; caso impar: simétrica.  
 f) Caso par: reflexiva, simétrica, transitiva; caso impar: simétrica.  
 g) Simétrica  
 h) Reflexiva, simétrica  
 i) Reflexiva, transitiva  
 j) Reflexiva, simétrica, transitiva
- a) Para todo  $x \in A$ ,  $(x, x) \in \mathcal{R}_1, \mathcal{R}_2$ , de modo que  $(x, x) \in \mathcal{R}_1 \cap \mathcal{R}_2$  y  $\mathcal{R}_1 \cap \mathcal{R}_2$  es reflexiva.  
 b) (i)  $(x, y) \in \mathcal{R}_1 \cap \mathcal{R}_2 \Rightarrow (x, y) \in \mathcal{R}_1, \mathcal{R}_2 \Rightarrow (y, x) \in \mathcal{R}_1, \mathcal{R}_2 \Rightarrow (y, x) \in \mathcal{R}_1 \cap \mathcal{R}_2$  y  $\mathcal{R}_1 \cap \mathcal{R}_2$  es simétrica.  
 (ii)  $(x, y), (y, x) \in \mathcal{R}_1 \cap \mathcal{R}_2 \Rightarrow (x, y), (y, x) \in \mathcal{R}_1, \mathcal{R}_2$ . Por la antisimetría de  $\mathcal{R}_1$  (o  $\mathcal{R}_2$ ),  $x = y$  y  $\mathcal{R}_1 \cap \mathcal{R}_2$  es antisimétrica.  
 (iii)  $(x, y), (y, z) \in \mathcal{R}_1 \cap \mathcal{R}_2 \Rightarrow (x, y), (y, z) \in \mathcal{R}_1, \mathcal{R}_2 \Rightarrow (x, z) \in \mathcal{R}_1, \mathcal{R}_2$  (propiedad transitiva)  
 $\Rightarrow (x, z) \in \mathcal{R}_1 \cap \mathcal{R}_2$ , de modo que  $\mathcal{R}_1 \cap \mathcal{R}_2$  es transitiva.
- a) Verdadera b) Falsa: sean  $A = \{1, 2\}$  y  $\mathcal{R} = \{(1, 2), (2, 1)\}$ .  
 c) (i) Reflexiva: verdadera  
 (ii) Simétrica: falsa. Sean  $A = \{1, 2\}$ ,  $\mathcal{R}_1 = \{(1, 1)\}$  y  $\mathcal{R}_2 = \{(1, 1), (1, 2)\}$ .  
 (iii) Antisimétrica y transitiva: falsa. Sean  $A = \{1, 2\}$ ,  $\mathcal{R}_1 = \{(1, 2)\}$  y  $\mathcal{R}_2 = \{(1, 2), (2, 1)\}$ .  
 d) (i) Reflexiva: falsa. Sean  $A = \{1, 2\}$ ,  $\mathcal{R}_1 = \{(1, 1)\}$  y  $\mathcal{R}_2 = \{(1, 1), (2, 2)\}$ .  
 (ii) Simétrica: falsa. Sean  $A = \{1, 2\}$ ,  $\mathcal{R}_1 = \{(1, 2)\}$  y  $\mathcal{R}_2 = \{(1, 2), (2, 1)\}$ .  
 (iii) Antisimétrica: verdadera

INSTRUCTOR'S  
SOLUTIONS MANUAL

DISCRETE AND  
COMBINATORIAL MATHEMATICS  
FIFTH EDITION

Ralph P. Grimaldi

*Rose-Hulman Institute of Technology*



Boston San Francisco New York  
London Toronto Sydney Tokyo Singapore Madrid  
Mexico City Munich Paris Cape Town Hong Kong Montreal

## CHAPTER 6

### LANGUAGES: FINITE STATE MACHINES

## Section 6.1

12.

(a) Yes

(d) Yes

(b) Yes

(e) No

(c) Yes

(f) Yes

13. (a) Here  $A^*$  consists of all strings  $x$  of even length where if  $x \neq \lambda$ , then  $x$  starts with 0 and ends with 1, and the symbols (0 and 1) alternate.  
 (b) In this case  $A^*$  contains precisely those strings made up of  $3n$  0's, for  $n \in \mathbb{N}$ .  
 (c) Here a string  $x \in A^*$  if (and only if)  
 (i)  $x$  is a string of  $n$  0's, for  $n \in \mathbb{N}$ ; or  
 (ii)  $x$  is a string that starts and ends with 0, and has at least one 1 – but no consecutive 1's.  
 (d) For this last case  $A^*$  consists of the following:  
 (i) Any string of  $n$  1's, for  $n \in \mathbb{N}$ ; and  
 (ii) Any string that starts with 1 and contains at least one 0, but no consecutive 0's.
14. There are five possible choices:  
 (1)  $A = \{\lambda\}, B = \{01, 000, 0101, 0111, 01000, 010111\}$ ;  
 (2)  $A = \{01, 000, 0101, 0111, 01000, 010111\}, B = \{\lambda\}$ ;  
 (3)  $A = \{0\}, B = \{1, 00, 101, 111, 1000, 10111\}$ ;  
 (4)  $A = \{0, 010\}, B = \{1, 00, 111\}$ ; and  
 (5)  $A = \{\lambda, 01\}, B = \{01, 000, 0111\}$ .
15. Let  $\Sigma$  be an alphabet with  $\emptyset \neq A \subseteq \Sigma^*$ . If  $|A| = 1$  and  $x \in A$ , then  $xx = x$  since  $A^2 = A$ . But  $\|xx\| = 2\|x\| = \|x\| \implies \|x\| = 0 \implies x = \lambda$ . If  $|A| > 1$ , let  $x \in A$  where  $\|x\| > 0$  but  $\|x\|$  is minimal. Then  $x \in A^2 \implies x = yz$ , for some  $y, z \in A$ . Since  $\|x\| = \|y\| + \|z\|$ , if  $\|y\|, \|z\| > 0$ , then one of  $y, z$  is in  $A$  with length smaller than  $\|x\|$ . Consequently, one of  $\|y\|$  or  $\|z\|$  is 0, so  $\lambda \in A$ .
16. (a)  $p_a, s_a, r$   
 (b)  $r, d$  are onto;  $p_a(\Sigma^*) = \{a\}\Sigma^*; s_a(\Sigma^*) = \Sigma^*\{a\}$   
 (c)  $r$  is invertible and  $r^{-1} = r$ .  
 (d)  $25; 125; 5^{n/2}$  for  $n$  even,  $5^{(n+1)/2}$  for  $n$  odd.  
 (e)  $(d \circ p_a)(x) = x = (r \circ d \circ r \circ s_a)(x)$   
 (f)  $r^{-1}(B) = \{ea, ia, oa, oo, oie, uuoie\}$   
 $p_a^{-1}(B) = \{e, i, o\}$ .  
 $s_a^{-1}(B) = \emptyset$   
 $|d^{-1}(B)| = |\bigcup_{x \in B} d^{-1}(x)| = \sum_{x \in B} d^{-1}(x) = \sum_{x \in B} 5 = 6(5) = 30$
17. If  $A = A^2$  then it follows by mathematical induction that  $A = A^n$  for all  $n \in \mathbb{Z}^+$ . Hence  $A = A^+$ . From Exercise 15 we know that  $A = A^2 \implies \lambda \in A$ , so  $A = A^*$ .
18. Theorem 6.1(b):  $x \in (AB)C \iff x = (ab)c$ , for some  $a \in A, b \in B, c \in C \iff x = (a_1 a_2 \dots a_\ell b_1 b_2 \dots b_m)(c_1 c_2 \dots c_n)$ , where  $a_i \in A, 1 \leq i \leq \ell, b_j \in B, 1 \leq j \leq m, c_k \in C, 1 \leq k \leq n \iff x = a_1 a_2 \dots a_\ell b_1 b_2 \dots b_m c_1 c_2 \dots c_n$ , where  $a_i \in A, 1 \leq i \leq \ell, b_j \in B, 1 \leq j \leq m, c_k \in C, 1 \leq k \leq n \iff x = (a_1 a_2 \dots a_\ell)(b_1 b_2 \dots b_m c_1 c_2 \dots c_n)$ , where

$a_i \in A, 1 \leq i \leq \ell$ ,  $b_j \in B, 1 \leq j \leq m$ ,  $c_k \in C, 1 \leq k \leq n \iff x \in A(BC)$ . Hence  $(AB)C = A(BC)$ .

Theorem 6.2(b): For  $a \in A$ ,  $a = \lambda a$  with  $\lambda \in B^*$ . Hence  $A \subseteq B^*A$ .

Theorem 6.2(f): From Theorem 6.2(a)  $A^* \subseteq A^*A^*$ . Conversely,  $x \in A^*A^* \implies x = yz$  where  $y = a_1a_2\dots a_m, z = a'_1a'_2\dots a'_n$ , with  $a_i, a'_j \in A$ , for  $1 \leq i \leq m, 1 \leq j \leq n$ . Hence  $x \in A^*$ , so  $A^*A^* \subseteq A^*$  and the equality follows.

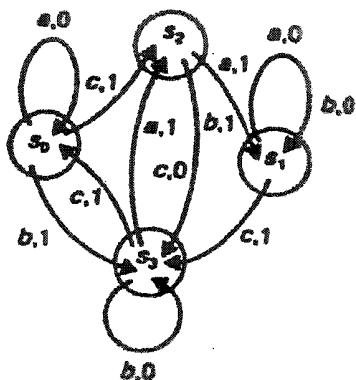
Since  $(A^*)^* = \bigcup_{n=0}^{\infty} (A^*)^n$ , it follows that  $A^* \subseteq (A^*)^*$ . Conversely, if  $x \in (A^*)^*$ , then  $x = x_1x_2\dots x_n$ , where  $x_i \in A^*$ , for  $1 \leq i \leq n$ . Each  $x_i = a_{i1}a_{i2}\dots a_{ik_i}$ , where  $a_{ij} \in A$ ,  $1 \leq j \leq k_i$ . Hence  $x \in A^*$ , so  $(A^*)^* \subseteq A^*$  and  $(A^*)^* = A^*$ .

$(A^*)^+ = \bigcup_{n=1}^{\infty} (A^*)^n \subseteq \bigcup_{n=0}^{\infty} (A^*)^n = (A^*)^*$ . If  $x \in (A^*)^*$ , then  $x = x_1x_2\dots x_n$ , where  $x_i \in A^*$ , for  $1 \leq i \leq n$ . Then  $x = a_{11}a_{12}\dots a_{1k_1}a_{21}a_{22}\dots a_{2k_2}\dots a_{n1}\dots a_{nk_n} \in A^* \subseteq \bigcup_{n=1}^{\infty} (A^*)^n = (A^*)^+$ , so  $(A^*)^* = (A^*)^+$ .

Since  $A^+ \subseteq A^*$ ,  $(A^+)^* \subseteq (A^*)^*$  by part (d) of this theorem. For  $x \in (A^*)^*$ , if  $x = \lambda$ , then  $x \in (A^+)^*$ . If  $x \neq \lambda$ , then as above  $x = a_{11}a_{12}\dots a_{1k_1}a_{21}\dots a_{2k_2}\dots a_{n1}\dots a_{nk_n} \in A^+ \subseteq (A^+)^*$  and the result follows.

- 23.**
- |   |   |
|---|---|
| <p>(a) <b>Steps</b></p> <ol style="list-style-type: none"> <li>1. <math>()</math> is in <math>A</math>.</li> <li>2. <math>(( ))</math> is in <math>A</math>.</li> <li>3. <math>(( ))( )</math> is in <math>A</math>.</li> </ol>   | <p><b>Reasons</b></p> <ol style="list-style-type: none"> <li>Part (1) of the recursive definition</li> <li>Step 1 and part (2(ii)) of the definition</li> <li>Steps 1, 2, and part (2(i)) of the definition</li> </ol>  |
| <p>(b) <b>Steps</b></p> <ol style="list-style-type: none"> <li>1. <math>()</math> is in <math>A</math>.</li> <li>2. <math>(( ))</math> is in <math>A</math>.</li> <li>3. <math>(( ))( )</math> is in <math>A</math>.</li> <li>4. <math>(( ))( )( )</math> is in <math>A</math>.</li> </ol>  | <p><b>Reasons</b></p> <ol style="list-style-type: none"> <li>Part (1) of the recursive definition</li> <li>Step 1 and part (2(ii)) of the definition</li> <li>Steps 1, 2, and part (2(i)) of the definition</li> <li>Steps 1, 3, and part (2(i)) of the definition</li> </ol> |
| <p>(c) <b>Steps</b></p> <ol style="list-style-type: none"> <li>1. <math>()</math> is in <math>A</math>.</li> <li>2. <math>( )( )</math> is in <math>A</math>.</li> <li>3. <math>(( )( ))</math> is in <math>A</math>.</li> <li>4. <math>(( )( )( ))</math> is in <math>A</math>.</li> </ol> | <p><b>Reasons</b></p> <ol style="list-style-type: none"> <li>Part (1) of the recursive definition</li> <li>Step 1 and part (2(i)) of the definition</li> <li>Step 2 and part (2(ii)) of the definition</li> <li>Steps 1, 3, and part (2(i)) of the definition</li> </ol>      |
- 24.** (1)  $\lambda \in A$  and  $s \in A$  for all  $s \in \Sigma$ ; and  
 (2) For each  $x \in A$  and  $s \in \Sigma$ , the string  $sxs$  is also in  $A$ .  
 [No other string from  $\Sigma^*$  is in  $A$ .]
- 25.** Length 3:  $\binom{3}{0} + \binom{2}{1} = 3$   
 Length 4:  $\binom{4}{0} + \binom{3}{1} + \binom{2}{2} = 5$   
 Length 5:  $\binom{5}{0} + \binom{4}{1} + \binom{3}{2} = 8$   
 Length 6:  $\binom{6}{0} + \binom{5}{1} + \binom{4}{2} + \binom{3}{3} = 13$  [Here the summand  $\binom{6}{0}$  counts the strings where there are no 0s; the summand  $\binom{5}{1}$  counts the strings where we arrange the symbols 1, 1, 1, 1, 00; the summand  $\binom{4}{2}$  is for the arrangements of 1, 1, 00, 00; and the summand  $\binom{3}{3}$  counts the arrangements of 00, 00, 00.]
- 26.** [Here  $\binom{9}{1}$  counts the arrangements for one 111 and eight 00's;  $\binom{8}{3}$  counts the arrangements for three 111's and five 00's; and  $\binom{7}{5}$  is for the arrangements of five 111's and two 00's.]
- A :** (1)  $\lambda \in A$   
 (2) If  $a \in A$ , then  $0a0, 0a1, 1a0, 1a1 \in A$ .
- 27.** **B :** (1)  $0, 1 \in A$ .  
 (2) If  $a \in A$ , then  $0a0, 0a1, 1a0, 1a1 \in A$ .
- 28.** Of the  $3 \cdot 3 \cdot 3 \cdot 3 = 3^4 = 81$  words in  $\Sigma^4$ , there are  $3 \cdot 3 \cdot 3 \cdot 2 = 27 \cdot 2 = 54$  words that start with one of the letters  $a, b$ , or  $c$  and end with a different letter. Consequently, one must select at least  $54 + 2 = 56$  words from  $\Sigma^4$  to guarantee that at least two start and end with the same letter.

## Section 6.2



4.  $S = \{s_i \mid 0 \leq i \leq 5\}$ , where at state  $s_i$ , the machine remembers the insertion of a total of  $5i$  cents.

$$I = \{5\text{¢}, 10\text{¢}, 25\text{¢}, \text{B}, \text{W}\}$$

$O = \{n \text{ (nothing)}, P(\text{peppermint}), S(\text{spearmint}), 5\text{\textcent}, 10\text{\textcent}, 15\text{\textcent}, 20\text{\textcent}, 25\text{\textcent}\}$

$\nu$					$\omega$				
5¢	10¢	25¢	B	W	5¢	10¢	25¢	B	W
$s_0$	$s_1$	$s_2$	$s_5$	$s_0$	$s_0$	n	n	n	n
$s_1$	$s_2$	$s_3$	$s_5$	$s_1$	$s_1$	n	n	5¢	n
$s_2$	$s_3$	$s_4$	$s_5$	$s_2$	$s_2$	n	n	10¢	n
$s_3$	$s_4$	$s_5$	$s_5$	$s_3$	$s_3$	n	n	15¢	n
$s_4$	$s_5$	$s_5$	$s_5$	$s_4$	$s_4$	n	5¢	20¢	n
$s_5$	$s_5$	$s_5$	$s_5$	$s_0$	$s_0$	5¢	10¢	25¢	S P

5. (a) 010000;  $s_2$       (b)  $(s_1)100000; s_2$      $(s_2)000000; s_2$      $(s_3)110010; s_2$

(c)

$\nu$			$\omega$
	0	1	0
$s_0$	$s_0$	$s_1$	0
$s_1$	$s_1$	$s_2$	1
$s_2$	$s_2$	$s_2$	0
$s_3$	$s_0$	$s_3$	0
$s_4$	$s_2$	$s_3$	1

(d)  $\mathcal{G}_1$

(e)  $x = 101$  (unique)

6. (a) The machine recognizes (with an output of 1) every 0 (in an input string  $x$ ) that is preceded by another 0.  
 (b) State  $s_1$  remembers that at least one 0 has been supplied from an input string  $x$ .  
 (c)  $A = \{1\}^*$ ,  $B = \{00\}$

- $$7. \quad (a) \quad (i) \quad 15 \qquad \qquad \qquad (ii) \quad 3^{15} \qquad \qquad \qquad (iii) \quad 2^{15} \qquad \qquad \qquad (b) \quad 6^{15}$$

8. (a)

Input	0	1	1	0	1	1	1	0	1	1
Output	0	0	0	0	0	0	0	1	0	

(b)

	$\nu$	$\omega$
	0	1
$s_0$	$s_0$	0 0
$s_1$	$s_1$	0 0
$s_2$	$s_2$	0 0
$s_3$	$s_3$	0 0
$s_4$	$s_4$	0 0
$s_5$	$s_5$	0 1

- (c)  $\omega(x, s_0) = 0000001$  for  $x = (1)1111101; (2)1111011; (3)1110111; (4)1101111;$   
 $(5)1011111$ ; and (6)0111111

- (d) The machine recognizes the occurrence of a sixth 1, a 12th 1, ... in an input  $x$ .

9.

(a)

	$\nu$	$\omega$
	0	1
$s_0$	$s_4$	0 0
$s_1$	$s_3$	0 0
$s_2$	$s_3$	0 1
$s_3$	$s_3$	0 0
$s_4$	$s_5$	0 0
$s_5$	$s_5$	1 0

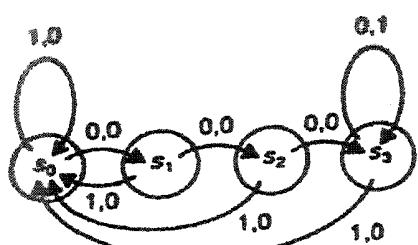
- (b) There are only two possibilities:  $x = 1111$  or  $x = 0000$ .

- (c)  $A = \{111\}\{1\}^* \cup \{000\}\{0\}^*$

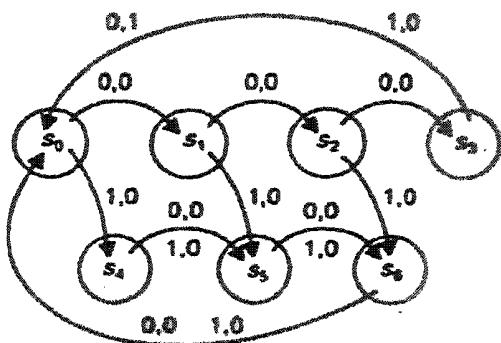
- (d) Here  $A = \{11111\}\{1\}^* \cup \{00000\}\{0\}^*$ .

### Section 6.3

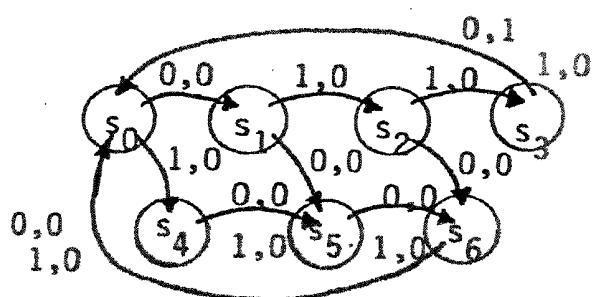
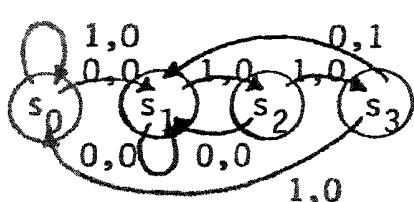
1. (a)



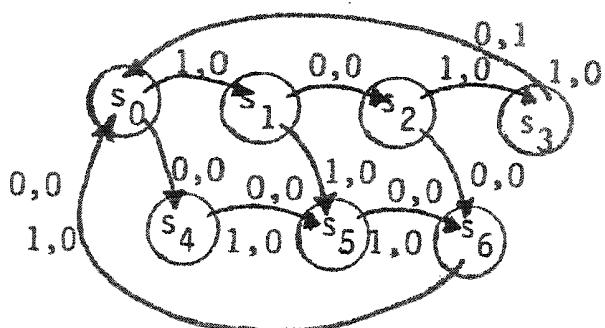
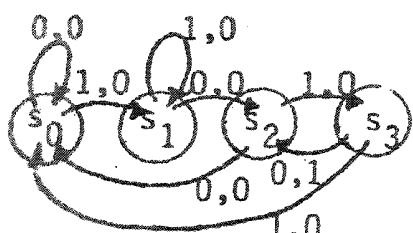
(b)



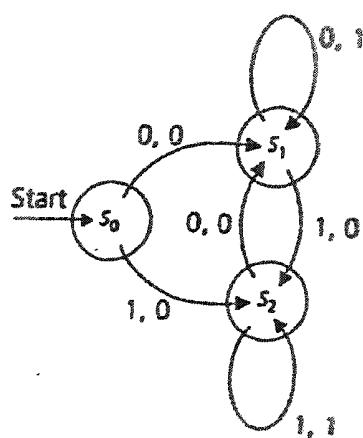
2. (0110)



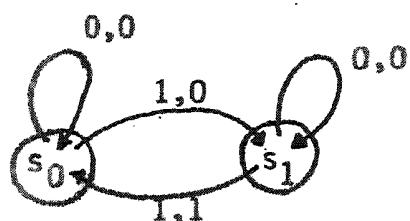
(1010)



3.



4.



5.



- (b)
- (i) Input 111  
Output 011
  - (ii) Input 1010  
Output 0101
  - (iii) Input 00011  
Output 00001

(c) The machine outputs a 0 followed by the first  $n - 1$  symbols of the  $n$  symbol input string  $x$ . Hence the machine is a unit delay.

(d) The machine here performs the same tasks as the one in Fig. 6.13 (and has only two states.)

6. Suppose the contrary and let the machine have  $n$  states, for some  $n \in \mathbb{Z}^+$ . Consider the input string  $0^{n+1}1^n$ . We expect the output here to be  $0^{n+1}1^n$ . As the 0's in this input string are processed we obtain  $n+1$  states  $s_1, s_2, \dots, s_n, s_{n+1}$  from the function  $\nu$ . Consequently, by the Pigeonhole Principle, there are two states  $s_i, s_j$  where  $i < j$  but  $s_i = s_j$ . So if the states  $s_m$ , for  $i+1 \leq m \leq j$ , are removed, along with their inputs of 0, then this machine will recognize the sequence  $0^{n+1-(j-i)}1^n$ , where  $n+1-(j-i) \leq n$ . But the string  $0^{n+1-(j-i)}1^n \notin A$ .
7. (a) The transient states are  $s_0, s_1$ . State  $s_4$  is a sink state.  $\{s_1, s_2, s_3, s_4, s_5\}, \{s_4\}, \{s_2, s_3, s_5\}$  (with the corresponding restrictions on the given function  $\nu$ ) constitute submachines. The strongly connected submachines are  $\{s_4\}$  and  $\{s_2, s_3, s_5\}$ .
- (b) States  $s_2, s_3$  are transient. The only sink state is  $s_4$ . The set  $\{s_0, s_1, s_3, s_4\}$  provides the states for a submachine;  $\{s_0, s_1\}, \{s_4\}$  provide strongly connected submachines.
- (c) Here there are no transient states. State  $s_6$  is a sink state. There are three submachines:  $\{s_2, s_3, s_4, s_5, s_6\}, \{s_3, s_4, s_5, s_6\}$ , and  $\{s_6\}$ . The only strongly connected submachine is  $\{s_6\}$ .
8. Either 110 or 111 provides a transfer sequence from  $s_2$  to  $s_5$ .

### Supplementary Exercises

1. (a) True      (b) False      (c) True  
 (d) True      (e) True      (f) True
2. No. Let  $x \in \Sigma$  with  $A = \{x, xx\}, B = \{x\}$ . Then  $A^* = B^* = \{x^n | n \geq 0\}$ , but  $A \not\subseteq B$ .
3. Let  $x \in \Sigma$  and  $A = \{x\}$ . Then  $A^2 = \{x^2\}$  and  $(A^2)^* = \{\lambda, x^2, x^4, \dots\}$ . However  $A^* = \{\lambda, x, x^2, \dots\}$  and  $(A^*)^2 = A^*$ , so  $(A^*)^2 \neq (A^2)^*$ .
4. (a)  $A^* \subset B^*$ . [For example,  $111 \in B^*$  but  $111 \notin A^*$ .]  
 (b)  $A^* = C^*$ .
5.  $O_{02}$  : Starting at  $s_0$  we can return to  $s_0$  for any input from  $\{1, 00\}^*$ . To finish at state  $s_2$  requires an input of 0. Hence  $O_{02} = \{1, 00\}^* \{0\}$   
 $O_{22} : \{0\} \{1, 00\}^* \{0\}$   
 $O_{11} : \emptyset$   
 $O_{00} : \{1, 00\}^* - \{\lambda\}$   
 $O_{10} : \{1\} \{1, 00\}^* \cup \{10\} \{1, 00\}^*$

6. (a)

	$\nu$		$\omega$
	0	1	0
$s_0$	$s_0$	$s_1$	0
$s_1$	$s_1$	$s_2$	0
$s_2$	$s_2$	$s_3$	0
$s_3$	$s_3$	$s_0$	1

(b) For any input string  $x$ , this machine recognizes (with output 1) the occurrence of every fourth 1 in  $x$ .

(c)  $\binom{8}{8} + \binom{8}{4} + \binom{8}{0} = 72$ . (The first summand is for the sequence of eight 1's, the second summand for the sequences of four 1's and four 0's, and the last summand for the sequence of eight 0's.)

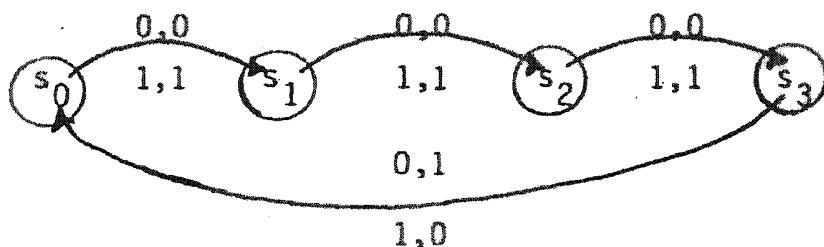
For  $\|x\| = 12$ , there are  $\binom{12}{12} + \binom{12}{8} + \binom{12}{4} + \binom{12}{0} = 992$  such sequences.

7. (a) By the Pigeonhole Principle there is a first state  $s$  that is encountered twice. Let  $y$  be the output string that resulted since  $s$  was first encountered until we reach this state a second time. Then from that point on the output is  $yyy\dots$ .

(b)  $n$  (c)  $n$

$$8. \quad x = 110$$

9.



10.

$\nu$	$\omega$	
	0	1
$s_0$	$s_1$	$s_2$
$s_1$	$s_2$	$s_1$
$s_2$	$s_2$	$s_2$
$s_3$	$s_1$	$s_0$

Here the table for  $\omega$  is obtained from Table 6.15 by reversing 0 and 1 (and, 1 and 0) for the columns under 0 and 1.

11.

	$\nu$		$\omega$	
	0	1	0	1
(a)	$(s_0, s_3)$	$(s_0, s_4)$	$(s_1, s_3)$	1 1
	$(s_0, s_4)$	$(s_0, s_3)$	$(s_1, s_4)$	0 1
	$(s_1, s_3)$	$(s_1, s_3)$	$(s_2, s_4)$	1 1
	$(s_1, s_4)$	$(s_1, s_4)$	$(s_2, s_3)$	1 1
	$(s_2, s_3)$	$(s_2, s_3)$	$(s_0, s_4)$	1 1
	$(s_2, s_4)$	$(s_2, s_4)$	$(s_0, s_3)$	1 0

(b)  $\omega((s_0, s_3), 1101) = 1111$ ;  $M_1$  is in state  $s_0$  and  $M_2$  is in state  $s_4$ .

12. The following program determines the output for the input string 1000011000.

```

10 Dim A(3,2), B(3,2)
20 Mat Read A,B
30 Data 2,1,3,1,3,1,0,0,0,0,1,1
40 Dim P(100), S(100)
50 Read N
60 For I = 1 To N
70     Read X
80     If I <> 1 Then 120
90     If X = 0 Then P(I) = B(1,1) Else P(I) = B(1,2)
100    If X = 0 Then S(I) = A(1,1) Else S(I) = A(1,2)
110    Go To 140
120    Y = X + 1
130    P(I) = B(S(I-1)Y) : S(I) = A(S(I-1),Y)
140 Next I
150 Data 10,1,0,0,0,0,1,1,0,0,0
160 Print "The output is";
170 For I = 1 To N-1
180     Print P(I);
190 Next I
200 Print P(N)
210 Print "The machine is now in state"; S(N)
220 End

```