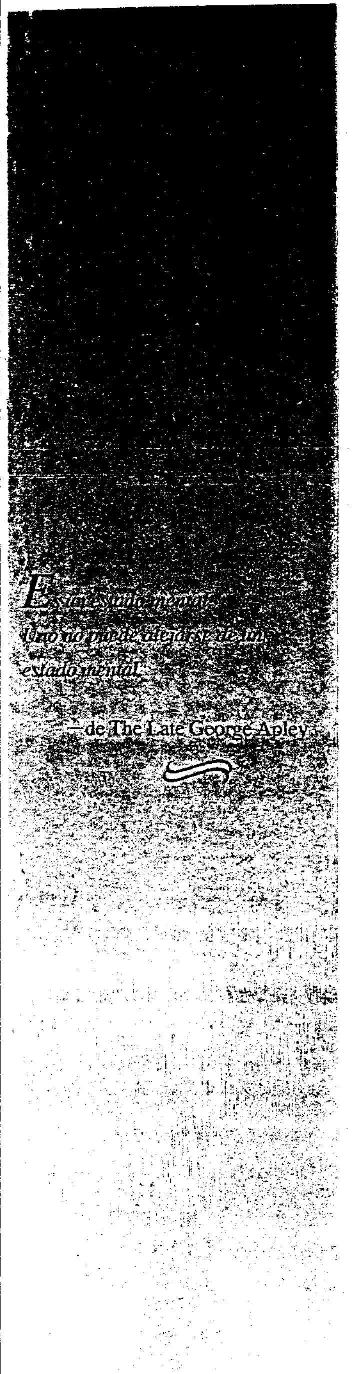


10

AUTÓMATAS, GRAMÁTICAS Y LENGUAJES

En el capítulo 9 analizamos los circuitos combinatorios donde la salida sólo dependía de la entrada. Estos circuitos no tienen memoria. En este capítulo iniciamos el estudio de los circuitos donde la salida no sólo depende de la entrada sino también del estado del sistema en el momento en que se introduce la entrada. El estado del sistema queda determinado por el procesamiento anterior. En este sentido, estos circuitos tienen memoria. Tales circuitos se llaman *circuitos secuenciales* y tienen una importancia obvia en el diseño de computadoras.

Las máquinas de estado finito son modelos abstractos de máquinas con una memoria interna primitiva. Un autómata de estado finito es un tipo particular de máquina de estado finito que está íntimamente ligada a un tipo particular de lenguaje. En la última parte de este capítulo analizaremos con cierto detalle las máquinas de estado finito, los autómatas de estado finito y los lenguajes.

10.1 *CIRCUITOS SECUENCIALES Y MÁQUINAS DE ESTADO FINITO*

Las operaciones dentro de una computadora digital se realizan a intervalos discretos de tiempo. La salida depende del estado del sistema, así como de la entrada. Supondremos que el estado del sistema cambia solamente en los instantes $t = 0, 1, \ldots$. Una forma sencilla de introducir la secuenciación en los circuitos consiste en utilizar un **retraso unitario de tiempo**.

DEFINICIÓN 10.1.1

Un *retraso unitario de tiempo* acepta como entrada un bit x_t en el instante t y tiene como salida x_{t-1}, el bit recibido como entrada en el instante $t - 1$. El retraso unitario de tiempo aparece en la figura 10.1.1.

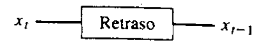


FIGURA 10.1.1
Retraso unitario de tiempo.

Como ejemplo del uso del retraso unitario de tiempo, analizaremos el **sumador en serie**.

DEFINICIÓN 10.1.2

Un *sumador en serie* acepta como entrada dos números binarios

$$x = 0x_N x_{N-1} \cdots x_0 \quad \text{y} \quad y = 0y_N y_{N-1} \cdots y_0$$

y produce como salida la suma $z_{N+1} z_N \cdots z_0$ de x y y. Los números x y y se introducen de manera secuencial por pares, $x_0, y_0; \ldots; x_N, y_N; 0, 0$. La suma sale como $z_0, z_1, \ldots, z_{N+1}$.

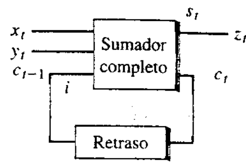


FIGURA 10.1.2
Un circuito sumador en serie.

EJEMPLO 10.1.3 *Circuito sumador en serie*

La figura 10.1.2 muestra un circuito que utiliza un retraso unitario de tiempo para implantar un sumador en serie.

Ahora mostraremos la forma en que el sumador en serie calcula la suma de

$$x = 010 \quad \text{y} \quad y = 011.$$

Primero hacemos $x_0 = 0$ y $y_0 = 1$. (Suponemos que en este instante $i = 0$. Esto se puede arreglar haciendo primero que $x = y = 0$.) El estado del sistema aparece en la figura 10.1.3a. A continuación, hacemos $x_1 = y_1 = 1$. El retraso unitario de tiempo envía $i = 0$ como el tercer bit al sumador completo. El estado del sistema aparece en la figura 10.1.3b. Por último, hacemos $x_2 = y_2 = 0$. Esta vez, el retraso unitario de tiempo envía $i = 1$ como tercer bit al sumador completo. El estado del sistema aparece en la figura 10.1.3c. Obtenemos la suma $z = 101$.

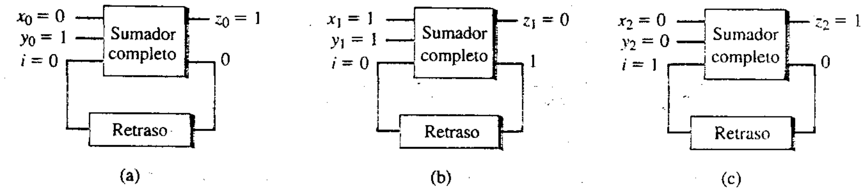


FIGURA 10.1.3 Cálculo de 010 + 011 con el circuito sumador en serie. □

Una **máquina de estado** finito es un modelo abstracto de una máquina con una memoria interna primitiva.

DEFINICION 10.1.4

Una *máquina de estado finito M* consta de

(a) Un conjunto finito I de *símbolos de entrada.*
(b) Un conjunto finito O de *símbolos de salida.*
(c) Un conjunto finito S de *estados.*
(d) Una *función de estado siguiente* f de $S \times I$ en S.
(e) Una *función de salida* g de $S \times I$ en O.
(f) Un *estado inicial* $\sigma \in S$.

Escribimos $M = (I, O, S, f, g, \sigma)$.

EJEMPLO 10.1.5

Sean $I = \{a, b\}$, $O = \{0, 1\}$ y $S = \{\sigma_0, \sigma_1\}$. Definimos el par de funciones $f : S \times I \to S$ y $g : S \times I \to O$ mediante las reglas dadas en la tabla 10.1.1.

TABLA 10.1.1

	f		g	
S \ I	a	b	a	b
σ_0	σ_0	σ_1	0	1
σ_1	σ_1	σ_1	1	0

Entonces $M = (I, O, S, f, g, \sigma_0)$ es una máquina de estado finito.

La tabla 10.1.1 se interpreta como sigue:

$$f(\sigma_0, a) = \sigma_0 \quad g(\sigma_0, a) = 0,$$
$$f(\sigma_0, b) = \sigma_1 \quad g(\sigma_0, b) = 1,$$
$$f(\sigma_1, a) = \sigma_1 \quad g(\sigma_1, a) = 1,$$
$$f(\sigma_1, b) = \sigma_1 \quad g(\sigma_1, b) = 0. \qquad \square$$

Las funciones de estado siguiente y de salida también se pueden definir mediante un **diagrama de transición.** Antes de definir este concepto, ilustraremos la forma de construirlo.

EJEMPLO 10.1.6

Dibujar el diagrama de transición para la máquina de estado finito del ejemplo 10.1.5.

El diagrama de transición es una digráfica. Los vértices son los estados (véase la figura 10.1.4). El estado inicial se indica mediante una flecha, como se muestra en la figura. Si estamos en el estado σ y la entrada i produce la salida o y nos pasa al estado σ', trazamos una arista dirigida del vértice σ al vértice σ' y la etiquetamos como i/o. Por ejemplo, si estamos en el estado σ_0, y utilizamos la entrada a, la tabla 10.1.1 nos dice que producimos la salida 0 y permanecemos en el estado σ_0. Así, trazamos un lazo dirigido sobre el vértice σ_0 y lo etiquetamos $a/0$ (véase la figura 10.1.4).

Por otro lado, si estamos en el estado σ_0 y utilizamos la entrada b, la salida es 1 y pasamos al estado σ_1. Así, trazamos una arista dirigida de σ_0 a σ_1 y la etiquetamos $b/1$. Al considerar todas las posibilidades, obtenemos el diagrama de transición de la figura 10.1.4.

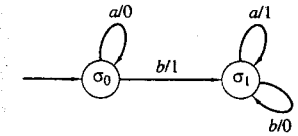


FIGURA 10.1.4 Un diagrama de transición. □

DEFINICION 10.1.7

Sea $M = (I, O, S, f, g, \sigma)$ una máquina de estado finito. El *diagrama de transición* de M es una digráfica G cuyos vértices son los miembros de S. Una flecha indica el estado inicial σ. Una arista dirigida (σ_1, σ_2) existe en G si existe una entrada i tal que $f(\sigma_1, i) = \sigma_2$. En este caso, si $g(\sigma_1, i) = o$, la arista (σ_1, σ_2) se etiqueta i/o.

Podemos considerar a la máquina de estado finito $M = (I, O, S, f, g, \sigma)$ como una computadora sencilla. Partimos del estado σ, introducimos una cadena sobre I, y producimos una cadena de salida.

DEFINICION 10.1.8

Sea $M = (I, O, S, f, g, \sigma)$ una máquina de estado finito. Una *cadena de entrada* para M es una cadena en I. La cadena

$$y_1 \cdots y_n$$

es la *cadena de salida* para M correspondiente a la cadena de entrada

$$\alpha = x_1 \cdots x_n$$

si existen estados $\sigma_0, \ldots, \sigma_n \in S$ tales que

$$\sigma_0 = \sigma$$
$$\sigma_i = f(\sigma_{i-1}, x_i) \quad \text{para } i = 1, \ldots, n;$$
$$y_i = g(\sigma_{i-1}, x_i) \quad \text{para } i = 1, \ldots, n.$$

EJEMPLO 10.1.9

Determinar la cadena de salida correspondiente a la cadena de entrada

$$aababba \tag{10.1.1}$$

para la máquina de estado finito del ejemplo 10.1.5.

Al principio, estamos en el estado σ_0. La primera entrada es a. En el diagrama de transición de M (figura 10.1.4) localizamos la arista que sale de σ_0 con la etiqueta a/x, la cual nos dice que si a es la entrada, x es la salida. En nuestro caso, 0 es la salida. La arista apunta hacia el estado siguiente, σ_0. Ahora, a es nuevamente la entrada. Como antes, producimos la salida 0 y permanecemos en el estado σ_0. Ahora, la entrada es b. En este caso, la salida es 1 y pasamos al estado σ_1. Continuamos de esta forma para determinar que la cadena de salida es

$$0011001. \tag{10.1.2}$$

□

EJEMPLO 10.1.10 *Una máquina de estado finito para el sumador en serie*

Diseñar una máquina de estado finito que realice la suma en serie.

Representaremos la máquina de estado finito mediante su diagrama de transición. Como el sumador en serie acepta pares de bits, el conjunto de entrada será

$$\{00, 01, 10, 11\}.$$

El conjunto de salida es

$$\{0, 1\}.$$

Dada una entrada xy, realizamos una de las dos acciones siguientes: sumamos x y y, o sumamos x, y y 1, según si el bit de acarreo era 0 o 1. Así, existen dos estados, que llamaremos C (acarreo) y NC (sin acarreo). El estado inicial es NC. En este momento, podemos dibujar los vértices y designar el estado inicial en nuestro diagrama de transición (véase la figura 10.1.5).

FIGURA 10.1.5
Dos estados de la máquina de estado finito para el sumador en serie.

Ahora, consideremos las entradas posibles en cada vértice. Por ejemplo, si 00 es la entrada en NC, producimos como salida 0 y permanecemos en el estado NC. Así, NC tiene un lazo etiquetado 00/0. Como otro ejemplo, si 11 es la entrada de C, calculamos $1 + 1 + 1 = 11$. En este caso, la salida es 1 y permanecemos en el estado C. Así, C tiene un lazo etiquetado 11/1. Como ejemplo final, si estamos en el estado NC y la entrada es 11, producimos como salida 0 y pasamos al estado C. Al considerar todas las posibilidades, llegamos al diagrama de transición de la figura 10.1.6.

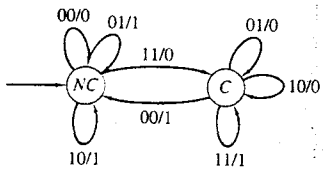


FIGURA 10.1.6 Una máquina de estado finito que realiza la suma en serie. □

EJEMPLO 10.1.11 *El flip-flop SR*

Un **flip-flop** es un componente básico de los circuitos digitales, pues sirve como una celda de memoria de un bit. El **flip-flop** SR (o **flip-flop set-reset**) se puede definir mediante la tabla siguiente

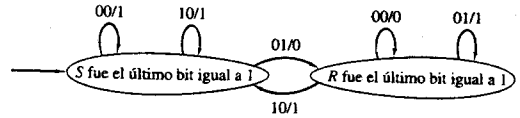


FIGURA 10.1.7 El flip-flop SR como una máquina de estado finito.

S	R	Q
1	1	No permitida
1	0	1
0	1	0
0	0	1 si S fue el último igual a 1 0 si R fue el último igual a 1

El flip-flop SR "recuerda" si S o R fue el último bit igual a 1. (Si $Q = 1$, S fue el último bit igual a 1; si $Q = 0$, R fue el último bit igual a 0.) Podemos modelar el flip-flop SR como una máquina de estado finito, definiendo dos estados: "S fue el último bit igual a 1" y "R fue el último bit igual a 1" (véase la figura 10.1.7). Definimos la entrada como los nuevos valores de S y R; la notación SR indica que $S = s$ y $R = r$. Definimos Q como la salida. De manera arbitraria hemos designado al estado inicial como "S fue el último bit igual a 1". La figura 10.1.8 muestra una implantación del flip-flop SR mediante un circuito secuencial.

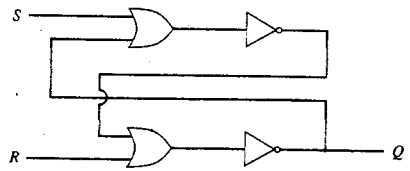


FIGURA 10.1.8 Una implantación del flip-flop SR mediante un circuito secuencial. □

Ejercicios

En los ejercicios 1-5, dibuje el diagrama de transición de la máquina de estado finito (I, O, S, f, g, σ_0).

1. $I = \{a, b\}$, $O = \{0, 1\}$, $S = \{\sigma_0, \sigma_1\}$

	f		g	
S \ I	a	b	a	b
σ_0	σ_1	σ_1	1	1
σ_1	σ_0	σ_1	0	1

2. $I = \{a, b\}$, $O = \{0, 1\}$, $S = \{\sigma_0, \sigma_1\}$

	f		g	
S \ I	a	b	a	b
σ_0	σ_1	σ_0	0	0
σ_1	σ_0	σ_0	1	1

3. $I = \{a, b\}$, $O = \{0, 1\}$, $S = \{\sigma_0, \sigma_1, \sigma_2\}$

	f		g	
S \ I	a	b	a	b
σ_0	σ_1	σ_1	0	1
σ_1	σ_2	σ_1	1	1
σ_2	σ_0	σ_0	0	0

4. $I = \{a, b, c\}$, $O = \{0, 1\}$, $S = \{\sigma_0, \sigma_1, \sigma_2\}$

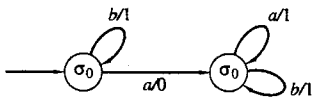
	f			g		
S \ I	a	b	c	a	b	c
σ_0	σ_0	σ_1	σ_2	0	1	0
σ_1	σ_1	σ_1	σ_0	1	1	1
σ_2	σ_2	σ_1	σ_0	1	0	0

5. $I = \{a, b, c\}$, $O = \{0, 1, 2\}$, $S = \{\sigma_0, \sigma_1, \sigma_2, \sigma_3\}$

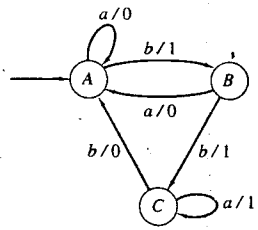
	f			g		
S \ I	a	b	c	a	b	c
σ_0	σ_1	σ_0	σ_2	1	1	2
σ_1	σ_0	σ_2	σ_2	2	0	0
σ_2	σ_3	σ_3	σ_0	1	0	1
σ_3	σ_1	σ_1	σ_0	2	0	2

En los ejercicios 6-10, determine los conjuntos I, O y S, el estado inicial y la tabla que define las funciones de estado siguiente y de salida para cada máquina de estado finito.

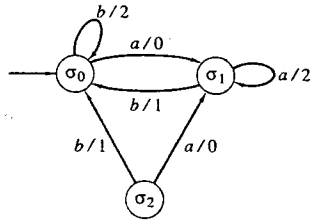
6.



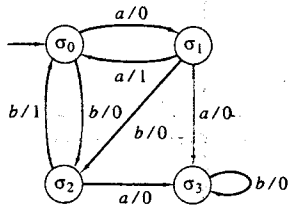
7.



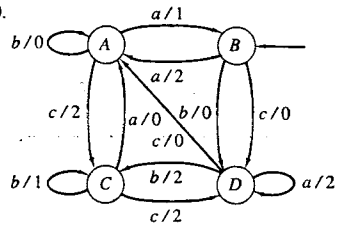
8.



9.



10.



En los ejercicios 11-20, determine la cadena de salida para la cadena de entrada y la máquina de estado finito dadas.

11. *abba*; ejercicio 1
12. *abba*; ejercicio 2
13. *aabbaba*; ejercicio 3
14. *aabbcc*; ejercicio 4
15. *aabaab*; ejercicio 5
16. *aaa*; ejercicio 6
17. *aabbabaab*; ejercicio 7
18. *baaba*; ejercicio 8
19. *bbababbabaaa*; ejercicio 9
20. *cacbccbaabac*; ejercicio 10

En los ejercicios 21-26, diseñe una máquina de estado finito con las propiedades dadas. La entrada siempre es una cadena de bits.

21. Produce la salida 1 si la entrada es un número par de bits; produce la salida 0 en caso contrario.
22. Produce la salida 1 si la entrada son k unos, donde k es un múltiplo de 3; produce la salida 0 en caso contrario.
23. Produce la salida 1 si la entrada son dos o más unos; produce la salida 0 en caso contrario.
24. Produce la salida 1 siempre que vea 101; produce la salida 0 en caso contrario.
25. Produce la salida 1 a partir de que vea 101; produce la salida 0 en caso contrario.
26. Produce la salida 1 cuando ve el primer 0 y hasta ver otro 0; a partir de ese momento, produce la salida 0; en los demás casos, produce la salida 0.
27. Sea $\alpha = x_1 \cdots x_n$ una cadena de bits. Sea $\beta = y_1 \cdots y_n$, donde

$$y_i = \begin{cases} a & \text{si } x_i = 0 \\ b & \text{si } x_i = 1 \end{cases}$$

para $i = 1, \ldots, n$. Sea $\gamma = y_n \cdots y_1$.

Muestre que si γ es la entrada de la máquina de estado finito de la figura 10.1.4, la salida es el complemento a dos de α (véase el algoritmo 9.5.16 para una descripción del complemento a dos).

Entrada	*Salida*
11	1
00	0
10	1
01	1
00	0
00	1
00	0
00	0

☆ 28. Muestre que no existe una máquina de estado finito que reciba una cadena de bits y produzca como salida 1 siempre que el número de unos de entrada sea igual al número de ceros de entrada y produzca 0 en caso contrario.

☆ 29. Muestre que no existe una máquina de estado finito que realice la multiplicación en serie. En específico, muestre que no existe una máquina de estado finito que reciba como entrada los números binarios $X = x_1 \cdots x_n$ y $Y = y_1 \cdots y_n$ como la secuencia de números de dos bits

$$x_n y_n, \quad x_{n-1} y_{n-1} \quad \ldots, \quad x_1 y_1, \quad 00, \quad \ldots, \quad 00,$$

donde existen n 00, y las salidas $z_{2n}, \ldots z_1$, donde $Z = z_1 \cdots z_{2n} = XY$. *Ejemplo*: Si tal máquina existiese, para multiplicar 101 × 1001 habría que introducir como entrada 11, 00, 10, 01, 00, 00, 00, 00. El primer par 11 es el par de los bits del extremo derecho (10<u>1</u>, 100<u>1</u>); el segundo par 00 es el siguiente par de bits (1<u>0</u>1, 10<u>0</u>1); y así sucesivamente. Rellenamos la cadena de entrada con cuatro pares de 00, la longitud del factor de mayor longitud, 1001. Como 101 × 1001 = 101101, se está afirmando que se obtiene la salida de la tabla anexa.

10.2 AUTÓMATAS DE ESTADO FINITO

Un **autómata de estado finito** es un tipo particular de máquina de estado finito. Los autómatas de estado finito tienen un interés especial, debido a su relación con los lenguajes, como veremos en la sección 10.5.

DEFINICIÓN 10.2.1

Un *autómata de estado finito* $A = (I, O, S, f, g, \sigma)$ es una máquina de estado finito en la que el conjunto de símbolos de salida es $\{0, 1\}$ y donde el estado actual determina la última salida. Aquellos estados para los cuales la última salida es 1 son los *estados de aceptación*.

EJEMPLO 10.2.2

Dibujar el diagrama de transición de la máquina de estado finito A definida mediante la siguiente tabla. El estado inicial es σ_0. Mostrar que A es un autómata de estado finito y determinar el conjunto de estados de aceptación.

	f		g	
I / S	a	b	a	b
σ_0	σ_1	σ_0	1	0
σ_1	σ_2	σ_0	1	0
σ_2	σ_2	σ_0	1	0

El diagrama de transición aparece en la figura 10.2.1. Si estamos en el estado σ_0, la última salida fue 0. Si estamos en el estado σ_1 o en el estado σ_2, la última salida fue 1; así, A es un autómata de estado finito. Los estados de aceptación son σ_1 y σ_2.

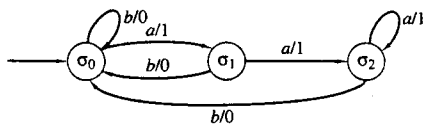


FIGURA 10.2.1 El diagrama de transición para el ejemplo 10.2.2. □

El ejemplo 10.2.2 muestra que la máquina de estado finito definida mediante un diagrama de transición será un autómata de estado finito si el conjunto de símbolos de salida es $\{0, 1\}$ y si, para cada estado σ, todas las aristas que llegan a σ tienen la misma etiqueta de salida.

Por lo general, el diagrama de transición de un autómata de estado finito se dibuja con los estados de aceptación encerrados en círculos dobles y omitiendo los símbolos de salida. Al dibujar el diagrama de transición de la figura 10.2.1 de esta forma, obtenemos el diagrama de transición de la figura 10.2.2.

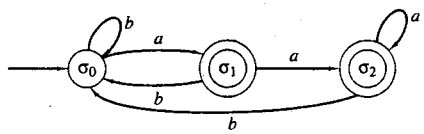


FIGURA 10.2.2 El diagrama de transición de la figura 10.2.1, con los estados de aceptación encerrados en círculos dobles y omitiendo los símbolos de salida.

EJEMPLO 10.2.3

Dibujar el diagrama de transición del autómata de estado finito de la figura 10.2.3 como el diagrama de transición de una máquina de estado finito.

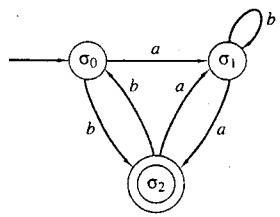


FIGURA 10.2.3 Un autómata de estado finito.

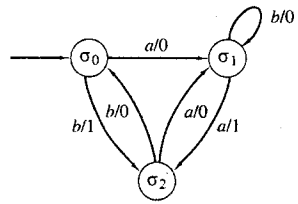


FIGURA 10.2.4 El autómata de estado finito de la figura 10.2.3, ahora como un diagrama de transición de una máquina de estado finito.

Como σ_2 es un estado de aceptación, etiquetamos todas sus aristas de entrada con la salida 1 (véase la figura 10.2.4). Los estados σ_0 y σ_1 no son de aceptación, así que etiquetamos todas sus aristas de entrada con la salida 0. Obtenemos el diagrama de transición de la figura 10.2.4. □

Como una alternativa a la definición 10.2.1, podemos considerar a un autómata de estado finito A como formado por

1. Un conjunto finito I de *símbolos de entrada.*
2. Un conjunto finito S de *estados.*
3. Una *función de estado siguiente* f de $S \times I$ en S.
4. Un subconjunto A de S de *estados de aceptación.*
5. Un estado inicial $\sigma \in S$.

Si utilizamos esta caracterización, escribimos $A = (I, S, f, A, \sigma)$.

EJEMPLO 10.2.4

El diagrama de transición del autómata de estado finito $A = (I, S, f, A, \sigma)$, donde

$$I = \{a, b\}; \quad S = \{\sigma_0, \sigma_1, \sigma_2\}; \quad A = \{\sigma_2\}; \quad \sigma = \sigma_0;$$

y f está dada por la siguiente tabla.

	f	
S \ I	a	b
σ_0	σ_0	σ_1
σ_1	σ_0	σ_2
σ_2	σ_0	σ_2

aparece en la figura 10.2.5.

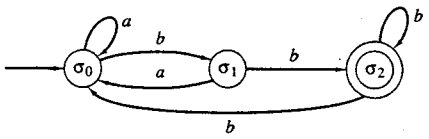


FIGURA 10.2.5 El diagrama de transición para el ejemplo 10.2.4. □

Si una cadena se utiliza como entrada de un autómata de estado finito, terminaremos en un estado de aceptación o en uno de no aceptación. La situación de este estado final determina si la cadena es **aceptada** por el autómata de estado finito.

DEFINICIÓN 10.2.5

Sea $A = (I, S, f, A, \sigma)$ un autómata de estado finito. Sea $\alpha = x_1 \cdots x_n$ una cadena sobre I. Si existen estados $\sigma_0, \ldots, \sigma_n$ tales que

(a) $\sigma_0 = \sigma$;

(b) $f(\sigma_{i-1}, x_i) = \sigma_i$ para $i = 1, \ldots, n$;

(c) $\sigma_n \in A$;

decimos que α es *aceptada por* A. La cadena nula es aceptada si y sólo si $\sigma \in A$. Sea $\mathrm{Ac}(A)$ el conjunto de cadenas aceptadas por A; decimos que A *acepta* $\mathrm{Ac}(A)$.

Sea $\alpha = x_1 \cdots x_n$ una cadena sobre I. Definimos los estados $\sigma_0, \ldots, \sigma_n$ mediante las condiciones (a) y (b) anteriores. Decimos que el camino (dirigido) $(\sigma_0, \ldots, \sigma_n)$ es el camino que *representa* a α en A.

La definición 10.2.5 implica que si el camino P representa a la cadena α en un autómata de estado finito A, entonces A acepta a α si y sólo si P termina en un estado de aceptación.

EJEMPLO 10.2.6

¿Es aceptada la cadena *abaa* por el autómata de estado finito de la figura 10.2.2?

Comenzamos en el estado σ_0. Cuando a es la entrada, pasamos al estado σ_1. Cuando b es la entrada, vamos al estado σ_0. Cuando a es la entrada, pasamos al estado σ_1. Por último, cuando se utiliza como entrada el último símbolo a, pasamos al estado σ_2. El camino $(\sigma_0, \sigma_1, \sigma_0, \sigma_1, \sigma_2)$ representa la cadena *abaa.* Como el estado final σ_2 es un estado de aceptación, la cadena *abaa* es aceptada por el autómata de estado finito de la figura 10.2.2. □

EJEMPLO 10.2.7

¿Es aceptada la cadena $\alpha = abbabba$ por el autómata de estado finito de la figura 10.2.3?

El camino que representa a α termina en σ_1. Como σ_1 no es un estado de aceptación, la cadena α no es aceptada por el autómata de estado finito de la figura 10.2.3. □

Ahora daremos dos ejemplos que ilustran los problemas del diseño.

EJEMPLO 10.2.8

Diseñar un autómata de estado finito que acepte precisamente aquellas cadenas sobre $\{a, b\}$ que no tengan letras a.

La idea es utilizar dos estados:

A: Se encontró una a.

NA: No se encontró una a.

El estado NA es el estado inicial y el único estado de aceptación. Ahora es sencillo dibujar las aristas (véase la figura 10.2.6). Observe que el autómata de estado finito acepta en forma correcta la cadena vacía. □

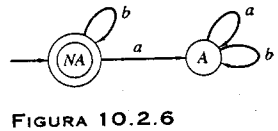


FIGURA 10.2.6
Un autómata de estado finito que acepta precisamente aquellas cadenas sobre $\{a, b\}$ que no tienen letras a.

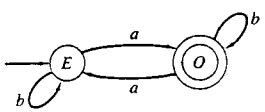


FIGURA 10.2.7
Un autómata de estado finito que acepta precisamente aquellas cadenas sobre $\{a, b\}$ que contienen un número impar de letras a.

EJEMPLO 10.2.9

Diseñar un autómata de estado finito que acepte precisamente aquellas cadenas sobre $\{a, b\}$ que contienen un número impar de letras a.

Esta vez los dos estados son

E: Se encontró un número par de a.

O: Se encontró un número impar de a.

El estado inicial es E y el estado de aceptación es O. Obtenemos el diagrama de transición de la figura 10.2.7. □

Un autómata de estado finito es en esencia un algoritmo que sirve para decidir si una cadena dada es aceptada o no. Como ejemplo, convertiremos el diagrama de transición de la figura 10.2.7 en un algoritmo.

ALGORITMO 10.2.10

Este algoritmo determina si una cadena sobre $\{a, b\}$ es aceptada por el autómata de estado finito cuyo diagrama de transición aparece en la figura 10.2.7.

Entrada: n, la longitud de la cadena ($n = 0$ designa a la cadena vacía)
$s_1 s_2 \cdots s_n$, la cadena

Salida: "Aceptar" si la cadena es aceptada
"Rechazar" si la cadena no es aceptada

```
procedure fsa (s, n)
  state := 'E'
  for i := 1 to n do
    begin
    if state = 'E' and s_i = 'a' then
      state := 'O'
    if state = 'O' and s_i = 'a' then
      state := 'E'
    end
  if state = 'O' then
    return("Aceptar")
  else
    return("Rechazar")
end fsa
```

Si dos autómatas de estado finito aceptan precisamente las mismas cadenas, decimos que los autómatas son **equivalentes**.

DEFINICIÓN 10.2.11

Los autómatas de estado finito A y A' son *equivalentes* si $\mathrm{Ac}(A) = \mathrm{Ac}(A')$.

EJEMPLO 10.2.12

Se puede verificar que los autómatas de estado finito de las figuras 10.2.6 y 10.2.8 son equivalentes (véase el ejercicio 33). □

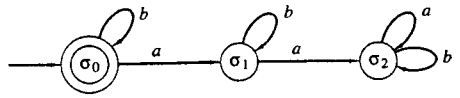


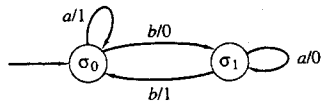
FIGURA 10.2.8 Un autómata de estado finito equivalente al de la figura 10.2.6.

Si definimos una relación R sobre un conjunto de autómatas de estado finito mediante la regla ARA' si A y A' son equivalentes (en el sentido de la definición 10.2.11), R es una relación de equivalencia. Cada clase de equivalencia consta de un conjunto de autómatas de estado finito equivalentes entre sí.

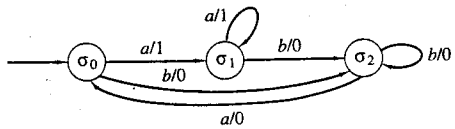
Ejercicios

En los ejercicios 1-3, muestre que cada máquina de estado finito es un autómata de estado finito y trace de nuevo el diagrama de transición como el diagrama de un autómata de estado finito.

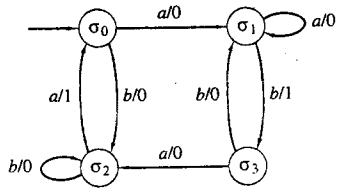
1.



2.

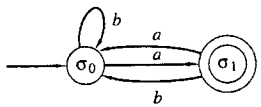


3.

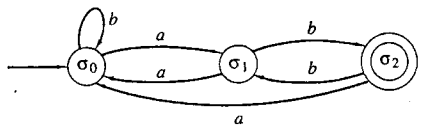


En los ejercicios 4-6, trace de nuevo el diagrama de transición del autómata de estado finito como el diagrama de transición de una máquina de estado finito.

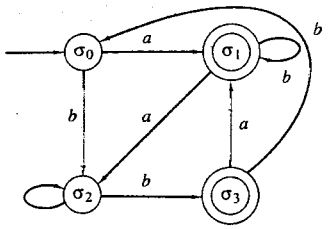
4.



5.



6.



En los ejercicios 7-9, dibuje el diagrama de transición del autómata de estado finito (I, S, f, A, σ_0).

7. $I = \{a, b\}, S = \{\sigma_0, \sigma_1, \sigma_2\}, A = \{\sigma_0\}$

	f	
S \ I	a	b
σ_0	σ_1	σ_0
σ_1	σ_2	σ_0
σ_2	σ_0	σ_2

8. $I = \{a, b\}, S = \{\sigma_0, \sigma_1, \sigma_2\}, A = \{\sigma_0, \sigma_2\}$

	f	
S \ I	a	b
σ_0	σ_1	σ_1
σ_1	σ_0	σ_2
σ_2	σ_0	σ_1

9. $I = \{a, b, c\}, S = \{\sigma_0, \sigma_1, \sigma_2, \sigma_3\}, A = \{\sigma_0, \sigma_2\}$

	f		
S \ I	a	b	c
σ_0	σ_1	σ_0	σ_2
σ_1	σ_0	σ_3	σ_0
σ_2	σ_3	σ_2	σ_0
σ_3	σ_1	σ_0	σ_1

10. Para cada uno de los autómatas de estado finito de los ejercicios 1-6, determine los conjuntos I, S y A, el estado inicial y la tabla que define la función de estado siguiente.

11. ¿Cuáles son las máquinas de estado finito de los ejercicios 1-10, sección 10.1, son autómatas de estado finito?

12. ¿Cómo debe verse la tabla de una máquina de estado finito M para que M sea un autómata de estado finito?

En los ejercicios 13-17, determine si la cadena dada es aceptada por el autómata de estado finito dado.

13. *abbaa*; figura 10.2.2
14. *abbaa*; figura 10.2.3
15. *aabaabb*; figura 10.2.5
16. *aaabbbaab*; ejercicio 5
17. *aaababbab*; ejercicio 6

18. Muestre que una cadena α sobre $\{a, b\}$ es aceptada por el autómata de estado finito de la figura 10.2.2 si y sólo si α termina en a.

19. Muestre que una cadena α sobre $\{a, b\}$ es aceptada por el autómata de estado finito de la figura 10.2.5 si y sólo si α termina en bb.

☆ 20. Caracterice las cadenas aceptadas por el autómata de estado finito de los ejercicios 1-9.

En los ejercicios 21-31, trace el diagrama de transición de un autómata de estado finito que acepte el conjunto dado de cadenas en $\{a, b\}$.

21. Número par de a
22. Exactamente una b
23. Al menos una b
24. Exactamente dos a
25. Al menos dos a
26. Contiene m letras a, donde m es un múltiplo de 3
27. Comienza con *baa*
☆ 28. Contiene *abba*
29. Toda b es seguida por a
★ **30.** Termina con *aba*
☆ 31. Inicia con *ab* y termina con *baa*

32. Escriba algoritmos, similares al algoritmo 10.2.10, que deciden si una cadena dada es aceptada o no por el autómata de estado finito de los ejercicios 1-9.

33. Proporcione un argumento formal para mostrar que los autómatas de estado finito de las figuras 10.2.6 y 10.2.8 son equivalentes.

34. Sea L un conjunto finito de cadenas sobre $\{a, b\}$. Muestre que existe un autómata de estado finito que acepta a L.

35. Sea L el conjunto de cadenas aceptadas por el autómata de estado finito del ejercicio 6. Sea S el conjunto de todas las cadenas sobre $\{a, b\}$. Diseñe un autómata de estado finito que acepte a $S - L$.

36. Sea L_i el conjunto de cadenas aceptadas por los autómatas de estado finito $A_i = (I, S_i, f_i, A_i, \sigma_i)$, $i = 1, 2$. Sea

$$A = (I, S_1 \times S_2, f, A, \sigma)$$

donde

$$f((S_1, S_2), x) = (f_1(S_1, x), f_2(S_2, x)),$$
$$A = \{(A_1, A_2) \mid A_1 \in A_1 \text{ y } A_2 \in A_2\},$$
$$\sigma = (\sigma_1, \sigma_2).$$

Muestre que $\text{Ac}(A) = L_1 \cap L_2$.

37. Sea L_i el conjunto de cadenas aceptadas por los autómatas de estado finito $A_i = (I, S_i, f_i, A_i, \sigma_i)$, $i = 1, 2$. Sea

$$A = (I, S_1 \times S_2, f, A, \sigma)$$

donde

$$f((S_1, S_2), x) = (f_1(S_1, x), f_2(S_2, x)),$$
$$A = \{(A_1, A_2) \mid A_1 \in A_1 \text{ o } A_2 \in A_2\},$$
$$\sigma = (\sigma_1, \sigma_2).$$

Muestre que $\text{Ac}(A) = L_1 \cup L_2$.

En los ejercicios 38-42, sea $L_i = \text{Ac}(A_i)$, $i = 1, 2$. Trace el diagrama de transición de los autómatas de estado finito que aceptan $L_1 \cap L_2$ y $L_1 \cup L_2$.

38. A_1 dada en el ejercicio 4; A_2 está dada en el ejercicio 5

39. A_1 dada en el ejercicio 4; A_2 está dada en el ejercicio 6

40. A_1 dada en el ejercicio 5; A_2 está dada en el ejercicio 6

41. A_1 dada en el ejercicio 6; A_2 está dada en el ejercicio 6

42. A_1 dada por la figura 10.5.7, sección 10.5; A_2 dada en el ejercicio 6

10.3 LENGUAJES Y GRAMÁTICAS

De acuerdo con el *Diccionario Webster*, un lenguaje es un "cuerpo de palabras y métodos de combinación de palabras utilizado y comprendido por una comunidad de tamaño considerable". Tales lenguajes se llaman **lenguajes naturales** para distinguirlos de los **lenguajes formales**, los cuales se utilizan para modelar los lenguajes naturales y para comunicarse con las computadoras. Las reglas de un lenguaje natural son muy complejas y es difícil caracterizarlas por completo. Por otro lado, es posible especificar completamente las reglas de construcción de ciertos lenguajes formales. Comenzaremos con la definición de un lenguaje formal.

DEFINICIÓN 10.3.1

Sea A un conjunto finito. Un *lenguaje* (*formal*) L sobre A es un subconjunto de A^*, el conjunto de todas las cadenas sobre A.

EJEMPLO 10.3.2

Sea $A = \{a, b\}$. El conjunto L de todas las cadenas sobre A que contiene un número impar de letras a es un lenguaje sobre A. Como vimos en el ejemplo 10.2.9, L es precisamente el conjunto de cadenas sobre A aceptadas por el autómata de estado finito de la figura 10.2.7. □

Una forma de definir un lenguaje es dar una lista de reglas que éste debe cumplir.

DEFINICIÓN 10.3.3

Una *gramática con estructura de frases* (o simplemente una *gramática*) G consta de

(a) Un conjunto finito N de *símbolos no terminales*.

(b) Un conjunto finito T de *símbolos terminales*, donde $N \cap T = \varnothing$.

(c) Un subconjunto finito P de $[(N \cup T)^* - T^*] \times (N \cup T)^*$, llamado el conjunto de *producciones*.

(d) Un *símbolo inicial* $\sigma \in N$.

Escribimos $G = (N, T, P, \sigma)$.

Por lo general, una producción $(A, B) \in P$ se escribe

$$A \to B.$$

La definición 10.3.3c establece que en la producción $A \to B$, $A \in (N \cup T)^* - T^*$ y $B \in (N \cup T)^*$; así, A debe incluir al menos un símbolo no terminal, mientras que B puede constar de cualquier combinación de símbolos terminales y no terminales.

EJEMPLO 10.3.4

Sean

$$N = \{\sigma, S\}$$
$$T = \{a, b\}$$
$$P = \{\sigma \to b\sigma, \sigma \to aS, S \to bS, S \to b\}.$$

Entonces $G = (N, T, P, \sigma)$ es una gramática. □

Dada una gramática G, podemos construir un lenguaje $L(G)$ a partir de G utilizando las producciones para deducir las cadenas que forman a $L(G)$. La idea es partir del símbolo inicial y luego utilizar varias veces las producciones hasta obtener una cadena de símbolos terminales. El lenguaje $L(G)$ es el conjunto de todas las cadenas obtenidas de esta forma. La definición 10.3.5 da los detalles formales.

DEFINICIÓN 10.3.5

Sea $G = (N, T, P, \sigma)$ una gramática.

Si $\alpha \to \beta$ es una producción y $x\,\alpha\,y \in (N \cup T)^*$, decimos que $x\,\beta\,y$ se *deriva directamente* de $x\alpha y$ y escribimos

$$x\alpha y \Rightarrow x\beta y.$$

Si $\alpha_i \in (N \cup T)^*$ para $i = 1, \ldots, n$ y α_{i+1} se deriva directamente de α_i para $i = 1, \ldots, n-1$, decimos que α_n se *deriva de* α_1 y escribimos

$$\alpha_1 \Rightarrow \alpha_n.$$

Decimos que

$$\alpha_1 \Rightarrow \alpha_2 \Rightarrow \cdots \Rightarrow \alpha_n$$

es la *derivación de* α_n (*a partir de* α_1). Por convención, cualquier elemento de $(N \cup T)^*$ se deriva de sí mismo.

El *lenguaje generado por* G, denotado $L(G)$, consta de todas las cadenas sobre T derivables de σ.

EJEMPLO 10.3.6

Sea G la gramática del ejemplo 10.3.4.

La cadena $abSbb$ se deriva directamente de $aSbb$, lo cual se escribe

$$aSbb \Rightarrow abSbb$$

utilizando la producción $S \to bS$.

La cadena $bbab$ se deriva de σ, lo cual se escribe

$$\sigma \Rightarrow bbab.$$

La derivación es

$$\sigma \Rightarrow b\sigma \Rightarrow bb\sigma \Rightarrow bbaS \Rightarrow bbab.$$

Las únicas derivaciones de σ son

$$\begin{aligned}
&\sigma \Rightarrow b\sigma \\
&\quad\vdots \\
&\Rightarrow b^n\sigma \qquad n \geq 0 \\
&\Rightarrow b^n aS \\
&\quad\vdots \\
&\Rightarrow b^n ab^{m-1}S \\
&\Rightarrow b^n ab^m \quad n \geq 0, \quad m \geq 1.
\end{aligned}$$

Así, $L(G)$ consta de las cadenas sobre $\{a, b\}$ que contienen precisamente una a y terminan con b. □

Una forma alternativa para establecer las producciones de una gramática es utilizar la **forma normal de Backus** (o **forma Backus-Naur**, o **FBN**). En la FBN, los símbolos no terminales comienzan por lo general con "<" y terminan con ">". La producción $S \to T$ se escribe $S ::= T$. Las producciones de la forma

$$S ::= T_1, \quad S ::= T_2, \quad \ldots, \quad S ::= T_n$$

se pueden combinar como

$$S ::= T_1 \mid T_2 \mid \cdots \mid T_n.$$

La barra "|" se lee "o".

EJEMPLO 10.3.7 *Una gramática para los enteros*

Un entero se define como una cadena que consta de un signo opcional (+ o −) seguido de una cadena de dígitos (0 a 9). La siguiente gramática genera a todos los enteros.

$$\begin{aligned}
&< \text{dígito} > ::= 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9 \\
&< \text{entero} > ::= < \text{entero con signo} > \mid < \text{entero sin signo} > \\
&< \text{entero con signo} > ::= + < \text{entero sin signo} > \mid - < \text{entero sin signo} > \\
&< \text{entero sin signo} > ::= < \text{dígito} > \mid < \text{dígito} > < \text{entero sin signo} >
\end{aligned}$$

El símbolo inicial es < entero >.

Por ejemplo, la derivación del entero −901 es

$$\begin{aligned}
< \text{entero} > &\Rightarrow < \text{entero con signo} > \\
&\Rightarrow - < \text{entero sin signo} > \\
&\Rightarrow - < \text{dígito} > < \text{entero sin signo} > \\
&\Rightarrow - < \text{dígito} > < \text{dígito} > < \text{entero sin signo} > \\
&\Rightarrow - < \text{dígito} > < \text{dígito} > < \text{dígito} > \\
&\Rightarrow -9 < \text{dígito} > < \text{dígito} > \\
&\Rightarrow -90 < \text{dígito} > \\
&\Rightarrow -901.
\end{aligned}$$

En la notación de la definición 10.3.3, este lenguaje consta de

1. El conjunto $N = \{< \text{dígito} >, < \text{entero} >, < \text{entero con signo} >, < \text{entero sin signo} >\}$ de símbolos no terminales.
2. El conjunto $T = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, +, -\}$ de símbolos terminales.
3. Las producciones

$$\begin{aligned}
< \text{dígito} > &\to 0, \ldots, < \text{dígito} > \to 9, \\
< \text{entero} > &\to < \text{entero con signo} >, \\
< \text{entero} > &\to < \text{entero sin signo} >, \\
< \text{entero con signo} > &\to + < \text{entero sin signo} >, \\
< \text{entero con signo} > &\to - < \text{entero sin signo} >, \\
< \text{entero sin signo} > &\to < \text{dígito} >, \\
< \text{entero sin signo} > &\to < \text{dígito} > < \text{entero sin signo} >.
\end{aligned}$$

4. El símbolo inicial < entero >. □

Los lenguajes de computación, como FORTRAN, Pascal y C, se especifican por lo general en FBN. El ejemplo 10.3.7 muestra la forma de especificar una constante entera en un lenguaje de computación mediante su FBN.

Las gramáticas se clasifican según los tipos de producciones que las definen.

DEFINICIÓN 10.3.8

Sea G una gramática y λ la cadena nula.

(a) Si cada producción es de la forma

$$\alpha A\beta \to \alpha\delta\beta, \quad \text{donde } \alpha, \beta \in (N \cup T)^*, \quad A \in N, \quad \delta \in (N \cup T)^* - \{\lambda\}, \tag{10.3.1}$$

entonces G es una *gramática sensible al contexto* (o de *tipo* 1).

(b) Si cada producción es de la forma

$$A \to \delta, \quad \text{donde } A \in N, \quad \delta \in (N \cup T)^*, \tag{10.3.2}$$

entonces G es una *gramática libre de contexto* (o de *tipo* 2).

(c) Si cada producción es de la forma

$$A \to a \text{ o } A \to aB \text{ o } A \to \lambda, \quad \text{donde } A, B \in N, \quad a \in T,$$

decimos que G es una *gramática regular* (o de *tipo* 3).

Según (10.3.1), en una gramática sensible al contexto, podemos reemplazar A por δ si A está en el contexto de α y β. En una gramática libre de contexto, (10.3.2) establece que podemos reemplazar A por δ en cualquier momento. Una gramática regular tiene reglas de sustitución particularmente sencillas: Reemplazamos un símbolo no terminal por un símbolo terminal, por un símbolo terminal seguido de un símbolo no terminal, o por la cadena nula.

Observe que una gramática regular es una gramática libre de contexto y que una gramática libre de contexto sin producciones de la forma $A \to \lambda$ es una gramática sensible al contexto.

Algunas definiciones permiten reemplazar a por una cadena de terminales en la definición 10.3.8c; sin embargo, se puede mostrar (véase el ejercicio 32) que las dos definiciones producen los mismos lenguajes.

EJEMPLO 10.3.9

La gramática G definida por

$$T = \{a, b, c\}, \quad N = \{\sigma, A, B, C, D, E\},$$

con producciones

$$\sigma \to aAB, \quad \sigma \to aB, \quad A \to aAC, \quad A \to aC, \quad B \to Dc,$$
$$D \to b, \quad CD \to CE, \quad CE \to DE, \quad DE \to DC, \quad Cc \to Dcc,$$

y símbolo inicial σ es sensible al contexto. Por ejemplo, la producción $CE \to DE$ dice que podemos reemplazar C con D si C va seguida de E y la producción $Cc \to Dcc$ dice que podemos reemplazar C con Dc si C va seguida de c.

Podemos derivar DC de CD, pues

$$CD \Rightarrow CE \Rightarrow DE \Rightarrow DC.$$

La cadena $a^3b^3c^3$ está en $L(G)$, pues tenemos

$$\sigma \Rightarrow aAB \Rightarrow aaACB \Rightarrow aaaCCDc \Rightarrow aaaDCCc \Rightarrow aaaDCDcc$$
$$\Rightarrow aaaDDCcc \Rightarrow aaaDDDccc \Rightarrow aaabbbccc.$$

Se puede mostrar (véase el ejercicio 33) que

$$L(G) = \{a^n b^n c^n \mid n = 1, 2, \ldots\}. \qquad \square$$

Es natural permitir que el lenguaje $L(G)$ herede una propiedad de una gramática G. La siguiente definición precisa este concepto.

DEFINICIÓN 10.3.10

Un lenguaje L es *sensible al contexto* (respectivamente, *libre de contexto*, *regular*) si existe una gramática G sensible al contexto (respectivamente, libre de contexto, regular) tal que $L = L(G)$.

EJEMPLO 10.3.11

De acuerdo con el ejemplo 10.3.9, el lenguaje

$$L = \{a^n b^n c^n \mid n = 1, 2, \ldots\}$$

es sensible al contexto. Se puede mostrar (véase [Hopcroft, página 127]) que no existe una gramática libre de contexto G tal que $L = L(G)$; por tanto, L no es un lenguaje libre de contexto. $\square$

EJEMPLO 10.3.12

La gramática G definida por

$$T = \{a, b\}, \quad N = \{\sigma\},$$

con producciones

$$\sigma \to a\sigma b, \quad \sigma \to ab,$$

y símbolo inicial σ es libre de contexto. Las únicas derivaciones de σ son

$$\sigma \Rightarrow a\sigma b$$
$$\vdots$$
$$\Rightarrow a^{n-1}\sigma b^{n-1}$$
$$\Rightarrow a^{n-1}abb^{n-1} = a^n b^n.$$

Así, $L(G)$ consta de las cadenas sobre $\{a, b\}$ de la forma $a^n b^n$; $n = 1, 2, \ldots$. Este lenguaje es libre de contexto. En la sección 10.5 (véase el ejemplo 10.5.6) mostraremos que $L(G)$ no es regular. $\square$

Los ejemplos 10.3.11 y 10.3.12 muestran que el conjunto de lenguajes libres de contexto que no contienen a la cadena nula es un subconjunto propio del conjunto de lenguajes sensibles al contexto y que el conjunto de lenguajes regulares es un subconjunto propio del conjunto de lenguajes libres de contexto. También se puede mostrar que existen lenguajes que no son sensibles al contexto.

EJEMPLO 10.3.13

La gramática G definida en el ejemplo 10.3.4 es regular. Así, el lenguaje

$$L(G) = \{b^n a b^m \mid n = 0, 1, \ldots; m = 1, 2, \ldots\}$$

generado por ella es regular. $\square$

EJEMPLO 10.3.14

La gramática del ejemplo 10.3.7 es libre de contexto, pero no regular. Sin embargo, si cambiamos las producciones por

$$\begin{aligned}
&\langle \text{entero} \rangle ::= + \langle \text{entero sin signo} \rangle \mid - \langle \text{entero sin signo} \rangle \mid \\
&\qquad 0\langle \text{dígitos} \rangle \mid 1\langle \text{dígitos} \rangle \mid \cdots \mid 9\langle \text{dígitos} \rangle \\
&\langle \text{entero sin signo} \rangle ::= 0\langle \text{dígitos} \rangle \mid 1\langle \text{dígitos} \rangle \mid \cdots \mid 9\langle \text{dígitos} \rangle \\
&\langle \text{dígitos} \rangle ::= 0\langle \text{dígitos} \rangle \mid 1\langle \text{dígitos} \rangle \mid \cdots \mid 9\langle \text{dígitos} \rangle \mid \lambda,
\end{aligned}$$

la gramática resultante es regular. Como el lenguaje generado no se modifica, esto implica que el conjunto de cadenas que representan a los enteros es un lenguaje regular. $\square$

El ejemplo 10.3.14 motiva la siguiente definición.

DEFINICIÓN 10.3.15

Las gramáticas G y G' son *equivalentes* si $L(G) = L(G')$.

EJEMPLO 10.3.16

Las gramáticas de los ejemplos 10.3.7 y 10.3.14 son equivalentes. □

Si definimos una relación R sobre un conjunto de gramáticas mediante la regla $G\,R\,G'$ si G y G' son equivalentes (en el sentido de la definición 10.3.15), R es una relación de equivalencia. Cada clase de equivalencia consta de un conjunto de gramáticas equivalentes entre sí.

Concluimos esta sección presentando de manera breve otro tipo de gramática que se puede utilizar para generar curvas fractales.

DEFINICIÓN 10.3.17

Una *gramática Lindenmayer interactiva libre de contexto* consta de

(a) Un conjunto finito N de *símbolos no terminales*.

(b) Un conjunto finito T de *símbolos terminales*, donde $N \cap T = \varnothing$.

(c) Un conjunto finito P de *producciones* $A \to B$, donde $A \in N \cup T$ y $B \in (N \cup T)^*$.

(d) Un *símbolo inicial* $\sigma \in N$.

La diferencia entre una gramática Lindenmayer interactiva libre de contexto y una gramática libre de contexto es que la primera permite el uso de producciones de la forma $A \to B$, donde A es un símbolo terminal o no terminal. (En una gramática libre de contexto, A debe ser no terminal.)

Las reglas para derivar cadenas en una gramática Lindenmayer interactiva libre de contexto son diferentes de las reglas para derivar cadenas en una gramática con estructura de frases (véase la definición 10.3.5). En una gramática Lindenmayer interactiva libre de contexto, para derivar la cadena β de la cadena α, hay que reemplazar *todos* los símbolos en α *de manera simultánea*. A continuación damos la definición formal.

DEFINICIÓN 10.3.18

Sea $G = (N, T, P, \sigma)$ una gramática Lindenmayer interactiva libre de contexto. Si

$$\alpha = x_1 \cdots x_n$$

y existen producciones

$$x_i \to \beta_i$$

en P, para $i = 1, \ldots, n$, escribimos

$$\alpha \Rightarrow \beta_1 \cdots \beta_n$$

y decimos que $\beta_1 \cdots \beta_n$ es *derivable de manera directa* de α. Si α_{i+1} es derivable de manera directa de α_i para $i = 1, \ldots, n-1$, decimos que α_n es *derivable de* α_1 y escribimos

$$\alpha_1 \Rightarrow \alpha_n.$$

Decimos que

$$\alpha_1 \Rightarrow \alpha_2 \Rightarrow \cdots \Rightarrow \alpha_n$$

es la *derivación de* α_n (*a partir de* α_1). El *lenguaje generado por* G, denotado $L(G)$, consta de todas las cadenas sobre T derivables a partir de σ.

EJEMPLO 10.3.19 ***El copo de nieve de von Koch***

Sean

$$N = \{D\}$$
$$T = \{d, +, -\}$$
$$P = \{D \to D - D + + D - D, D \to d, + \to +, - \to -\}.$$

Consideramos a $G(N, T, P, D)$ como una gramática Lindenmayer interactiva libre de contexto. Como un ejemplo de derivación de D, tenemos

$$D \Rightarrow D - D + + D - D \Rightarrow d - d + + d - d.$$

Así, $d - d + + d - d \in L(G)$.

Ahora daremos un significado a las cadenas de $L(G)$. Interpretamos el símbolo d como una instrucción para trazar una línea recta de una longitud fija en la dirección actual; interpretamos $+$ como una instrucción para girar 60° hacia la derecha; e interpretamos $-$ como una instrucción para girar 60° hacia la izquierda. Si comenzamos del lado izquierdo y el primer movimiento es horizontal y hacia la derecha, la interpretación de la cadena $d - d + + d - d$ produce la curva de la figura 10.3.1a.

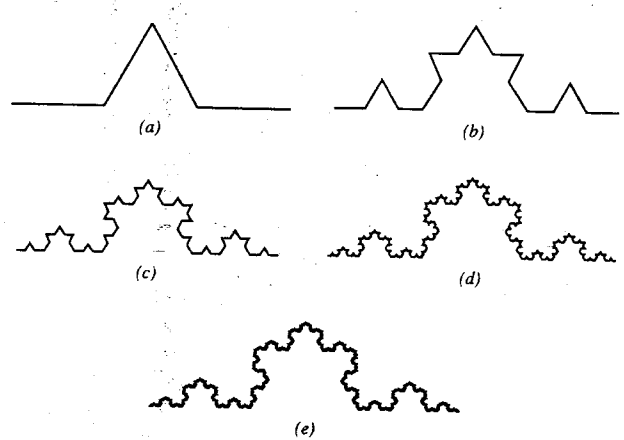


FIGURA 10.3.1 Copos de nieve de von Koch.

La siguiente cadena más larga en $L(G)$ es

$$d - d + + d - d - d - d + + d - d + + d - d + + d - d - d - d + + d - d$$

cuya derivación es

$$\begin{aligned} D &\Rightarrow D - D + + D - D \\ &\Rightarrow D - D + + D - D - D - D + + D - D + + D \\ &\quad - D + + D - D - D - D + + D - D \\ &\Rightarrow d - d + + d - d - d - d + + d - d + + d \\ &\quad - d + + d - d - d - d + + d - d. \end{aligned}$$

No es posible obtener una cadena más corta debido a que al usar las producciones debemos reemplazar *todos* los símbolos de manera simultánea (definición 10.3.18). Si reemplazamos algunas D por d y otras D por $D - D + + D - D$, no podemos derivar alguna cadena de la cadena resultante, no sólo una cadena terminal, pues d no aparece en el lado izquierdo de ninguna producción.

Al interpretar la cadena

$$d - d + + d - d - d - d + + d - d + + d - d + + d - d - d - d + + d - d$$

obtenemos la curva de la figura 10.3.1b.

Las curvas obtenidas al interpretar las cadenas en $L(G)$ aparecen en la figura 10.3.1c-e. Estas curvas se conocen como los **copos de nieve de von Koch**. □

Las curvas como el copo de nieve de von Koch se llaman **curvas fractales** (véase [Peitgen]). Una característica de las curvas fractales es que una parte del todo se parece al todo. Por ejemplo, como muestra la figura 10.3.2, cuando la parte indicada del copo de nieve de von Koch se extrae y se amplía, recuerda a la curva original.

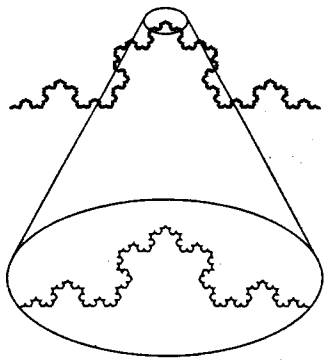


FIGURA 10.3.2 La naturaleza fractal del copo de nieve de von Koch. Cuando la parte superior del copo de nieve de von Koch se extrae y se amplía, se asemeja a la curva original.

Las gramáticas Lindenmayer interactivas libres de contexto y sensibles al contexto fueron ideadas en 1968 por A. Lindenmayer (véase [Lindenmayer]) para modelar el crecimiento de las plantas. Como lo sugiere el ejemplo 10.3.19, estas gramáticas se pueden utilizar en la graficación por computadora para generar imágenes (véase [Prusinkiewicz 1986, 1988; Smith]). Se puede mostrar (véase [Wood, página 503]) que la clase de lenguajes generados por las gramáticas Lindenmayer interactiva sensibles al contexto es exactamente igual a la clase de lenguajes generados por las gramáticas con estructura de frases.

Ejercicios

En los ejercicios 1-6, determine si la gramática dada es sensible al contexto, libre de contexto, regular, o ninguna de éstas. Indique todas las caracterizaciones que se le apliquen.

1. $T = \{a, b\}, N = \{\sigma, A\}$, con producciones

$$\sigma \to b\sigma, \quad \sigma \to aA, \quad A \to a\sigma,$$
$$A \to bA, \quad A \to a, \quad \sigma \to b,$$

y símbolo inicial σ.

2. $T = \{a, b, c\}, N = \{\sigma, A, B\}$, con producciones

$$\sigma \to AB, \quad AB \to BA, \quad A \to aA,$$
$$B \to Bb, \quad A \to a, \quad B \to b,$$

y símbolo inicial σ.

3. $T = \{a, b\}, N = \{\sigma, A, B\}$, con producciones

$$\sigma \to A, \quad \sigma \to AAB, \quad Aa \to ABa, \quad A \to aa,$$
$$Bb \to ABb, \quad AB \to ABB, \quad B \to b,$$

y símbolo inicial σ.

4. $T = \{a, b, c\}, N = \{\sigma, A, B\}$, con producciones

$$\sigma \to BAB, \quad \sigma \to ABA, \quad A \to AB, \quad B \to BA,$$
$$A \to aA, \quad A \to ab, \quad B \to b,$$

y símbolo inicial σ.

5. $<S> ::= b<S> \mid a<A> \mid a$
$<A> ::= a<S> \mid b<B>$
$<B> ::= b<A> \mid a<S> \mid b$

con símbolo inicial $<S>$.

6. $T = \{a, b\}, N = \{\sigma, A, B\}$, con producciones

$$\sigma \to AA\sigma, \quad AA \to B, \quad B \to bB, \quad A \to a,$$

y símbolo inicial σ.

En los ejercicios 7-11, muestre que la cadena dada α está en $L(G)$ para la gramática G dada, mostrando una derivación de α.

7. *bbabbab*, ejercicio 1
8. *abab*, ejercicio 2
9. *aabbaab*, ejercicio 3
10. *abbbaabab*, ejercicio 4
11. *abaabbabba*, ejercicio 5

12. Escriba las gramáticas de los ejemplos 10.3.4 y 10.3.9 y los ejercicios 1-4 y 6 en FBN.

☆ 13. Sea G la gramática del ejercicio 1. Muestre que $\alpha \in L(G)$ si y sólo si α es no nula y contiene un número par de letras a.

☆ 14. Sea G la gramática del ejercicio 5. Caracterice $L(G)$.

En los ejercicios 15-24, escriba una gramática que genere las cadenas con la propiedad dada.

15. Cadenas sobre $\{a, b\}$ que comienzan con a
16. Cadenas sobre $\{a, b\}$ que terminan con ba
17. Cadenas sobre $\{a, b\}$ que contienen ba
★ 18. Cadenas sobre $\{a, b\}$ que no terminan con ab
19. Enteros sin ceros al principio
20. Números con punto flotante (números como .294, 89., 67.284)
21. Números exponenciales (números que incluyen a los números con punto flotante y números como 6.9E3, 8E12, 9.6E-4, 9E-10)
22. Expresiones booleanas en $X_1, \ldots, X_n$
23. Todas las cadenas sobre $\{a, b\}$
24. Cadenas $x_1 \cdots x_n$ sobre $\{a, b\}$ tales que $x_1 \cdots x_n = x_n \cdots x_1$

Se propone que cada una de las gramáticas de los ejercicios 25-31 genera el conjunto L de cadenas sobre $\{a, b\}$ que contiene la misma cantidad de letras a y letras b. Si la gramática genera L, demuestre que lo hace. Si la gramática no genera L, dé un contraejemplo y demuestre que su contraejemplo es correcto. En cada gramática, S es el símbolo inicial.

25. $S \to aSb \mid bSa \mid \lambda$
26. $S \to aSb \mid bSa \mid SS \mid \lambda$
27. $S \to aB \mid bA \mid \lambda, B \to b \mid bA, A \to a \mid aB$
28. $S \to abS \mid baS \mid aSb \mid bSa \mid \lambda$
29. $S \to aSb \mid bSa \mid abS \mid baS \mid Sab \mid Sba \mid \lambda$
30. $S \to aB \mid bA, A \to a \mid SA, B \to b \mid SB$
31. $S \to aSbS \mid bSaS \mid \lambda$

★ 32. Sea G una gramática y λ la cadena nula. Muestre que si cada producción es de la forma

$$A \to \alpha \text{ o } A \to \alpha B \text{ o } A \to \lambda, \quad \text{donde } A, B \in N, \quad a \in T^* - \{\lambda\},$$

entonces existe una gramática regular G' con $L(G) = L(G')$.

☆ 33. Sea G la gramática del ejemplo 10.3.9. Muestre que

$$L(G) = \{a^n b^n c^n \mid n = 1, 2, \ldots\}.$$

34. Muestre que el lenguaje

$$\{a^n b^n c^k \mid n, k \in \{1, 2, \ldots\}\}$$

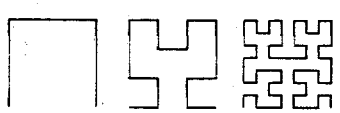
es un lenguaje libre de contexto.

35. Sean

$$N = \{S, D\}$$
$$T = \{d, +, -\}$$
$$P = \{S \to D + D + D + D,$$
$$D \to D + D - D - DD + D + D - D \mid d, + \to +, - \to -\}.$$

Considere a $G = (N, T, P, S)$ como una gramática Lindenmayer libre de contexto. Interprete el símbolo d como una instrucción para trazar una línea recta, con una longitud fija, en la dirección actual; interprete + como una instrucción para girar a la derecha 90°; e interprete − como una instrucción para girar a la izquierda 90°. Genere las dos cadenas de menor longitud en $L(G)$ y trace las curvas correspondientes. Estas curvas se conocen como las *islas cuadráticas de Koch*.

☆ 36. La siguiente figura



muestra las tres primeras etapas de la *curva de Hilbert*. Defina una gramática Lindenmayer libre de contexto que genere cadenas tales que, al ser interpretadas en forma adecuada, generen la curva de Hilbert.

10.4 AUTÓMATAS DE ESTADO FINITO NO DETERMINISTAS

En esta y la siguiente sección mostraremos que las gramáticas regulares y los autómatas de estado finito son esencialmente lo mismo, en el sentido de que cada una especifica un lenguaje regular. Comenzamos con un ejemplo que ilustra la forma de convertir un autómata de estado finito en una gramática regular.

EJEMPLO 10.4.1

Escribir la gramática regular dada por el autómata de estado finito de la figura 10.2.7.

Los símbolos terminales son los símbolos de entrada $\{a, b\}$. Los estados E y O se convierten en los símbolos no terminales. El estado inicial E se convierte en el símbolo inicial. Las producciones corresponden a las aristas dirigidas. Si existe una arista etiquetada x de S a S', escribimos la producción

$$S \to x S'.$$

En nuestro caso, obtenemos las producciones

$$E \to bE, \quad E \to aO, \quad O \to aE, \quad O \to bO. \tag{10.4.1}$$

Además, si S es un estado de aceptación, incluimos la producción

$$S \to \lambda.$$

En nuestro caso, obtenemos la producción adicional

$$O \to \lambda. \tag{10.4.2}$$

Entonces la gramática $G = (N, T, P, E)$, con $N = \{O, E\}$, $T = \{a, b\}$ y P formado por las producciones (10.4.1) y (10.4.2), genera el lenguaje $L(G)$, que es igual al conjunto de cadenas aceptadas por el autómata de estado finito de la figura 10.2.7. □

TEOREMA 10.4.2

Sea A un autómata de estado finito dado como un diagrama de transición. Sea σ el estado inicial. Sea T el conjunto de símbolos de entrada y sea N el conjunto de estados. Definimos las producciones

$$S \to x S'$$

si existe una arista etiquetada x de S a S', y

$$S \to \lambda$$

si S es un estado de aceptación. Sea G la gramática regular

$$G = (N, T, P, \sigma).$$

Entonces el conjunto de cadenas aceptadas por A es igual a $L(G)$.

Demostración. Primero mostraremos que $\text{Ac}(A) \subseteq L(G)$. Sea $\alpha \in \text{Ac}(A)$. Si α es la cadena nula, entonces σ es un estado de aceptación. En este caso, G contiene la producción

$$\sigma \to \lambda.$$

La derivación

$$\sigma \Rightarrow \lambda \tag{10.4.3}$$

muestra que $\alpha \in L(G)$.

Ahora, supongamos que $\alpha \in \text{Ac}(A)$ y que α no es la cadena nula. Entonces $\alpha = x_1 \cdots x_n$ para $x_i \in T$. Como α es aceptada por A, existe un camino $(\sigma, S_1, \ldots, S_n)$, donde S_n es un estado de aceptación, con las aristas etiquetadas en forma sucesiva como $x_1, \ldots, x_n$. Esto implica que G contiene las producciones

$$\sigma \to x_1 S_1$$

$$S_{i-1} \to x_i S_i \quad \text{para } i = 2, \ldots, n.$$

Como S_n es un estado de aceptación, G también contiene a la producción

$$S_n \to \lambda.$$

La derivación

$$\begin{aligned} \sigma &\Rightarrow x_1 S_1 \\ &\Rightarrow x_1 x_2 S_2 \\ &\vdots \\ &\Rightarrow x_1 \cdots x_n S_n \\ &\Rightarrow x_1 \cdots x_n \end{aligned} \tag{10.4.4}$$

muestra que $\alpha \in L(G)$.

Concluiremos la demostración mostrando que $L(G) \subseteq \text{Ac}(A)$. Supongamos que $\alpha \in L(G)$. Si α es la cadena nula, α debe surgir de la derivación (10.4.3), pues una derivación que comienza con cualquier otra producción daría como resultado una cadena no nula. Así, la producción $\sigma \to \lambda$ está en la gramática. Por tanto, σ es un estado de aceptación en A. Esto implica que $\alpha \in \text{Ac}(A)$.

Ahora, supongamos que $\alpha \in L(G)$ y que α no es la cadena nula. Entonces $\alpha = x_1 \cdots x_n$ para $x_i \in T$. Esto implica que existe una derivación de la forma (10.4.4). Si, en el diagrama de transición, comenzamos con σ y seguimos el camino $(\sigma, S_1, \ldots, S_n)$, podemos generar la cadena α. La última producción utilizada en (10.4.4) es $S_n \to \lambda$; así, el último estado alcanzado es un estado de aceptación. Por tanto, α es aceptada por A, de modo que $L(G) \subseteq \text{Ac}(A)$. Con esto concluimos la demostración. ■

A continuación, consideremos la situación inversa. Dada una gramática regular G, querémos construir un autómata de estado finito A de modo que $L(G)$ sea precisamente el conjunto de cadenas aceptadas por A. A primera vista, podría parecer que basta invertir el procedimiento del teorema 10.4.2. Sin embargo, el siguiente ejemplo muestra que la situación es un poco más compleja.

EJEMPLO 10.4.3

Consideremos la gramática regular definida por

$$T = \{a, b\}, \quad N = \{\sigma, C\}$$

con producciones

$$\sigma \to b\sigma, \quad \sigma \to aC, \quad C \to bC, \quad C \to b$$

y símbolo inicial σ.

Los símbolos no terminales se convierten en estados, donde σ es el estado inicial. Para cada producción de la forma

$$S \to xS'$$

trazamos una arista del estado S al estado S' y la etiquetamos x. Las producciones

$$\sigma \to b\sigma, \quad \sigma \to aC, \quad C \to bC$$

proporcionan la gráfica de la figura 10.4.1. La producción $C \to b$ es equivalente a las dos producciones

$$C \to bF, \quad F \to \lambda,$$

donde F es un símbolo no terminal adicional. Las producciones

$$\sigma \to b\sigma, \quad \sigma \to aC, \quad C \to bC, \quad C \to bF$$

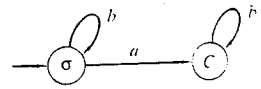


FIGURA 10.4.1
La gráfica correspondiente a las producciones $\sigma \to b\sigma$, $\sigma \to aC$, $C \to bC$.

proporcionan la gráfica de la figura 10.4.2. La producción

$$F \to \lambda$$

nos dice que F debe ser un estado de aceptación (véase la figura 10.4.2). □

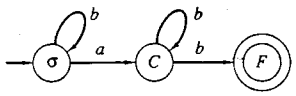


FIGURA 10.4.2 El autómata de estado finito no determinista correspondiente a la gramática $\sigma \to b\sigma$, $\sigma \to aC$, $C \to bC$, $C \to b$.

Por desgracia, la gráfica de la figura 10.4.2 no es un autómata de estado finito. Existen varios problemas. El vértice C no tiene una arista de salida con la etiqueta a, y el vértice F no tiene aristas de salida. Además, el vértice C tiene dos aristas de salida con la etiqueta b. Un diagrama como el de la figura 10.4.2 define otro tipo de autómata llamado un **autómata de estado finito no determinista**. La razón de las palabras "no determinista" es que cuando estamos en un estado donde hay varias aristas de salida, todas ellas con la misma etiqueta x, si se introduce x como entrada, tenemos una situación no determinista, pues hay que elegir el estado siguiente. Por ejemplo, si en la figura 10.4.2 estamos en el estado C y la entrada es b, podemos elegir entre dos estados siguientes: permanecer en el estado C o ir al estado F.

DEFINICIÓN 10.4.4

Un *autómata de estado finito no determinista* A consta de

(a) Un conjunto finito I de *símbolos de entrada.*

(b) Un conjunto finito S de *estados.*

(c) Una *función de estado siguiente* f de $S \times I$ en $P(S)$.

(d) Un subconjunto A de S de *estados de aceptación.*

(e) Un *estado inicial* $\sigma \in S$.

Escribimos $A = (I, S, f, A, \sigma)$.

La única diferencia entre un autómata de estado finito no determinista y un autómata de estado finito es que en este último la función de estado siguiente nos lleva a un estado definido de manera única, mientras que en un autómata de estado finito no determinista la función de estado siguiente nos lleva a un conjunto de estados.

EJEMPLO 10.4.5

Para el autómata de estado finito no determinista de la figura 10.4.2, tenemos

$$I = \{a, b\}, \quad S = \{\sigma, C, F\}, \quad A = \{F\}.$$

El estado inicial es σ y la función de estado siguiente f está dada por

	f	
S \ I	a	b
σ	$\{C\}$	$\{\sigma\}$
C	$\varnothing$	$\{C, F\}$
F	$\varnothing$	$\varnothing$

□

El diagrama de transición de un autómata de estado finito no determinista se traza de una manera análoga al de un autómata de estado finito. Trazamos una arista del estado S a cada estado en el conjunto $f(S, x)$ y los etiquetamos x.

EJEMPLO 10.4.6

El diagrama de transición del autómata de estado finito no determinista

$$I = \{a, b\}, \quad S = \{\sigma, C, D\}, \quad A = \{C, D\}$$

con estado inicial σ y función de estado siguiente

	f	
S \ I	a	b
σ	$\{\sigma, C\}$	$\{D\}$
C	$\varnothing$	$\{C\}$
D	$\{C, D\}$	$\varnothing$

aparece en la figura 10.4.3.

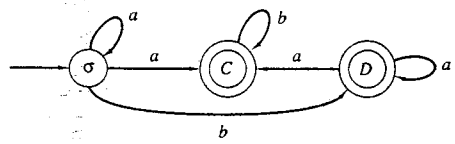


FIGURA 10.4.3 El diagrama de transición del autómata de estado finito no determinista del ejemplo 10.4.6. □

Una cadena α es aceptada por un autómata de estado finito no determinista A si existe algún camino que represente a α en el diagrama de transición de A que parta del estado inicial y llegue a un estado de aceptación. A continuación damos la definición formal.

DEFINICIÓN 10.4.7

Sea $A = (I, S, f, A, \sigma)$ un autómata de estado finito no determinista. La cadena nula es *aceptada* por A si y sólo si $\sigma \in A$. Si $\alpha = x_1 \cdots x_n$ es una cadena no nula sobre I y existen estados $\sigma_0, \ldots, \sigma_n$ tales que

(a) $\sigma_0 = \sigma$;

(b) $\sigma_i \in f(\sigma_{i-1}, x_i)$ para $i = 1, \ldots, n$;

(c) $\sigma_n \in A$;

decimos que α es *aceptada* por A. $\mathrm{Ac}(A)$ denota el conjunto de cadenas aceptadas por A y decimos que A *acepta* $\mathrm{Ac}(A)$.

Si A y A' son autómatas de estado finito no deterministas y $\mathrm{Ac}(A) = \mathrm{Ac}(A')$, decimos que A y A' son *equivalentes*.

Si $\alpha = x_1 \cdots x_n$ es una cadena sobre I y existen estados $\sigma_0, \ldots, \sigma_n$ que satisfagan las condiciones (a) y (b), el camino $(\sigma_0, \ldots, \sigma_n)$ es un *camino que representa* a α en A.

EJEMPLO 10.4.8

La cadena

$$\alpha = bbabb$$

es aceptada por el autómata de estado finito no determinista de la figura 10.4.2, pues el camino $(\sigma, \sigma, \sigma, C, C, F)$, el cual termina en un estado de aceptación, representa a α. Observe que el camino $P = (\sigma, \sigma, \sigma, C, C, C)$ también representa a α, pero P no termina en un estado de aceptación. Sin embargo, la cadena α es aceptada pues existe al menos un camino que representa a α y que termina en un estado de aceptación. Una cadena β no es aceptada si ningún camino representa a β o bien si todos los caminos que representan a β terminan en un estado que no es de aceptación. □

EJEMPLO 10.4.9

La cadena $\alpha = aabaabbb$ es aceptada por el autómata de estado finito no determinista de la figura 10.4.3. El lector debe localizar el camino que representa a α, y que termina en el estado C. □

EJEMPLO 10.4.10

La cadena $\alpha = abba$ no es aceptada por el autómata de estado finito no determinista de la figura 10.4.3. Al salir de σ, cuando la entrada es a, existen dos opciones: Ir a C o permanecer en σ. Si vamos a C, al introducir dos letras b, nuestros movimientos quedan determinados y permanecemos en C. Pero cuando se introduce la a final, no existe una arista a lo largo de la cual moverse. Por otro lado, supongamos que cuando introducimos la primera a, permanecemos en σ. Entonces, al introducir b, nos movemos a D. Pero ahora, al introducir la siguiente b, no existe una arista sobre la cual moverse. Como no existe un camino que represente a α en la figura 10.4.3, la cadena α no es aceptada por el autómata de estado finito no determinista de la figura 10.4.3. □

Formulemos la construcción del ejemplo 10.4.3 como un teorema.

TEOREMA 10.4.11

Sea $G = (N, T, P, \sigma)$ una gramática regular. Sean

$$I = T$$
$$S = N \cup \{F\}, \text{ donde } F \notin N \cup T$$
$$f(S, x) = \{S' \mid S \to xS' \in P\} \cup \{F \mid S \to x \in P\}$$
$$A = \{F\} \cup \{S \mid S \to \lambda \in P\}.$$

Entonces el autómata de estado finito no determinista $A = (I, S, f, A, \sigma)$ acepta precisamente las cadenas $L(G)$.

Demostración. La demostración es esencialmente igual a la demostración del teorema 10.4.2 y por tanto se omite. ■

Podría parecer que un autómata de estado finito no determinista es un concepto más general que el de un autómata de estado finito; sin embargo, en la siguiente sección mostraremos que dado un autómata de estado finito no determinista A, podemos construir un autómata de estado finito que es equivalente a A.

Ejercicios

En los ejercicios 1-5, trace el diagrama de transición del autómata de estado finito no determinista (I, S, f, A, σ_0).

1. $I = \{a, b\}$, $S = \{\sigma_0, \sigma_1, \sigma_2\}$, $A = \{\sigma_0\}$

S \ I	a	b
σ_0	$\varnothing$	$\{\sigma_1, \sigma_2\}$
σ_1	$\{\sigma_2\}$	$\{\sigma_0, \sigma_1\}$
σ_2	$\{\sigma_0\}$	$\varnothing$

2. $I = \{a, b\}$, $S = \{\sigma_0, \sigma_1, \sigma_2\}$, $A = \{\sigma_0, \sigma_1\}$

S \ I	a	b
σ_0	$\{\sigma_1\}$	$\{\sigma_0, \sigma_2\}$
σ_1	$\varnothing$	$\{\sigma_2\}$
σ_2	$\{\sigma_1\}$	$\varnothing$

EJEMPLO 10.5.7

Sea L el conjunto de cadenas aceptada por el autómata de estado finito A de la figura 10.5.4. Construir un autómata de estado finito que acepte las cadenas

$$L^R = \{x_n \cdots x_1 \mid x_1 \cdots x_n \in L\}.$$

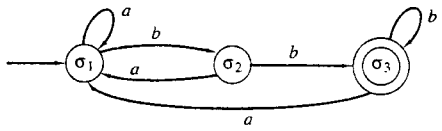


FIGURA 10.5.4 El autómata de estado finito del ejemplo 10.5.7, que acepta a L.

Queremos convertir A en un autómata de estado finito que acepte L^R. La cadena $\alpha = x_1 \cdots x_n$ es aceptada por A si existe un camino P en A que represente a α, comience en σ_1 y termina en σ_3. Si comenzamos en σ_3 y recorremos P en reversa, terminamos en σ_1 y procesamos las aristas en el orden $x_n, \ldots, x_1$. Así, sólo necesitamos invertir todas las flechas de la figura 10.5.4 y hacer de σ_3 el estado inicial y σ_1 el estado de aceptación (véase la figura 10.5.5). El resultado es un autómata de estado finito no determinista que acepta L^R.

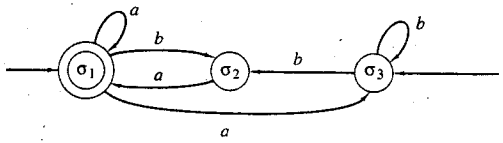


FIGURA 10.5.5 Un autómata de estado finito no determinista que acepta L^R.

Después de determinar un autómata de estado finito equivalente y eliminar los estados inalcanzables, obtenemos el autómata de estado finito equivalente de la figura 10.5.6.

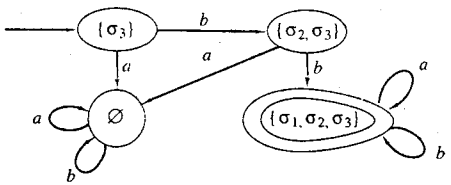


FIGURA 10.5.6 Un autómata de estado finito que acepta L^R. □

EJEMPLO 10.5.8

Sea L el conjunto de cadenas aceptadas por el autómata de estado finito A de la figura 10.5.7. Construir un autómata de estado finito no determinista que acepte las cadenas

$$L^R = \{x_n \cdots x_1 \mid x_1 \cdots x_n \in L\}.$$

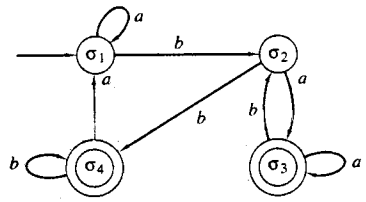


FIGURA 10.5.7 El autómata de estado finito del ejemplo 10.5.8, que acepta a L.

Si A sólo tuviera un estado de aceptación, podríamos utilizar el procedimiento del ejemplo 10.5.7 para construir el autómata de estado finito no determinista deseado. Así, primero construimos un autómata de estado finito no determinista equivalente a A con un estado de aceptación. Para hacer esto, introducimos un estado adicional σ_5. Luego hacemos que los caminos que terminan en σ_3 o σ_4 terminen de manera opcional en σ_5 (véase la figura 10.5.8). El autómata de estado finito no determinista deseado se obtiene a partir de la figura 10.5.8 por el método del ejemplo 10.5.7 (véase la figura 10.5.9). Por supuesto, si se desea, podríamos construir un autómata de estado finito equivalente. □

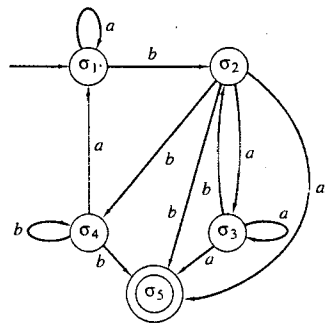


FIGURA 10.5.8 Un autómata de estado finito no determinista con un estado de aceptación equivalente al autómata de estado finito de la figura 10.5.7.

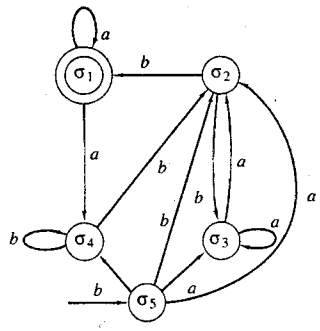


FIGURA 10.5.9 Un autómata de estado finito no determinista que acepta L^R.

Ejercicios

1. Determine los autómatas de estado finito equivalentes a los autómatas de estado finito no deterministas de los ejercicios 1-10, sección 10.4.

En los ejercicios 2-6, determine autómatas de estado finito que acepten las cadenas generadas por las gramáticas regulares.

2. La gramática del ejercicio 1, sección 10.3
3. La gramática del ejercicio 5, sección 10.3
4. $<S> ::= a<A> \mid a<B>$
   $<A> ::= a<B> \mid b<S> \mid b$
   $<B> ::= b<S> \mid b$
   con símbolo inicial $<S>$
5. $<S> ::= a<S> \mid a<A> \mid b<C> \mid a$
   $<A> ::= b<A> \mid a<C>$
   $<B> ::= a<S> \mid a$
   $<C> ::= a<B> \mid a<C>$
   con símbolo inicial $<S>$
6. $<S> ::= a<A> \mid a<B>$
   $<A> ::= b<S> \mid b$
   $<B> ::= a<B> \mid a<C>$
   $<C> ::= a<S> \mid b<A> \mid a<C> \mid a$
   con símbolo inicial $<S>$
7. Determine los autómatas de estado finito que acepten las cadenas de los ejercicios 21-29, sección 10.4.
8. Elimine los estados inalcanzables del autómata de estado finito de la figura 10.5.3, para determinar un autómata de estado finito equivalente, pero más sencillo.
9. Muestre que el autómata de estado finito no determinista de la figura 10.5.5 acepta una cadena α sobre $\{a, b\}$ si y sólo si α comienza con bb.
★ 10. Caracterice las cadenas aceptadas por los autómatas de estado finito no deterministas de las figuras 10.5.7 y 10.5.9.

En los ejercicios 11-21, determine un autómata de estado finito no determinista que acepte al conjunto dado de cadenas. Si S_1 y S_2 son conjuntos de cadenas, hacemos

$$S_1^+ = \{u_1u_2\cdots u_n \mid u_i \in S_1, n \in \{1, 2, \ldots\}\}; \quad S_1S_2 = \{uv \mid u \in S_1, v \in S_2\}.$$

11. $\mathrm{Ac}(A)^R$, donde A es el autómata del ejercicio 4, sección 10.2
12. $\mathrm{Ac}(A)^R$, donde A es el autómata del ejercicio 5, sección 10.2
13. $\mathrm{Ac}(A)^R$, donde A es el autómata del ejercicio 6, sección 10.2
14. $\mathrm{Ac}(A)^+$, donde A es el autómata del ejercicio 4, sección 10.2
15. $\mathrm{Ac}(A)^+$, donde A es el autómata del ejercicio 5, sección 10.2
16. $\mathrm{Ac}(A)^+$, donde A es el autómata del ejercicio 6, sección 10.2
17. $\mathrm{Ac}(A)^+$, donde A es el autómata de la figura 10.5.7
18. $\mathrm{Ac}(A_1)\mathrm{Ac}(A_2)$, donde A_1 es el autómata del ejercicio 4, sección 10.2, y A_2 es el autómata del ejercicio 5, sección 10.2
19. $\mathrm{Ac}(A_1)\mathrm{Ac}(A_2)$, donde A_1 es el autómata del ejercicio 5, sección 10.2, y A_2 es el autómata del ejercicio 6, sección 10.2

20. $\mathrm{Ac}(A_1)\mathrm{Ac}(A_1)$, donde A_1 es el autómata del ejercicio 6, sección 10.2
21. $\mathrm{Ac}(A_1)\mathrm{Ac}(A_2)$, donde A_1 es el autómata de la figura 10.5.7 y A_2 es el autómata del ejercicio 5, sección 10.2
22. Determine una gramática regular que genere el lenguaje L^R, donde L es el lenguaje generado por la gramática del ejercicio 5, sección 10.3.
23. Determine una gramática regular que genere el lenguaje L^+, donde L es el lenguaje generado por la gramática del ejercicio 5, sección 10.3.
24. Sea L_1 (respectivamente, L_2) el lenguaje generado por la gramática del ejercicio 5, sección 10.3 (respectivamente, ejemplo 10.5.5). Determine una gramática regular que genere el lenguaje L_1L_2.
★ 25. Muestre que el conjunto
$$L = \{x_1\cdots x_n \mid x_1\cdots x_n = x_n\cdots x_1\}$$
de cadenas sobre $\{a, b\}$ no es un lenguaje regular.
26. Muestre que si L_1 y L_2 son lenguajes regulares sobre I y S es el conjunto de todas las cadenas sobre I, entonces cada uno de los conjuntos $S - L_1$, $L_1 \cup L_2$, $L_1 \cap L_2$, L_1^+ y L_1L_2 es un lenguaje regular.
☆ 27. Muestre, mediante un ejemplo, que existen lenguajes libres de contexto L_1 y L_2 tales que $L_1 \cap L_2$ no sea libre de contexto.
★ 28. Demuestre o refute la afirmación: Si L es un lenguaje regular, también lo es
$$\{u^n \mid u \in L, n \in \{1, 2, \ldots\}\}.$$

NOTAS

Las referencias generales acerca de los autómatas, las gramáticas y los lenguajes son [Carroll; Cohen; Davis; Hopcroft; Kelley; McNaughton; Sudkamp; y Wood].

El desarrollo sistemático de la geometría fractal fue iniciado por Benoit B. Mandelbrot (véase [Mandelbrot, 1977, 1982]).

Una máquina de estado finito tiene una memoria primitiva interna, en el sentido de que recuerda el estado en el que se encuentra. Al permitir la existencia de una memoria externa, en la cual la máquina pueda leer y escribir datos, podemos definir máquinas más poderosas. También se obtienen otras mejoras permitiendo que la máquina revise la cadena de entrada en cualquier dirección, o bien permitiendo que la máquina modifique la cadena de entrada. Entonces, es posible caracterizar las clases de máquinas que aceptan lenguajes libres de contexto, lenguajes sensibles al contexto y lenguajes generados por gramáticas con estructura de frases.

Las **máquinas de Turing** forman una clase de máquinas con particular importancia. Como una máquina de estado finito, una máquina de Turing siempre se encuentra en un estado particular. Se supone que la cadena de entrada de una máquina de Turing está en una banda que es infinita en ambas direcciones. Una máquina de Turing revisa un carácter a la vez y, después de revisarlo se detiene o realiza una, ninguna o todas las acciones siguientes: modificar el carácter, moverse una posición a la izquierda o a la derecha, o cambiar estados. En particular, se puede modificar la cadena de entrada. Una máquina de Turing T acepta una cadena α si, al introducir α como entrada a T, T se detiene en un estado de aceptación. Se puede mostrar que un lenguaje L es generado por una gramática con estructura de frases si y sólo si existe una máquina de Turing que acepta L.

La importancia real de las máquinas de Turing surge de la creencia, ampliamente compartida, en el sentido de que cualquier función que se pueda calcular mediante alguna computadora digital, inclusive hipotética, se puede calcular mediante alguna máquina de Turing. Esta última afirmación se conoce como la **hipótesis de Turing** o **tesis de Church**. La tesis de Church implica que una máquina de Turing es el modelo abstracto correcto de una computadora digital. Estas ideas también sugieren la siguiente definición formal de algoritmo: Un **algoritmo** es una máquina de Turing tal que, al darle una cadena de entrada, se detendrá en algún momento.

CONCEPTOS BÁSICOS DEL CAPÍTULO

Sección 10.1

Retraso unitario de tiempo
Sumador en serie
Máquina de estado finito
Símbolo de entrada
Símbolo de salida
Estado
Función de estado siguiente
Función de salida
Estado inicial
Diagrama de transición
Cadenas de entrada y de salida para una máquina de estado finito
Flip-flop *SR*

Sección 10.2

Autómata de estado finito
Estado de aceptación
Cadena aceptada por un autómata de estado finito
Autómata de estado finito equivalente

Sección 10.3

Lenguaje natural
Lenguaje formal
Gramática con estructura de frases
Símbolo no terminal
Símbolo terminal
Producción
Símbolo inicial
Cadena derivable de manera directa
Cadena derivable
Derivación
Lenguaje generado por una gramática
Forma normal de Backus (= Forma de Backus-Naur = FBN)
Gramática sensible al contexto (= gramática de tipo 1)
Gramática libre de contexto (= gramática de tipo 2)
Gramática regular (= gramática de tipo 3)
Lenguaje sensible al contexto
Lenguaje libre de contexto
Lenguaje regular
Gramática Lindenmayer interactiva libre de contexto
Copo de nieve de von Koch
Curvas fractales

Sección 10.4

Dado un autómata de estado finito A, cómo construir una gramática regular G, tal que el conjunto de cadenas aceptadas por A sea igual al lenguaje generado por G (véase el teorema 10.4.2)
Autómata de estado finito no determinista
Cadena aceptada por un autómata de estado finito no determinista
Autómata de estado finito no determinista equivalente
Dada una gramática regular G, cómo construir un autómata de estado finito no determinista A tal que el lenguaje generado por G sea igual al conjunto de cadenas aceptadas por A (véase el teorema 10.4.11)

Sección 10.5

Dado un autómata de estado finito no determinista, cómo construir un autómata de estado finito determinista equivalente (véase el teorema 10.5.3)
Un lenguaje L es regular si y sólo si existe un autómata de estado finito que acepta las cadenas en L

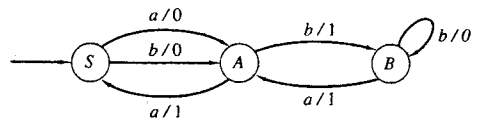
AUTOEVALUACIÓN DEL CAPÍTULO

Sección 10.1

1. Trace el diagrama de transición de la máquina de estado finito $(I, O, S, f, g, \sigma_0)$, donde $I = \{a, b\}$, $O = \{0, 1\}$ y $S = \{\sigma_0, \sigma_1\}$.

	f		g	
S \ I	a	b	a	b
σ_0	σ_1	σ_0	0	1
σ_1	σ_0	σ_1	1	0

2. Determine los conjuntos I, O y S, el estado inicial, la tabla que define el estado siguiente, y las funciones de salida para la siguiente máquina de estado finito.



3. Para la máquina de estado finito del ejercicio 1, determine la cadena de salida para la cadena de entrada *bbaa*.

4. Diseñe una máquina de estado finito cuya entrada sea una cadena de bits, tal que produzca 0 como salida a partir de que vea 001, y 1 en caso contrario.

Sección 10.2

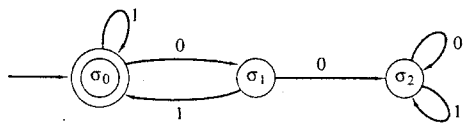
5. Trace el diagrama de transición del autómata de estado finito (I, S, f, A, S), donde $I = \{0, 1\}$, $S = \{S, A, B\}$ y $A = \{A\}$.

	f	
S \ I	**0**	**1**
S	A	S
A	S	B
B	A	S

6. ¿Es aceptada la cadena 11010 por el autómata de estado finito del ejercicio 5?

7. Dibuje el diagrama de transición de un autómata de estado finito que acepte al conjunto de cadenas sobre $\{0, 1\}$ que contengan un número par de ceros y un número impar de unos.

8. Caracterice el conjunto de cadenas aceptadas por el siguiente autómata de estado finito.



3. $I = \{a, b\}$, $S = \{\sigma_0, \sigma_1, \sigma_2, \sigma_3\}$, $A = \{\sigma_1\}$

S \ I	a	b
σ_0	$\varnothing$	$\{\sigma_3\}$
σ_1	$\{\sigma_1, \sigma_2\}$	$\{\sigma_3\}$
σ_2	$\varnothing$	$\{\sigma_0, \sigma_1, \sigma_3\}$
σ_3	$\varnothing$	$\varnothing$

4. $I = \{a, b, c\}$, $S = \{\sigma_0, \sigma_1, \sigma_2\}$, $A = \{\sigma_0\}$

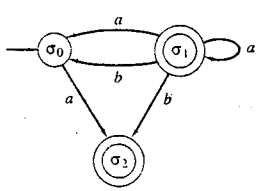
S \ I	a	b	c
σ_0	$\{\sigma_1\}$	$\varnothing$	$\varnothing$
σ_1	$\{\sigma_0\}$	$\{\sigma_2\}$	$\{\sigma_0, \sigma_2\}$
σ_2	$\{\sigma_0, \sigma_1, \sigma_2\}$	$\{\sigma_0\}$	$\{\sigma_0\}$

5. $I = \{a, b, c\}$, $S = \{\sigma_0, \sigma_1, \sigma_2, \sigma_3\}$, $A = \{\sigma_0, \sigma_3\}$

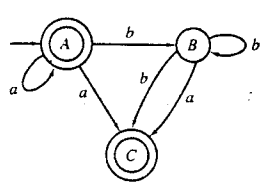
S \ I	a	b	c
σ_0	$\{\sigma_1\}$	$\{\sigma_0, \sigma_1, \sigma_3\}$	$\varnothing$
σ_1	$\{\sigma_2, \sigma_3\}$	$\varnothing$	$\varnothing$
σ_2	$\varnothing$	$\{\sigma_0, \sigma_3\}$	$\{\sigma_1, \sigma_2\}$
σ_3	$\varnothing$	$\varnothing$	$\{\sigma_0\}$

Para cada autómata de estado finito no determinista de los ejercicios 6-10, determine los conjuntos I, S y A, el estado inicial, y la tabla que define la función de estado siguiente.

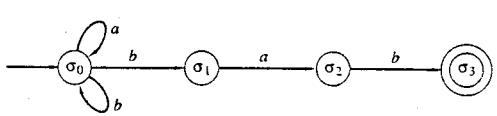
6.



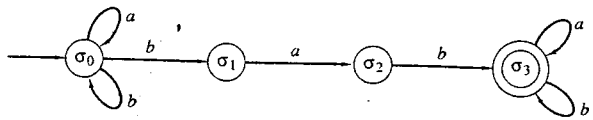
7.



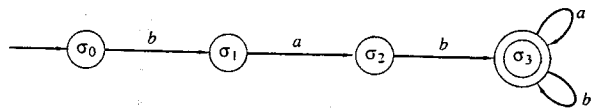
8.



9.



10.



11. Escriba las gramáticas regulares dadas por los autómatas de estado finito de los ejercicios 4-9, sección 10.2.

12. Represente las gramáticas de los ejercicios 1 y 5, sección 10.3, y el ejemplo 10.3.14 mediante autómatas de estado finito no deterministas.

13. ¿Es aceptada la cadena *bbabbb* por el autómata de estado finito no determinista de la figura 10.4.2? Demuestre su respuesta.

14. ¿Es aceptada la cadena *bbabab* por el autómata de estado finito no determinista de la figura 10.4.2? Demuestre su respuesta.

15. Demuestre que una cadena α sobre $\{a, b\}$ es aceptada por el autómata de estado finito no determinista de la figura 10.4.2 si y sólo si α contiene exactamente una a y termina con b.

16. ¿Es aceptada la cadena *aaabba* por el autómata de estado finito no determinista de la figura 10.4.3? Demuestre su respuesta.

17. ¿Es aceptada la cadena *aaaab* por el autómata de estado finito no determinista de la figura 10.4.3? Demuestre su respuesta.

18. Caracterice las cadenas aceptadas por el autómata de estado finito no determinista de la figura 10.4.3.

19. Muestre que las cadenas aceptadas por el autómata de estado finito no determinista del ejercicio 8 son precisamente aquellas cadenas sobre $\{a, b\}$ que terminan con *bab*.

☆ 20. Caracterice las cadenas aceptadas por los autómatas de estado finito no deterministas de los ejercicios 1-7, 9 y 10.

Diseñe autómatas de estado finito no deterministas que acepten las cadenas sobre $\{a, b\}$ con las propiedades dadas en los ejercicios 21-29.

21. Que comiencen con *abb* o *ba*

22. Que terminen con *abb* o *ba*

23. Que contengan *abb* o *ba*

★ **24.** Que contengan *bab* y *bb*

25. Que tengan cada *b* precedida y seguida por una *a*

26. Que comiencen con *abb* y terminen con *ab*

★ **27.** Que comiencen con *ab* pero que no terminen con *ab*

28. Que no contengan *ba* ni *bbb*

☆ 29. Que no contengan *abba* ni *bbb*

30. Escriba gramáticas regulares que generen las cadenas de los ejercicios 21-29.

10.5 RELACIONES ENTRE LENGUAJES Y AUTÓMATAS

En la sección anterior mostramos (teorema 10.4.2) que si A es un autómata de estado finito, entonces existe una gramática regular G, tal que $L(G) = \mathrm{Ac}(A)$. Como un recíproco parcial, mostramos (teorema 10.4.11) que si G es una gramática regular, entonces existe un autómata de estado finito no determinista A tal que $L(G) = \mathrm{Ac}(A)$. En esta sección mostramos (teorema 10.5.4) que si G es una gramática regular, entonces existe un autómata de estado finito A tal que $L(G) = \mathrm{Ac}(A)$. Este resultado será deducido del teorema 10.4.11, mostrando que cualquier autómata de estado finito no determinista se puede convertir en un autómata de estado finito equivalente (teorema 10.5.3). Primero ilustraremos el método por medio de un ejemplo.

EJEMPLO 10.5.1

Determinar un autómata de estado finito equivalente al autómata de estado finito no determinista de la figura 10.4.2.

El conjunto de símbolos de entrada no se modifica. Los estados constan de todos los subconjuntos

$$\varnothing, \quad \{\sigma\}, \quad \{C\}, \quad \{F\}, \quad \{\sigma, C\}, \quad \{\sigma, F\}, \quad \{C, F\}, \quad \{\sigma, C, F\}$$

del conjunto original $S = \{\sigma, C, F\}$ de estados. El estado inicial es $\{\sigma\}$. Los estados de aceptación son todos los subconjuntos

$$\{F\}, \quad \{\sigma, F\}, \quad \{C, F\}, \quad \{\sigma, C, F\}$$

de S que contienen un estado de aceptación del autómata de estado finito no determinista original.

Trazamos una arista de X a Y y la etiquetamos como x si $X = \varnothing = Y$ o si

$$\bigcup_{S \in X} f(S, x) = Y.$$

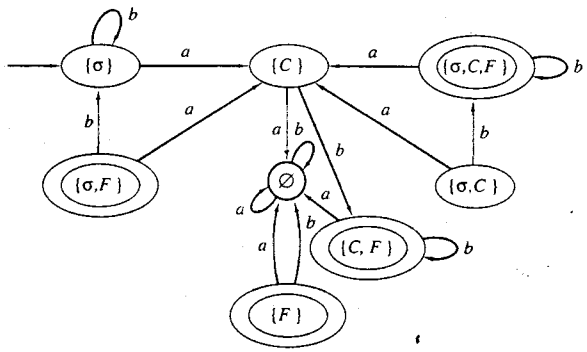


FIGURA 10.5.1 Un autómata de estado finito equivalente al autómata de estado finito no determinista de la figura 10.4.2.

Obtenemos el autómata de estado finito de la figura 10.5.1. Los estados

$$\{\sigma, F\}, \quad \{\sigma, C\}, \quad \{\sigma, C, F\}, \quad \{F\},$$

que nunca se pueden alcanzar, se pueden eliminar. Así, obtenemos el autómata de estado finito equivalente y simplificado de la figura 10.5.2.

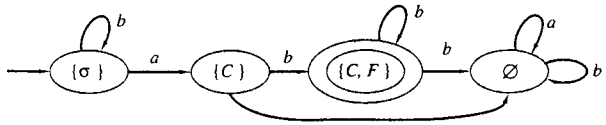


FIGURA 10.5.2 Una versión simplificada de la figura 10.5.1 (con los estados inalcanzables eliminados). □

EJEMPLO 10.5.2

El autómata de estado finito equivalente al autómata de estado finito no determinista del ejemplo 10.4.6 aparece en la figura 10.5.3.

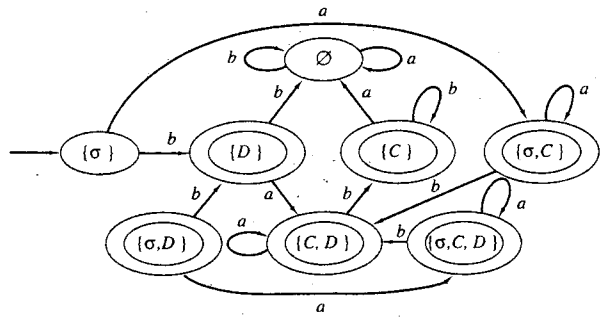


FIGURA 10.5.3 Un autómata de estado finito equivalente al autómata de estado finito no determinista de la figura 10.4.6. □

Ahora justificaremos formalmente el método de los ejemplos 10.5.1 y 10.5.2.

TEOREMA 10.5.3

Sea $A = (I, S, f, A, \sigma)$ un autómata de estado finito no determinista. Sean

(a) $S' = P(S)$.

(b) $S' = I$.

(c) $\sigma' = \{\sigma\}$.

(d) $A' = \{X \subseteq S \mid X \cap A \neq \varnothing\}$.

(e) $$f'(X, x) = \begin{cases} \varnothing & \text{si } X = \varnothing \\ \bigcup_{S \in X} f(S, x) & \text{si } X \neq \varnothing. \end{cases}$$

Entonces el autómata de estado finito $A' = (I', S', f', A', \sigma')$ es equivalente a A.

Demostración. Supongamos que la cadena $\alpha = x_1 \cdots x_n$ es aceptada por A. Entonces existen estados $\sigma_0, \ldots, \sigma_n \in S$ tales que

$$\sigma_0 = \sigma;$$
$$\sigma_i \in f(\sigma_{i-1}, x_i) \quad \text{para } i = 1, \ldots, n;$$
$$\sigma_n \in A.$$

Hacemos $Y_0 = \{\sigma_0\}$ y

$$Y_i = f'(Y_{i-1}, x_i) \quad \text{para } i = 1, \ldots, n.$$

Como

$$Y_1 = f'(Y_0, x_1) = f'(\{\sigma_0\}, x_1) = f(\sigma_0, x_1),$$

esto implica que $\sigma_1 \in Y_1$. Ahora,

$$\sigma_2 \in f(\sigma_1, x_2) \subseteq \bigcup_{S \in Y_1} f(S, x_2) = f'(Y_1, x_2) = Y_2.$$

De nuevo,

$$\sigma_3 \in f(\sigma_2, x_3) \subseteq \bigcup_{S \in Y_2} f(S, x_3) = f'(Y_2, x_3) = Y_3.$$

El argumento puede continuar (formalmente utilizaríamos la inducción) para mostrar que $\sigma_n \in Y_n$. Como σ_n es un estado de aceptación en A, Y_n es un estado de aceptación en A'. Así, en A', tenemos

$$f'(\sigma', x_1) = f'(Y_0, x_1) = Y_1$$
$$f'(Y_1, x_2) = Y_2$$
$$\vdots$$
$$f'(Y_{n-1}, x_n) = Y_n.$$

Por tanto, α es aceptada por A'.

Ahora, supongamos que la cadena $\alpha = x_1 \cdots x_n$ es aceptada por A'. Entonces existen subconjuntos $Y_0, \ldots, Y_n$ de S tales que

$$Y_0 = \sigma' = \{\sigma\};$$
$$f'(Y_{i-1}, x_i) = Y_i \quad \text{para } i = 1, \ldots, n;$$

existe un estado $\sigma_n \in Y_n \cap A$.

Como

$$\sigma_n \in Y_n = f'(Y_{n-1}, x_n) = \bigcup_{S \in Y_{n-1}} f(S, x_n),$$

existe $\sigma_{n-1} \in Y_{n-1}$ con $\sigma_n \in f(\sigma_{n-1}, x_n)$. De manera análoga, como

$$\sigma_{n-1} \in Y_{n-1} = f'(Y_{n-2}, x_{n-1}) = \bigcup_{S \in Y_{n-2}} f(S, x_{n-1}),$$

existe $\sigma_{n-2} \in Y_{n-2}$ con $\sigma_{n-1} \in f(\sigma_{n-2}, x_{n-1})$. Continuamos de esta manera para obtener

$$\sigma_i \in Y_i \quad \text{para } i = 0, \ldots, n,$$

con

$$\sigma_i \in f(\sigma_{i-1}, x_i) \quad \text{para } i = 1, \ldots, n.$$

En particular,

$$\sigma_0 \in Y_0 = \{\sigma\}.$$

Así, $\sigma_0 = \sigma$, el estado inicial en A. Como σ_n es un estado de aceptación en A, la cadena α es aceptada por A. ■

El siguiente teorema resume estos resultados y los de la sección anterior.

TEOREMA 10.5.4

Un lenguaje L es regular si y sólo si existe un autómata de estado finito que acepte precisamente las cadenas en L.

Demostración. Este teorema enuncia de otra forma los teoremas 10.4.2, 10.4.11 y 10.5.3. ■

EJEMPLO 10.5.5

Determinar un autómata de estado finito A que acepte precisamente las cadenas generadas por la gramática regular G con las producciones

$$\sigma \to b\sigma, \quad \sigma \to aC, \quad C \to bC, \quad C \to b.$$

El símbolo inicial es σ, el conjunto de símbolos terminales $\{a, b\}$, y el conjunto de símbolos no terminales es $\{\sigma, C\}$.

El autómata de estado finito no determinista A' que acepta $L(G)$ aparece en la figura 10.4.2. Un autómata de estado finito equivalente a A' aparece en la figura 10.5.1 y un autómata de estado finito simplificado equivalente A aparece en la figura 10.5.2. El autómata de estado finito A acepta precisamente las cadenas generadas por G. □

Cerramos esta sección dando algunas aplicaciones de los métodos y teoría hasta ahora desarrollados.

EJEMPLO 10.5.6 ***Un lenguaje que no es regular***

Mostrar que el lenguaje

$$L = \{a^n b^n \mid n = 1, 2, \ldots\}$$

no es regular.

Si L es regular, existe un autómata de estado finito A tal que $\text{Ac}(A) = L$. Supongamos que A tiene k estados. La cadena $\alpha = a^k b^k$ es aceptada por A. Consideremos el camino P que representa a α. Como existen k estados, algún estado σ es visitado al menos dos veces en la parte del camino que representa a a^k. Así, existe un ciclo C, tal que todas sus aristas tienen la etiqueta a y que contiene a σ. Modificamos el camino P para obtener un camino P', como sigue. Al llegar a σ en P, seguimos C. Después de regresar a σ sobre C, continuamos sobre P hasta el final. Si la longitud de C es j, el camino P' representa la cadena $\alpha' = a^{j+k} b^k$. Como P y P' terminan en el mismo estado σ' y σ' es un estado de aceptación, α' es aceptada por A. Esto es una contradicción, pues α' no es de la forma $a^n b^n$. Por tanto, L no es regular. □