Autómatas finitos no deterministas (AFnD)

Elvira Mayordomo

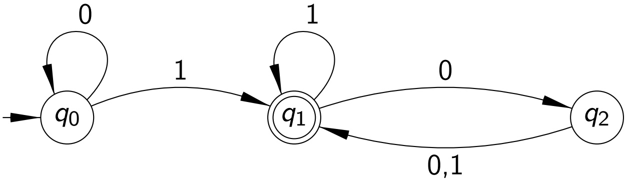
Universidad de Zaragoza

1 de octubre de 2012

Contenido de este tema

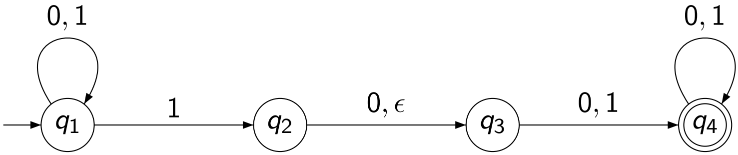
- Introducción y ejemplos de autómatas finitos no deterministas
- Definición de autómata finito no determinista
- Equivalencia de AFD y AFnD (autómatas finitos deterministas y no deterministas)
- Método para convertir un AFnD en AFD

Recordad los autómatas finitos deterministas



- La computación del autómata con entrada 011 es (q_0, q_0, q_1, q_1) que me dice la secuencia de estados por los que pasa con entrada 011
- Cada entrada me da exactamente una computación. Tengo siempre como mucho una opción desde un estado si leo un símbolo $\leftarrow$ Esto se llama determinismo

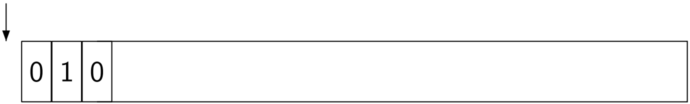
Primer ejemplo de autómata finito no determinista

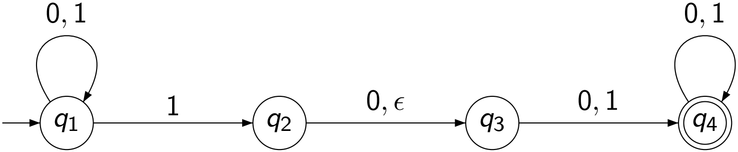


- Desde q_1 con el símbolo 1 hay dos opciones posibles
- Desde q_2 hay una posibilidad de moverse sin leer ningún símbolo (la marcada como ϵ)

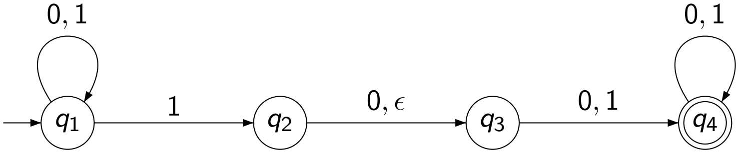
Cómo funciona el autómata con entrada 010

Configuración inicial





Cómo funciona el autómata con entrada 010

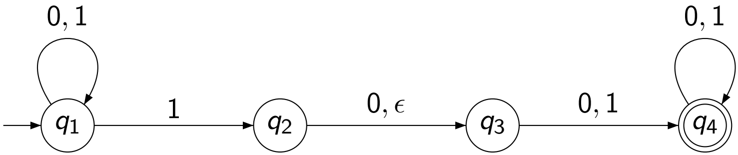


- Puede seguir la computación (q_1, q_1, q_1, q_1)
- Puede seguir (q_1, q_1, q_2, q_3)
- Puede seguir $(q_1, q_1, q_2, q_3, q_4)$

¿Cuál es la buena? Todas son posibles

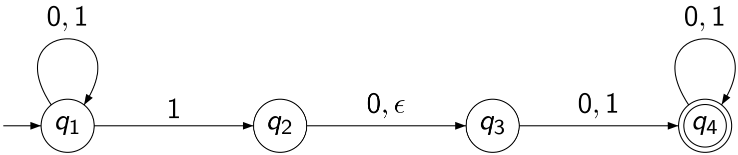
¿Acepta 010? Sí, porque alguna de las computaciones posibles lleva a estado final (q_4)

Con entrada 010110



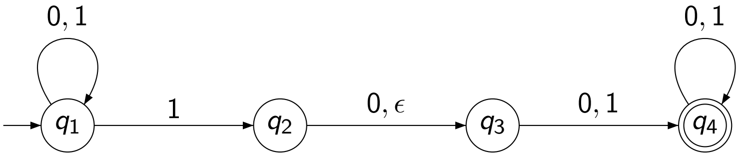
Acepta porque una de las posibles computaciones termina en q_4

Con entrada 01



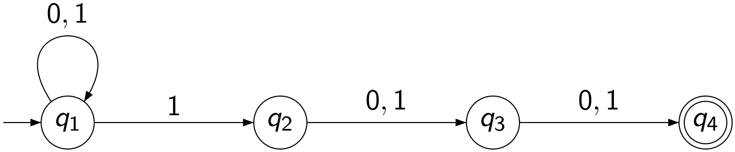
Rechaza porque ninguna de las posibles computaciones termina en estado final

¿Qué lenguaje acepta?



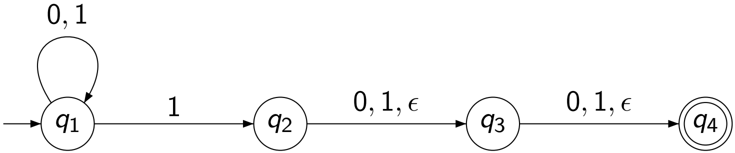
Las cadenas con un 1 que no sea el último símbolo

¿Qué lenguaje acepta este?



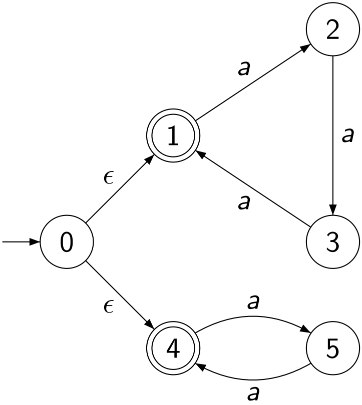
Las cadenas que tienen 1 como antepenúltimo símbolo

¿Y este?



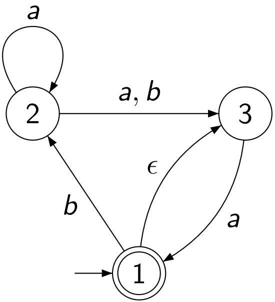
Las cadenas que tienen 1 como último, penúltimo o antepenúltimo símbolo

Dos últimos ejemplos



$(aa)^* + (aaa)^*$

Dos últimos ejemplos



$(a + ba^*ba)^*$, las cadenas que tienen un número par de bs y después de cada b par tienen una a, y la cadena vacía

Definición formal de autómata finito NO determinista

Definición

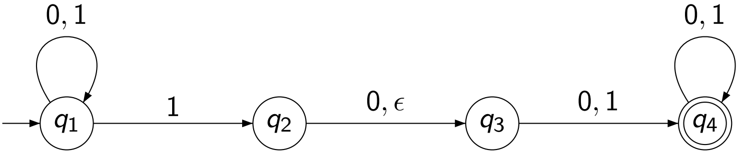
Un autómata finito no determinista (AFD) es $M = (Q, \Sigma, \delta, q_0, F)$ *tal que*

- Q *es el conjunto finito de estados*
- Σ *es el alfabeto de entrada*
- $q_0 \in Q$ *es el estado inicial*
- $F \subseteq Q$ *es el conjunto de los estados finales.*
- $\delta : Q \times (\Sigma \cup \epsilon) \to \mathcal{P}(Q)$ *es la función de transición*
  $\delta(q, a) = R$ *quiere decir que si estoy en el estado* q *y leo el símbolo* a *puedo ir a cualquiera de los estados* $q' \in R$

Notación: $\mathcal{P}(Q)$ es el conjunto de subconjuntos de Q

Representado un autómata

- Lo más usual es la representación gráfica



Representado un autómata

- También podemos indicar quiénes son los estados, estado inicial, estados finales y tabla de transición

$$Q = \{q_1, q_2, q_3, q_4\}$$

$$\textit{Estado inicial } q_1$$

$$F = \{q_4\}$$

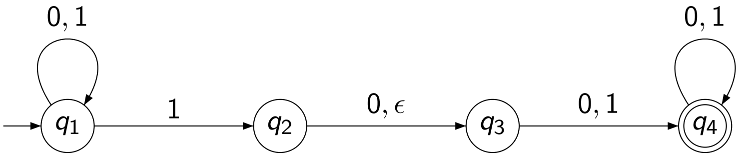
δ	0	1	ϵ
q_1	$\{q_1\}$	$\{q_1, q_2\}$	$\emptyset$
q_2	$\{q_3\}$	$\emptyset$	$\{q_3\}$
q_3	$\{q_4\}$	$\{q_4\}$	$\emptyset$
q_4	$\{q_4\}$	$\{q_4\}$	$\emptyset$

Computación de un autómata no determinista

Dado un AFnD $M = (Q, \Sigma, \delta, q_0, F)$

- Una computación de M con entrada $w = w_1 \dots w_n$ es $(r_0, r_1, \dots, r_m)$ que cumple:
  - $w = y_1 \dots y_m$ con $y_i \in \Sigma \cup \epsilon$
  - r_0 es el estado inicial
  - $r_{i+1} \in \delta(r_i, y_{i+1})$
- Una computación aceptadora de M con entrada w es una computación $(r_0, r_1, \dots, r_m)$ de M con entrada w que cumple $r_m \in F$

Ejemplo de computación



Si M es este autómata

- (q_1, q_1, q_1, q_1) es una computación de M con entrada 010
- $(q_1, q_1, q_2, q_3, q_4)$ es una computación aceptadora de M con entrada 010

Lenguaje aceptado por un autómata no determinista

Formalmente, dado un AFnD $M = (Q, \Sigma, \delta, q_0, F)$ el lenguaje aceptado por M es $L(M)$ definido como

$$L(M) = \{w \quad | \quad \text{existe una computación aceptadora de } M \text{ con entrada } w\}$$

Autómatas finitos deterministas y no deterministas

- Vamos a ver que los AFD y AFnD aceptan los mismos lenguajes
- La ventaja de los AFnD es que pueden ser mucho más pequeños/simples (Buscar un AFnD que acepte las cadenas de longitud múltiplo de 2 o múltiplo de 3)
- La ventaja de los AFD es que son más fáciles de analizar y simplificar

Equivalencia de AFD y AFnD

Teorema

Dado un autómata finito no determinista (AFnD) M, existe un autómata finito determinista (AFD) M' tal que $L(M) = L(M')$.

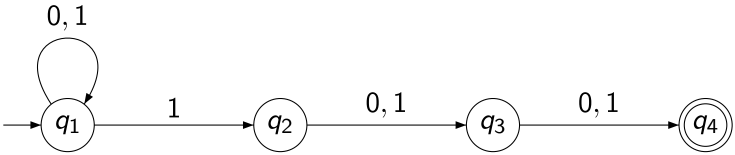
Demostración de la equivalencia de AFD y AFnD

- La veremos en la pizarra (y hay un resumen en la web)
- De la demostración sacaremos un método que usaremos para convertir AFnD en AFD
- Primero trataremos un caso más fácil y luego el general

Método para determinizar (AFnD sin ϵ-transiciones)

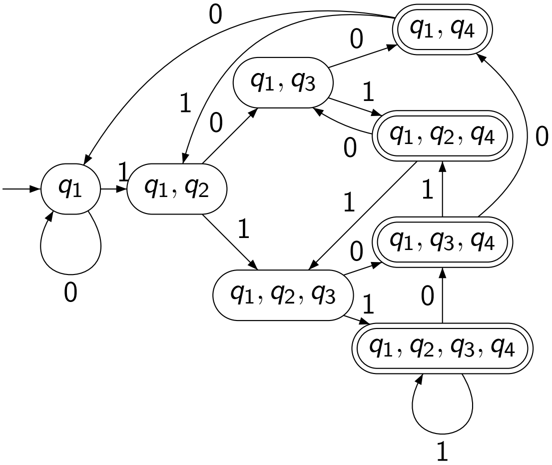
1. Construir una tabla con columnas una por cada $a \in \Sigma$.
2. En la primera fila escribir $\{q_0\}$ y en la columna a escribir $\delta(\{q_0\}, a)$, es decir, todos los estados a los que puedo llegar desde q_0 con entrada a.
3. Copiar las casillas de la fila anterior como principio de nuevas filas.
4. Para cada fila R pendiente, rellenar la fila R escribiendo en cada columna a $\delta(R, a)$, es decir, todos los estados a los que puedo llegar desde algún estado de R con entrada a.
5. Copiar las casillas de la fila anterior como principio de nuevas filas.
6. Repetir los pasos 4 y 5 hasta que no queden filas por rellenar.

Ejemplo de determinizar (AFnD sin ϵ-transiciones)

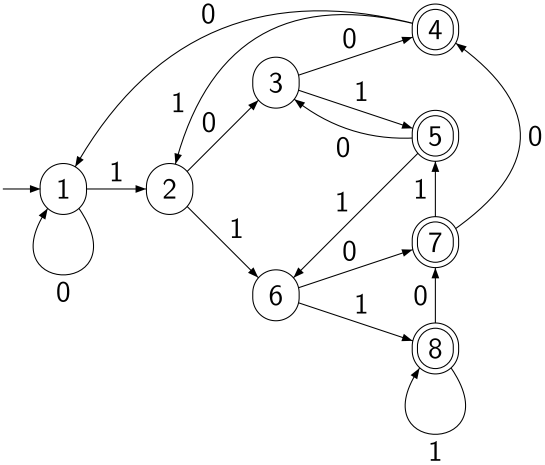


	0	1
$\{q_1\}$	$\{q_1\}$	$\{q_1, q_2\}$
$\{q_1, q_2\}$	$\{q_1, q_3\}$	$\{q_1, q_2, q_3\}$
$\{q_1, q_3\}$	$\{q_1, q_4\}$	$\{q_1, q_2, q_4\}$
$\{q_1, q_2, q_3\}$	$\{q_1, q_3, q_4\}$	$\{q_1, q_2, q_3, q_4\}$
$\{q_1, q_4\}$	$\{q_1\}$	$\{q_1, q_2\}$
$\{q_1, q_2, q_4\}$	$\{q_1, q_3\}$	$\{q_1, q_2, q_3\}$
$\{q_1, q_3, q_4\}$	$\{q_1, q_4\}$	$\{q_1, q_2, q_4\}$
$\{q_1, q_2, q_3, q_4\}$	$\{q_1, q_3, q_4\}$	$\{q_1, q_2, q_3, q_4\}$

Ejemplo de determinizar (AFnD sin ϵ-transiciones)



Ejemplo de determinizar (AFnD sin ϵ-transiciones)

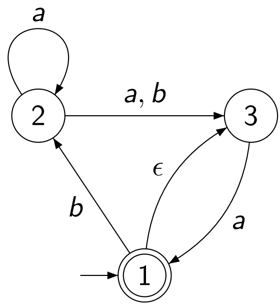


- Es útil cambiar los nombres de los estados

Método para determinizar (AFnD con ϵ-transiciones)

1. Construir una tabla con columnas una por cada $a \in \Sigma$.
2. En la primera fila escribir el inicial $I = E(\{q_0\})$, es decir, todos los estados a los que puedo llegar desde q_0 con ϵ^*.
3. En la primera fila, en la columna a escribir $\bigcup_{r \in I} E(\delta(r, a))$, es decir, todos los estados a los que puedo llegar desde I con entrada $a\epsilon^*$.
4. Copiar las casillas de la fila anterior como principio de nuevas filas.
5. Para cada fila R pendiente, rellenar la fila R escribiendo en cada columna a, $\bigcup_{r \in R} E(\delta(r, a))$, es decir, todos los estados a los que puedo llegar desde algún estado de R con entrada $a\epsilon^*$.
6. Copiar las casillas de la fila anterior como principio de nuevas filas.
7. Repetir los pasos 5 y 6 hasta que no queden filas por rellenar.

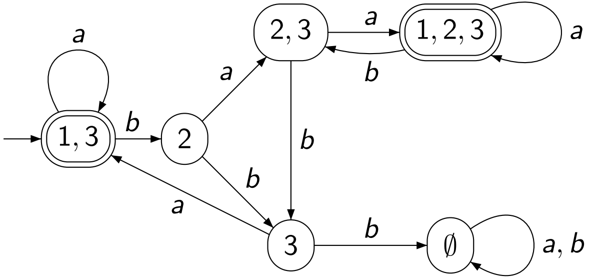
Ejemplo de determinizar (AFnD con ϵ-transiciones)



	a	b
$\{1,3\}$	$\{1,3\}$	$\{2\}$
$\{2\}$	$\{2,3\}$	$\{3\}$
$\{2,3\}$	$\{1,2,3\}$	$\{3\}$
$\{3\}$	$\{1,3\}$	$\emptyset$
$\{1,2,3\}$	$\{1,2,3\}$	$\{2,3\}$
$\emptyset$	$\emptyset$	$\emptyset$

Ejemplo de determinizar (AFnD con ϵ-transiciones)

	a	b
$\{1,3\}$	$\{1,3\}$	$\{2\}$
$\{2\}$	$\{2,3\}$	$\{3\}$
$\{2,3\}$	$\{1,2,3\}$	$\{3\}$
$\{3\}$	$\{1,3\}$	$\emptyset$
$\{1,2,3\}$	$\{1,2,3\}$	$\{2,3\}$
$\emptyset$	$\emptyset$	$\emptyset$



Teorema

- (Recordad la definición) Un lenguaje A es regular si existe un AFD M con $A = L(M)$.

- Teorema
  Un lenguaje A es regular si existe un AFnD N con $A = L(N)$.

¿Qué hemos aprendido?

- Para cada AFnD (autómata no determinista) existe un AFD (autómata determinista) que acepta el mismo lenguaje
- Así que para ver que existe un AFD que reconoce un lenguaje basta con encontrar un AFnD
- Por tanto para saber si un lenguaje es regular basta con encontrar un autómata no determinista que lo reconozca

¿Qué hemos aprendido?

- En prácticas vimos como convertir cada AFD en un AFD mínimo que hace lo mismo
- El AFD mínimo es único
- **Para saber si dos AFnD A_1 y A_2 hacen lo mismo** (reconocen el mismo lenguaje) basta con
  - Convertirlos a AFDs M_1 y M_2
  - Minimizar M_1 y M_2, convirtiéndolos en N_1 y N_2
  - Si N_1 y N_2 son el mismo autómata entonces A_1 y A_2 hacen lo mismo

Bibliografía

- Sipser (2a edición), páginas 47 a 58 (en sección 1.2) .
- Kelley, secciones 2.5, 2.6 y 2.7.