Capítulo 3

Máquinas de estado finito y expresiones regulares

En este tema definiremos y estudiaremos máquinas de estado finito, llamadas también máquinas de estado finito secuenciales o autómatas finitos. Estos objetos matemáticos son los modelos para los ordenadores digitales y constituyen una importante herramienta en el diseño de circuitos físico-secuenciales, en el estudio de los lenguajes formales y de compiladores e intérpretes de varios lenguajes de programación. Desde un punto de vista teórico, las máquinas de estado finito son casos especiales de objetos más generales, tales como las máquinas de Turing, esenciales en el estudio de problemas en computabilidad.

Mientras que en un circuito combinacional la salida depende únicamente de la entrada en ese instante, una máquina secuencial es un sistema que acepta unas entradas y genera unas salidas en instantes de tiempo discretos, la salida en un instante depende de la entrada y de la condición interna (estado) de la máquina en ese instante y, además, esa entrada y el estado en ese instante determinan el estado que la máquina tendrá en el instante siguiente.

Todas las máquinas de estado finito tienen un conjunto de estados, incluído el estado inicial, un alfabeto fuente y una función de transición que a cada pareja de estado y dato de entrada le asigna el estado siguiente. Los estados de la máquina le dan unas capacidades de memoria limitadas. Algunas máquinas de estado finito producen un símbolo como dato de salida para cada transición y pueden utilizarse para modelar gran variedad de máquinas, entre las que se incluyen las máquinas expendedoras, los semáforos, los sumadores binarios y los reconocedores de lenguajes. También estudiaremos máquinas de estado finito que no generan datos de salida, pero tienen estados finales. Estas máquinas (autómatas) se utilizan con gran frecuencia en el reconocimiento

de lenguajes. Las cadenas que se reconocen son aquellas que transforman el estado inicial en un estado final. Los conceptos de gramática y máquina de estado finito pueden relacionarse entre sí. De hecho, los conjuntos que son reconocidos por una máquina de estado finito son los conjuntos generados por cierto tipo de gramática.

Supongamos que una máquina expendedora tiene dos tipos de productos A y B. El precio de cada producto es de 10 céntimos. La máquina admite únicamente monedas de 5 y 10 céntimos y devuelve el cambio necesario. Dispone de un botón rojo que expide el producto A y uno verde que hace lo mismo con el producto B. Elena quiso comprar el producto A y para ello introdujo consecutivamente dos monedas de 5 céntimos. Luego apretó el botón rojo y obtuvo el producto deseado. Representemos el proceso con una tabla, donde t_0 es el instante inicial cuando inserta su primera moneda y t_i con $i = 1, 2, 3$ son los instantes posteriores:

	t_0	t_1	t_2	t_3
Estado	s_0	s_1	s_2	s_0
Entrada	5	5	R	
Salida	nada	nada	A	

La máquina está en un estado de espera en el estado s_0. Espera que un cliente comience a insertar monedas hasta un total de 10 céntimos o más y oprima el botón para obtener el producto deseado. Si en cualquier momento del proceso, el total de monedas insertadas supera los diez céntimos, la máquina devuelve el cambio necesario antes de que el cliente oprima el botón correspondiente. En el instante t_0, Elena inserta su primera moneda de 5 céntimos. No recibe nada pero en el instante siguiente (t_1) la máquina está en el estado s_1 (tiene almacenados cinco céntimos). En este instante t_1, la máquina no devuelve nada pero, al depositar una nueva moneda de 5 céntimos, en el instante t_2 la máquina se encuentra en un nuevo estado s_2 (tiene almacenados diez céntimos). Elena todavía no recibe nada puesto que la máquina no sabe qué tipo de producto quiere. Al oprimir el botón rojo en el instante t_2, la máquina expide el producto A y en el instante siguiente t_3 se coloca de nuevo en el estado inicial s_0 (ningún céntimo almacenado). Las principales características de esta máquina son:

- En un instante dado, la máquina sólo puede estar en un estado de un conjunto finito de estados internos posibles.
- La máquina sólo acepta un número finito de entradas (en el ejemplo, monedas de cinco y diez céntimos, botones rojo y verde).

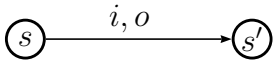
- Mediante cada combinación de entrada y estado interno, se produce una salida y un estado siguiente. El conjunto de salidas, para nuestra máquina es nada, monedas de cinco y diez céntimos y productos A y B.
- Los procesos de la máquina son secuenciales y se producen en instantes distintos. La máquina es determinista ya que la salida queda detemi-nada por el estado y la entrada.

3.1. Máquinas de estado finito con salida

Definición 53. *Una máquina de estado finito con salida $M = (S, I, O, f, g, s_0)$ consiste en un conjunto finito de estados S, un alfabeto (conjunto finito no vacío) de entradas I, un alfabeto de salidas O, un estado inicial s_0, una función de transición $f : S \times I \to S$ y una función de salida $g : S \times I \to O$.*

Las máquinas de este tipo se llaman *máquinas de Mealy* porque fue G. H. Mealy, en 1955, el primero que las estudió. Hay otro tipo importante de máquina de estado finito con salida, donde la salida está determinada sólo por el estado. Este tipo de máquina se llama *máquina de Moore* en honor a E. F. Moore, quien la definió en 1956.

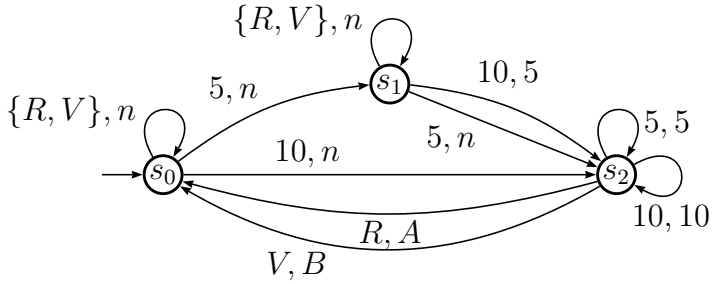
Una máquina $M = (S, I, O, f, g, s_0)$ puede describirse por una *tabla de estados*, que indica los valores de las funciones f y g, o por un *diagrama de estados*, grafo dirigido donde los vértices representan los estados de la máquina, el estado inicial se indica mediante una flecha que no proviene de otro estado y existe una flecha, etiquetada por "i, o", del estado s al estado s' si $f(s, i) = s'$ y $g(s, i) = o$.



Ejemplo 47. *La tabla de estados que le corresponde al ejemplo de la máquina expendendora de antes sería:*

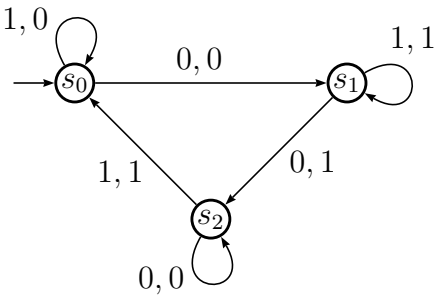
Estados	*Transicion*				*Salida*			
	Entrada				*Entrada*			
	5	10	R	V	5	10	R	V
s_0	s_1	s_2	s_0	s_0	n	n	n	n
s_1	s_2	s_2	s_1	s_1	n	5	n	n
s_2	s_2	s_2	s_0	s_0	5	10	A	B

El correspondiente diagrama de estados sería:



Ejemplo 48. *Tabla de estados y diagrama de estados de una máquina de estado finito:*

Estados	*Transición*		*Salida*	
	Entrada		*Entrada*	
	0	1	0	1
s_0	s_1	s_0	0	0
s_1	s_2	s_1	1	1
s_2	s_2	s_0	0	1



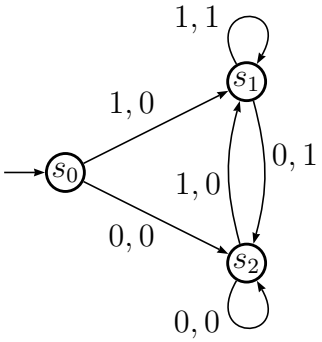
Para cada cadena de entrada $x = x_1 \dots x_k$ la máquina de estado finito produce una cadena de salida $y = y_1 \dots y_k$, siendo

$$\begin{aligned} t_0 &= s_0 \\ t_i &= f(t_{i-1}, x_i) \\ y_i &= g(t_{i-1}, x_i) \end{aligned}$$

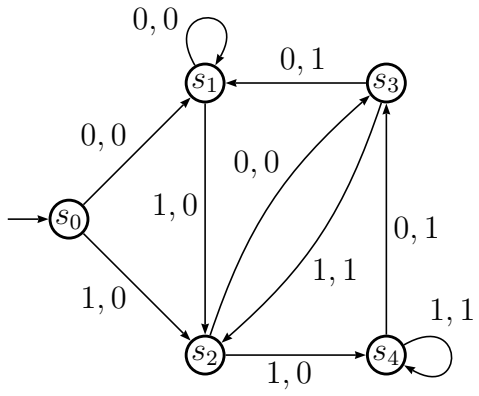
donde $i = 1, 2, \dots, k$.

En el ejemplo anterior, si $x = 10011$ es la secuencia de entrada, la secuencia de salida es $y = 00110$.

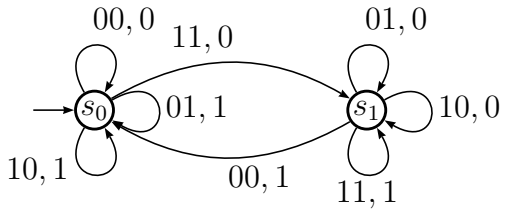
Ejemplo 49. ▪ *Se quiere diseñar una máquina de estado finito con una unidad de retardo que, dada una cadena de entrada $x = x_1 \dots x_k$ devuelva $0x_1 \dots x_{k-1}$. La máquina debe tener un estado inicial s_0 y debe recordar si la entrada previa ha sido 1 (s_1) o 0 (s_2). El diagrama de estados sería:*



- *En una máquina de estado finito con dos unidades de retardo que, dada una cadena de entrada* $x = x_1 \dots x_k$ *devuelva* $00x_1 \dots x_{k-2}$*, debemos tener un estado inicial* s_0 *y queremos recordar si la entrada previa ha sido* 00 (s_1)*,* 01 (s_2)*,* 10 (s_3) *o* 11 (s_4)*. El diagrama de estados sería:*

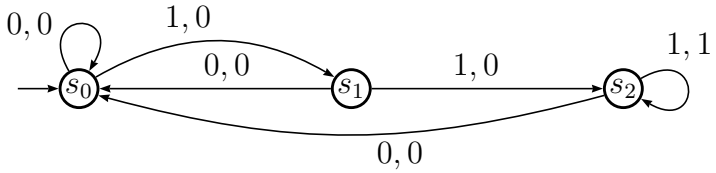


- *Se quiere diseñar una máquina de estado finito que efectúe suma binaria en serie. Las entradas posibles son los pares* $\{00, 01, 10, 11\}$*. Si la cifra de arrastre de la suma anterior es* 0*, el estado es* s_0 *y si la cifra de arrastre es* 1*, el estado es* s_1*. El diagrama de estados sería:*



- *En cierto esquema de codificación, el receptor de un mensaje sabe que ha habido un error de transmisión cuando aparecen tres unos consecutivos en un mensaje. Si se desea construir una máquina de estado finito que devuelva un* 1 *como salida si, y sólo si, los últimos tres bits recibidos son todos* 1*, se necesitan tres estados. El estado inicial* s_0 *recuerda que*

la entrada anterior, si existe, no era 1*. El estado* s_1 *recuerda que la entrada anterior era* 1*, pero que el valor previo al anterior, si existe, no era* 1*. El estado* s_2 *recuerda que las dos entradas previas han sido unos. El correspondiente diagrama de estados sería:*



Esta máquina es un ejemplo de *reconocedor de lenguajes* porque, dada una secuencia de entradas, la secuencia de salidas termina en 1 si, y sólo si, la cadena de entrada leída tiene una determinada propiedad. El reconocimiento de lenguajes es una aplicación importante de las máquinas de estado finito. Esta aplicación desempeña un papel fundamental en el diseño y construcción de compiladores para los lenguajes de programación.

3.2. Autómatas finitos

Un autómata finito (determinista) es un modelo matemático de una máquina que permite saber si una cadena de símbolos pertenece o no a un lenguaje definido sobre cierto alfabeto. Consiste en un conjunto finito de estados y un conjunto de transiciones entre estos estados, que dependen de los símbolos de la cadena de entrada. El autómata acepta una cadena de entrada si al terminar de leer todos los símbolos de esa cadena la máquina está en alguno de los posibles estados finales; si el estado no es final, entonces la cadena no pertenece al lenguaje que reconoce la máquina.

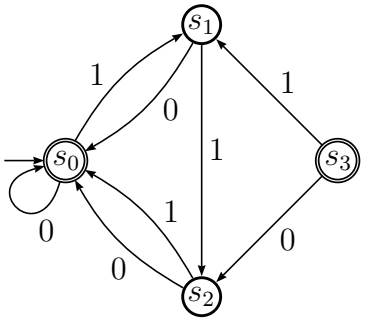
Definición 54. *Una autómata de estado finito* $M = (S, I, f, s_0, F)$ *consiste en un conjunto finito de estados* S*, un alfabeto de entradas* I*, un estado inicial* s_0*, una función de transición* $f : S \times I \to S$ *y un conjunto* F *de estados finales o de aceptación* ($F \subseteq S$)*.*

En el diagrama de transición de un autómata finito los estados finales están encerrados en un círculo doble. Para cada estado s_i y un símbolo de entrada a hay una única flecha de s_i a $f(s_i, a)$, que se etiqueta como a.

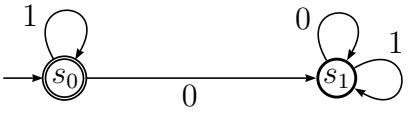
Ejemplo 50. ▪ *El diagrama de estados del autómata finito* $M = (S, I, f, s_0, F)$ *con* $S = \{s_0, s_1, s_2, s_3\}$*,* $I = \{0, 1\}$*,* $F = \{s_0, s_3\}$ *y cuya función de transición viene dada por la tabla:*

Estados	*Transicion*	
	Entrada	
	0	1
s_0	s_0	s_1
s_1	s_0	s_2
s_2	s_0	s_0
s_3	s_2	s_1

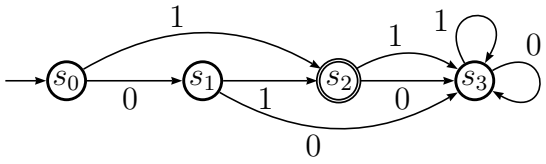
es el siguiente:



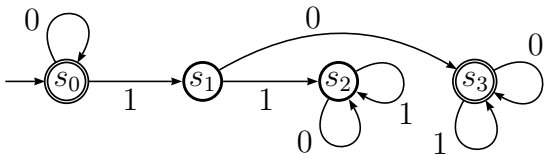
- $M_1 = (S, I, f, s_0, F)$ *si* $S = \{s_0, s_1\}$, $I = \{0, 1\}$, $F = \{s_0\}$



- $M_2 = (S, I, f, s_0, F)$ *si* $S = \{s_0, s_1, s_2, s_3\}$, $I = \{0, 1\}$, $F = \{s_2\}$



- $M_3 = (S, I, f, s_0, F)$ *si* $S = \{s_0, s_1, s_2, s_3\}$, $I = \{0, 1\}$, $F = \{s_0, s_3\}$



3.3. Lenguaje aceptado por un autómata

Dado un alfabeto cualquiera I se define I^* como el conjunto de todas las cadenas finitas que se pueden formar con los elementos de I, es decir los elementos de I^* son secuencias finitas $x = x_1 \dots x_k$ con $x_i \in I, k \in \mathbb{N}$ (se dice que x es de longitud k). Además si $k = 0$, se habla de la cadena vacía λ.

Si $x = x_1 \dots x_k$ e $y = y_1 \dots y_m$ son elementos de I^*, la concatenación de x e y es la cadena $xy = x_1 \dots x_k y_1 \dots y_m$, que también pertenece a I^*.

Definición 55. *Dado un alfabeto I, un lenguaje sobre I es un subconjunto de I^*.*

La función de transición $f : S \times I \to S$ de un autómata se puede extender a $f^* : S \times I^* \to S$ del modo siguiente. Dado $t \in S$ y $x = x_1 \dots x_k \in I^*$, llamaremos $t_0 = t$ y

$$t_i = f(t_{i-1}, x_i)$$

para $i = 1, 2, \dots, k$. De este modo, $f^*(t, x) := t_k$. Para cualquier estado $s \in S$, se tiene que $f^*(s, \lambda) = s$.

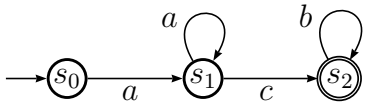
Definición 56. *Una palabra o cadena $x = x_1 \dots x_k \in I^*$ es* aceptada o reconocida *por M si $f^*(s_0, x) \in F$. El lenguaje formado por todas las palabras reconocidas por M se denota $L(M)$.*

Ejemplo 51. *En los autómatas M_1, M_2 y M_3 de los ejemplos anteriores, se tiene que*

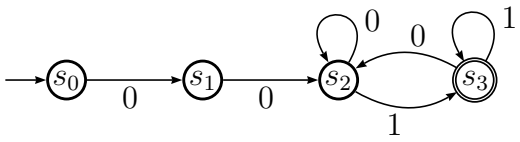
$$L(M_1) = \{1^n\,;\, n = 0, 1, 2, \dots\}$$
$$L(M_2) = \{1, 01\}$$
$$L(M_3) = \{0^n, 0^n 10x\,;\, x \text{ cualquier cadena binaria}\}$$

Ejemplo 52.

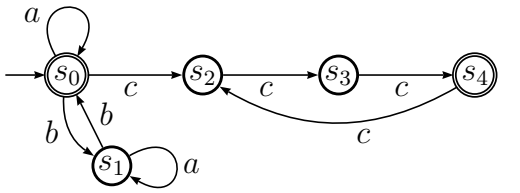
- *El autómata finito que acepta el lenguaje $L = \{a^n c b^m\,;\, n \geq 1, m \geq 0\}$ es $M = (S, I, f, s_0, F)$ con $S = \{s_0, s_1, s_2\}$, $I = \{a, b, c\}$, $F = \{s_2\}$ y diagrama de transición*



- *El autómata finito que acepta el lenguaje $L = \{00x1\,;\, x \in \{0, 1\}^*\}$ es $M = (S, I, f, s_0, F)$ con $S = \{s_0, s_1, s_2, s_3\}$, $I = \{0, 1\}$, $F = \{s_3\}$ y diagrama de transición*



- *El autómata finito que acepta el lenguaje* $L = \{xc^{3m}\,;\, x \in \{a,b\}^*, la\ cantidad\ de\ b's\ es\ par\ y\ m \geq 0\}$ *es* $M = (S, I, f, s_0, F)$ *con* $S = \{s_0, s_1, s_2, s_3, s_4\}$, $I = \{a, b, c\}$, $F = \{s_0, s_4\}$ *y diagrama de transición*



Es importante señalar que en cada una de las máquinas del ejemplo anterior falta determinar el comportamiento del autómata en algunos casos, sería suficiente añadir un estado "auxiliar", no terminal, que sería sumidero para aquellas flechas que faltan.

Nota. Para cada autómata se puede definir una máquina de Moore en la cual el conjunto de salidas es $O = \{0, 1\}$ y un estado es final si, y sólo si, la salida que tiene asociada es 1. Así, reconoce una cadena de entradas si la salida final es 1. Los autómatas son máquinas de estado finito aceptadoras (indican si una cadena pertenece o no al lenguaje), mientras que las máquinas de Mealy y de Moore son traductoras o transductoras (devuelven como resultado a una cadena de entrada una cierta cadena de salida).

3.4. Expresiones regulares y conjuntos regulares

Dados A y B dos subconjuntos de I^*, se define:

- La concatenación de A y B, que denotaremos AB, como el conjunto de todas las cadenas xy siendo x un elemento de A e y un elemento de B.
- A^k para $k \geq 0$, siendo $A^0 = \{\lambda\}$ y $A^{k+1} = A^k A$. De este modo, se tiene que

$$A^* = \bigcup_{k=0}^{\infty} A^k$$

A^* recibe el nombre de *cierre o clausura de Kleene* de A.

Definición 57. *Dado un alfabeto I, las expresiones regulares sobre I se definen por recurrencia del modo siguiente:*

- *el símbolo $\emptyset$ y el símbolo λ son expresiones regulares,*
- *el símbolo x para cada $x \in I$ es una expresión regular,*
- *si a y b son expresiones regulares sobre I, entonces ab, $a \vee b$ y a^* son expresiones regulares.*

Definición 58. *Toda expresión regular sobre I determina un conjunto regular. Los conjuntos regulares sobre I son subconjuntos de I^* que se definen por recurrencia del modo siguiente:*

- *$\emptyset$ y $\{\lambda\}$ son conjuntos regulares,*
- *Para cada $x \in I$, $\{x\}$ es un conjunto regular,*
- *Si A y B son conjuntos regulares, AB, $A \cup B$ y A^* son conjuntos regulares.*

Los símbolos $\emptyset$ y λ representan al conjunto vacío y al conjunto $\{\lambda\}$, respectivamente. Los símbolos x, donde $x \in I$, representan a los conjuntos $\{x\}$, donde $x \in I$. Si a y b representan a los conjuntos A y B, respectivamente, entonces ab, $(a \vee b)$ y a^* representan a los conjuntos AB, $(A \cup B)$ y A^*, respectivamente.

Un mismo conjunto regular puede estar representado por distintas expresiones regulares. Por ejemplo, si $I = \{a, b\}$, a^*b y $aa^* \vee b$ son expresiones regulares y ambas representan al conjunto $\{a^n b \ / \ n \in \mathbb{N} \cup 0\}$

Ejemplo 53. *Los conjuntos regulares definidos por las siguientes expresiones regulares son:*

- *para* 10^*, $\{1, 10, 100, \ldots\} = \{10^n \, ; \, n = 0, 1, \ldots\}$
- *para* $(10)^*$, $\{\lambda, 10, 1010, \ldots\} = \{(10)^n \, ; \, n = 0, 1, \ldots\}$
- *para* 1^*0^*, $\{1^n 0^m \, ; \, n, m = 0, 1, \ldots\}$
- *para* $0 \vee 01$, $\{0, 01\}$
- *para* $(0 \vee 1)^*$, $\{$*cadenas binarias de cualquier longitud*$\}$
- *para* $0(0 \vee 1)^*$, $\{$*cadenas binarias de cualquier longitud que empiezan por* $0\}$

- *para* (0^*1), $\{1, 01, 001, \ldots\} = \{0^n 1\,;\, n = 0, 1, \ldots\}$
- *para* $(0^*1)^*$, $\{\lambda, 1, 01, 001, \ldots, 11, 101, 1001, \ldots\} = \{\lambda\} \cup \{x1; x \in I^*\}$.

Teorema 5. *El lenguaje $L(M)$ reconocido por un autómata finito $M = (S, I, f, s_0, F)$ es un lenguaje regular sobre I.*

Demostración. Si M no posee estados finales, $L(M) = \emptyset$ luego $L(M)$ es regular. En caso contrario, si $F = \{s_1, \ldots, s_k\}$, se cumple que

$$L(M) = \bigcup_{i=1}^{k} \{\alpha \in I^* \;/\; f^*(s_0, \alpha) = s_i\}$$

es una unión de conjuntos. Teniendo en cuenta que cada uno de estos conjuntos es regular y que la unión de conjuntos regulares es regular, se obtiene que $L(M)$ es regular.

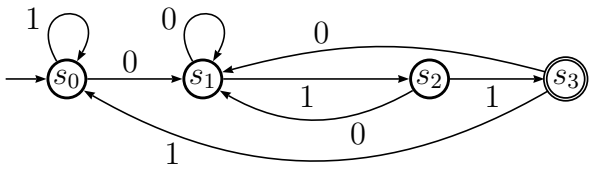
□

Teorema 6. *Un lenguaje L sobre un alfabeto I es regular si, y sólo si, existe un autómata finito M, con conjunto de entradas I, tal que $L = L(M)$.*

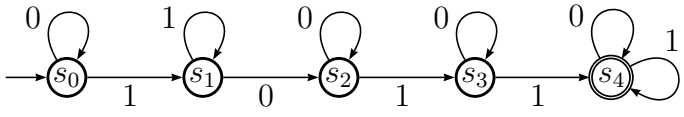
Se cumple que un autómata finito reconoce un solo lenguaje, pero un lenguaje puede ser reconocido por distintos autómatas.

Ejemplo 54.

- *El autómata finito que acepta todas las sucesiones binarias que terminan con los dígitos* 011 *es la dada por el diagrama de estados siguiente:* $L = \{x011 \;/\; x \in \{0, 1\}^*\}$ *(este conjunto regular es el que corresponde a la expresión regular* $(0 \vee 1)^*011$*).*



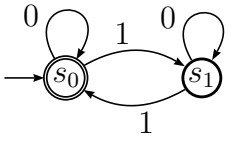
- *El autómata finito que acepta todas las sucesiones binarias que tienen la forma de cualquier cantidad de números* 0*, seguido por uno o más números* 1*, seguido por uno o más números* 0*, seguidos por un* 1*, seguido por cualquier cantidad de números* 0*, seguido por un* 1*, y entonces seguidos por cualquier cosa es:*
  $L = \{0^n 1^m 0^k 10^l 1x \;/\; n, l \in \mathbb{N} \cup \{0\},\ m, k \in \mathbb{N},\ x \in \{0, 1\}^*\}$

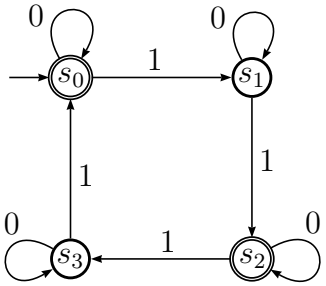


3.5. Simplificación de autómatas finitos

La simplificación de un autómata finito M implica la identificación de "estados equivalentes" en M de manera que podamos obtener un autómata con menos estados que acepte el mismo lenguaje.

Ejemplo 55. *Los autómatas M y M' aceptan aquellas cadenas formadas por un número par de 1's. Sin embargo, la estructura de M' es más complicada.*





Definición 59. *Si $M = (S, I, f, s_0, F)$ es un autómata finito, dos estados $s_i, s_j \in S$ son* $*$-equivalentes *si para toda cadena $x \in I^*$, se verifica que*

$$f^*(s_i, x) \in F \iff f^*(s_j, x) \in F.$$

Esta situación la denotaremos $s_i \, R_* \, s_j$ e indica que, para cualquier cadena de entrada, ambos estados son terminales o ninguno de ellos lo es.

R_* es una relación de equivalencia en el conjunto de estados S. El proceso para determinar si dos estados s_i y s_j son $*$-equivalentes es muy complejo puesto que requeriría analizar infinitas cadenas de entrada. En la práctica se procede de manera recursiva:

Definición 60. *Si $M = (S, I, f, s_0, F)$ es un autómata finito y k es un número natural, dos estados $s_i, s_j \in S$ son* k-equivalentes *si para toda cadena $x \in I^*$ de longitud menor o igual que k, se verifica que*

$$f^*(s_i, x) \in F \iff f^*(s_j, x) \in F.$$

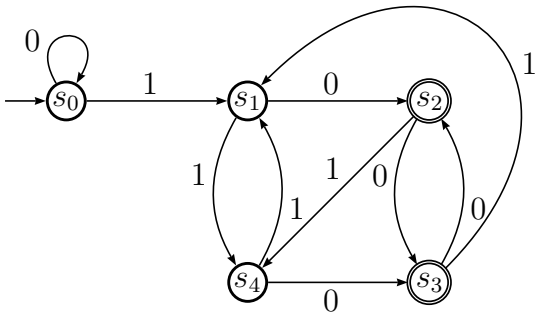
Esta situación la denotaremos $s_i \, R_k \, s_j$.[1] *R_k es una relación de equivalencia en el conjunto de estados S y divide al conjunto S en clases de equivalencia (partición).*

- Para cada $k \geq 1$, se tiene que si dos estados son k-equivalentes, también son $(k-1)$-equivalentes.
- Para cada $k \geq 1$, un clase de equivalencia de la relación R_k es un subconjunto de una clase de equivalencia de la relación R_{k-1}.
- Dos estados son $*$-equivalentes si, y sólo si, son k-equivalentes para cualquier $k \geq 0$.

Teorema 7. *Dos estados s_i y s_j son k-equivalentes si, y sólo si, son $(k-1)$-equivalentes y para cualquier $n \in I$ se tiene que $f(s_i, n)$ y $f(s_j, n)$ son $(k-1)$-equivalentes.*

Veamos en un ejemplo cómo se procede de modo recursivo para obtener estados equivalentes en un autómata.

Ejemplo 56. *Consideremos el autómata M cuyo diagrama de estados viene dado por:*



I) *Las clases de equivalencia para la relacion R_0 son $\{s_0, s_1, s_4\}$ y $\{s_2, s_3\}$*

II) *Nótese que s_0 y s_1 no son R_1 equivalentes ya que $f(s_0, 0) = s_0$, $f(s_1, 0) = s_2$ y s_0 y s_2 no son 0-equivalentes. Sin embargo $f(s_4, 0) = s_3$, $f(s_4, 1) = s_1$ y $f(s_1, 1) = s_4$ siendo $s_1 \, R_1 \, s_4$ y $s_2 \, R_1 \, s_3$. Las clases de equivalencia para R_1 son $\{s_0\}$, $\{s_1, s_4\}$ y $\{s_2, s_3\}$.*

III) *Las clases de equivalencia para R_2 son $\{s_0\}$, $\{s_1, s_4\}$ y $\{s_2, s_3\}$.*

[1] Dos estados son 0-equivalentes si ambos son terminales o los dos son no terminales.

Teorema 8. *Dado un autómata M, existe un entero k tal que las relaciones R_k y R_{k+1} son iguales y, en consecuencia, son iguales a la relación R_*. Por lo tanto, los conjuntos cocientes de las tres relaciones coinciden.*

Ejemplo 57. *En el ejemplo 56, se tiene que las clases de equivalencia para R_* son $\{s_0\}$, $\{s_1, s_4\}$ y $\{s_2, s_3\}$.*

Teorema 9. *Dado un autómata $M = (S, I, f, s_0, F)$, si $s_i \, R_* \, s_j$ entonces $f(s_i, n) \, R_* \, f(s_j, n)$ para cualquier $n \in I$.*

Demostración. $f(s_i, n) \, R_* \, f(s_j, n) \iff f^*(f(s_i, n), x) \, R_0 \, f^*(f(s_j, n), x), \; \forall x \in I^* \iff f^*(s_i, nx) \; R_0 \; f^*(s_j, nx), \; \forall x \in I^* \iff s_i \; R_* \; s_j$ □

Definición 61. *Dado un autómata finito $M = (S, I, f, s_0, F)$,* el autómata cociente *de M es $\bar{M} = (\bar{S}, I, \bar{f}, \bar{s_0}, \bar{F})$, siendo*

- *$\bar{S}$ el conjunto cociente de S por la relación de equivalencia R_*.*
- *$\bar{F} = \{\bar{s} \,;\, s \in F\}$*
- *$\bar{f}(\bar{s}, n) = \overline{f(s, n)}$, para cada $s \in S$ y $n \in I$.*

Ejemplo 58. *En el ejemplo 56, el autómata cociente tiene como diagrama de estados:*