Autómatas Finitos Deterministas

Alan Reyes-Figueroa

Teoría de la Computación (Aula 04) 17.julio.2024

Transición extendida
Grafos y tablas de transición
Algunas técnicas de demostración

Representación alternativa: *Tabla de Transición*

Estados finales (*)

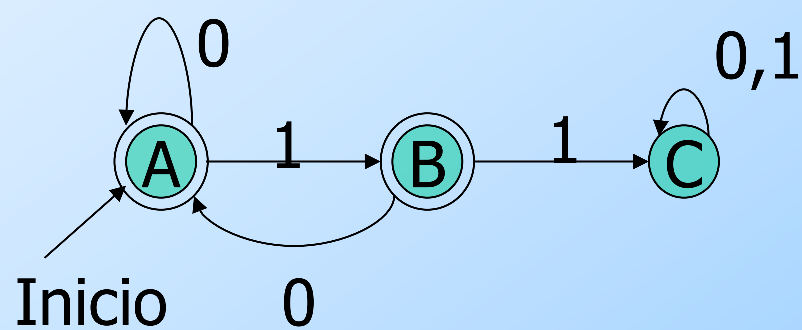
Columnas = Símbolos de entrada

Estado inicial (→)

	0	1
→ * A	A	B
* B	A	C
C	C	C

Filas = Estados

Cada entrada es δ Valuadad en (fila, columna).



Función de transición extendida

- Queremos describir el efecto de una cadena de entrada en un AFD, mediante extender la función de transición a

$$\hat{\delta}: K \times \Sigma^* \to K$$

(estado x cadena).

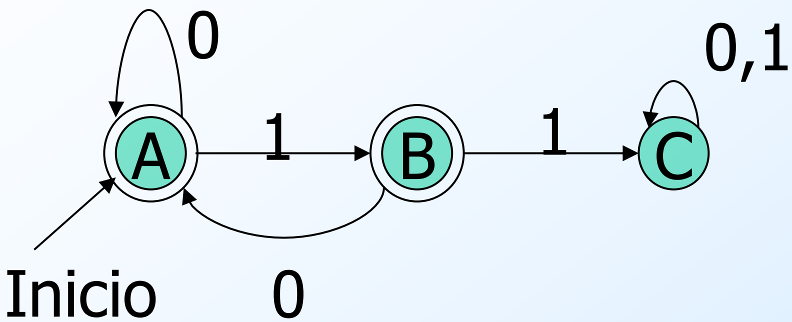
- Idea: extender δ para describir la transición de un estado q con una secuencia de entradas $a_1 a_2 a_3 \dots a_n$.

Definición recursiva de δ extendida

- La inducción se hace sobre la longitud de la cadena.

- Paso base: $\delta(q, \epsilon) = q$
- Inducción: $\delta(q,wa) = \delta(\delta(q,w),a)$
  - Nota: w es una cadena; a es un símbolo, por convención.

Ejemplo: δ extendida



	0	1
A	A	B
B	A	C
C	C	C

$$\delta(B,011) = \delta(\delta(B,01),1)$$
$$= \delta(\delta(\delta(B,0),1),1) = \delta(\delta(\delta(\delta(B,\varepsilon),0),1),1)$$
$$= \delta(\delta(\delta(B,0),1,1) = \delta(\delta(A,1),1)$$
$$= \delta(B,1) = C$$

Ejemplo: δ extendida

Otra forma: $w = 011$

$\delta(B,0) = A$

$\delta(B,01) = \delta(\delta(B,0),1) = \delta(A,1) = B$

$\delta(B,011) = \delta(\delta(B,01),1) = \delta(B,1) = C$

Ejemplo: δ extendida

En el libro de Lewis y Papadimitriou, definen:

Sean (q,w) y (q',w') configuraciones de M.

Si w = aw' y δ(q,a) = q', decimos que (q,w) produce (q',w') en un solo paso.

$$(q,w) \vdash_M (q',w')$$

En nuestro ejemplo:

$$\delta(B,011) \vdash_M \delta(A,11) \vdash_M \delta(B,1) \vdash_M \delta(C,\varepsilon) = C$$

Podemos escribir $\delta(B,011) \vdash_M^* C$

δ-hat

- No distinguimos entre la función delta original y la función delta extendida o $\hat{\delta}$.

El motivo:

- $\hat{\delta}(q,a) = \delta\big(\hat{\delta}(q,\epsilon),a\big) = \delta(q,a)$

Detas extendidas

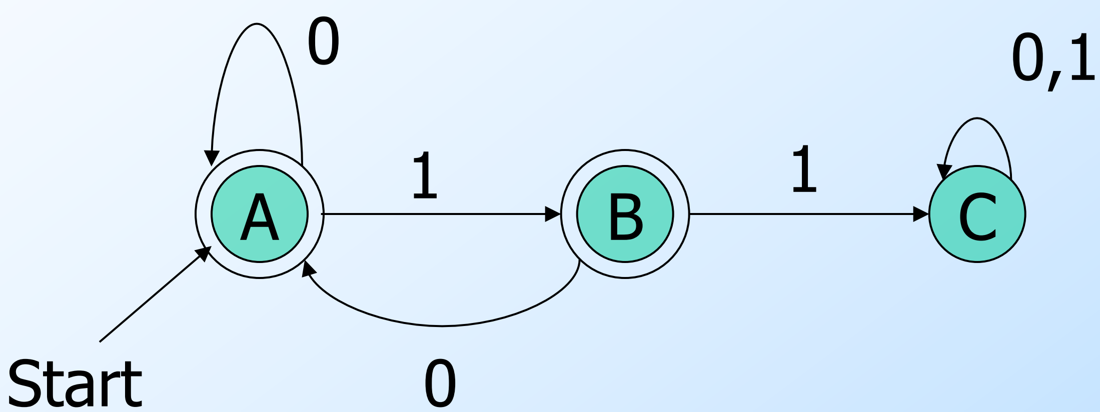
Lenguaje de un AFD

- Autómatas de todos los tipos definen lenguajes.
- Para un autómata finito determinista M, L(M) consiste del conjunto de todas las cadenas (o caminos) desde el estado inicial *s* a algún estado final.
- Formalmente:

  $L(M) = \{w \in \Sigma^*: \delta(q_0, w)$ está en $F\}$.

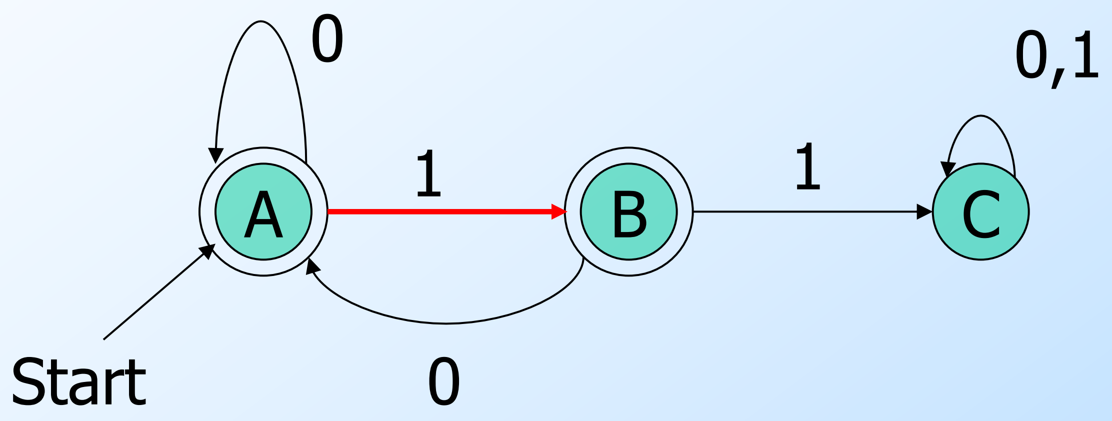
Ejemplo: Cadenas en un lenguaje

La cadena '101' está en el lenguaje aceptado por el siguiente autómata finito determinista:
Estado inicial = A



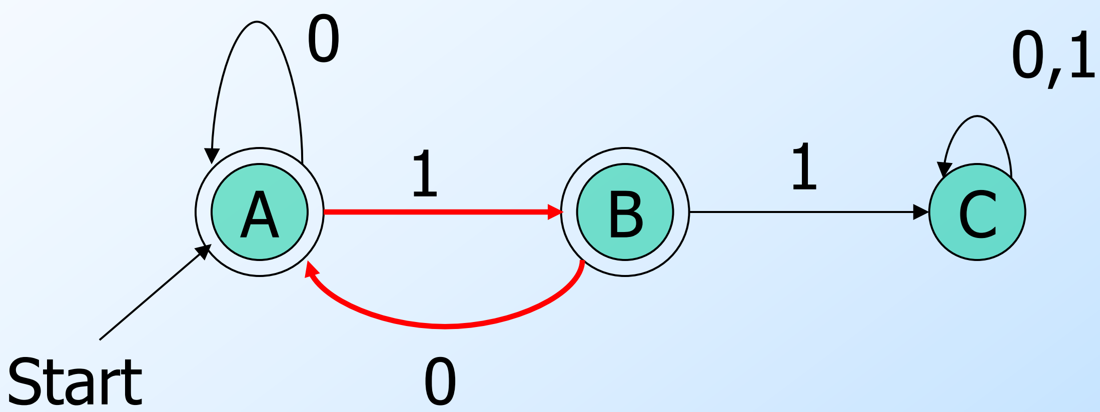
Ejemplo: Cadenas en un lenguaje

Seguir el arco 1



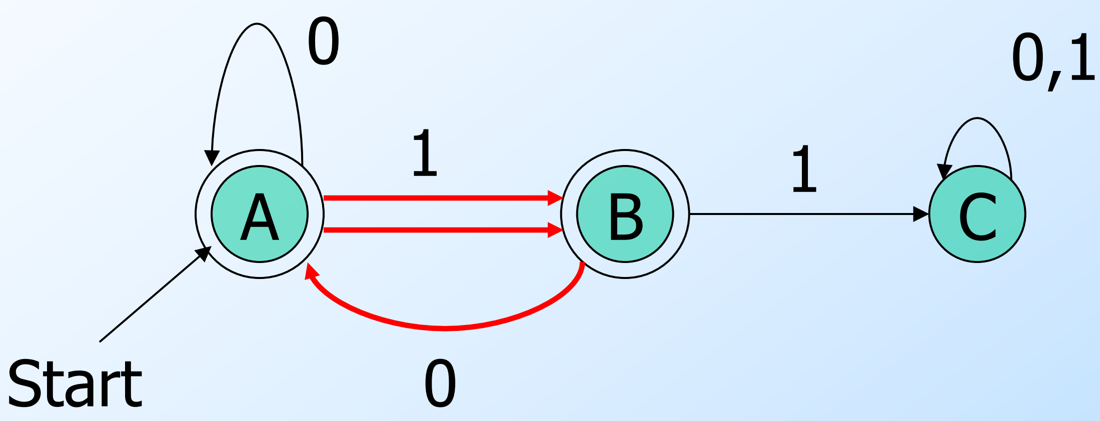
Ejemplo: Cadenas en un lenguaje

Luego seguir el arco 0



Ejemplo: Cadenas en un lenguaje

Finalmente seguir el arco 1 de nuevo.
Fin de la cadena: Como resultado obtenemos un estado de aceptación, la cadena se acepta.



Ejemplo – final

- El lenguaje del autómata finito determinista anterior es:

$\{w : w \in \{0,1\}^*$ y w no posee dos 1's consecutivos$\}$

Tales que…

Estas condiciones sobre w son ciertas.

Conjunto de cadenas w…

Pruebas de equivalencia (de conjuntos)

- En ocasiones, es necesario mostrar que dos descripciones de conjuntos son el mismo conjunto.
- Aquí, un conjunto es "el lenguaje L(M) del autómata antrerior," y el otro conjunto es "el conjunto de cadenas w de 0's y 1's sin dos 1's consecutivos."

Pruebas – (2)

- Recordemos que par probar S = T,

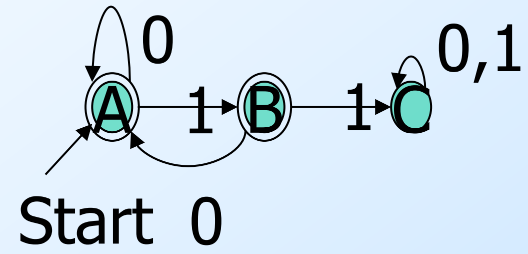
Debemos mostrar ambas partes: S ⊆ T and T ⊆ S. Esto es:

1. si w está en S, concluir que w está en T.
2. si w está en T, concluir que w está en S.

- Aquí, S = L(M)

  T = "w: w no posee 1's consecutivos"

Parte 1: $S \subseteq T$



- A mostrar: si w es aceptada por M, entonces w no tiene 1's consecutivos.
- La prueba se hace por inducción sobre la longitud de w.
- Truco Importante: Expandir la hipótesis de inducción para que sea más detallada que lo que se quiere mostrar.

La hipótesis de inducción

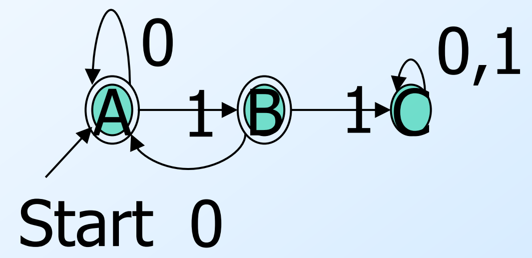
1. Si $\delta(A, w) = A$, entonces w no posee 1's consecutivos y no termina en 1.
2. If $\delta(A, w) = B$, entonces w no posee 1's consecutivos y termina en un 1.

- Base: $|w| = 0$; i.e., $w = \epsilon$.
  - (1) vale ya que ϵ no posee 1's.
  - (2) vale *por vacuidad*, ya que $\delta(A, \epsilon) \neq B$.

"longitud"

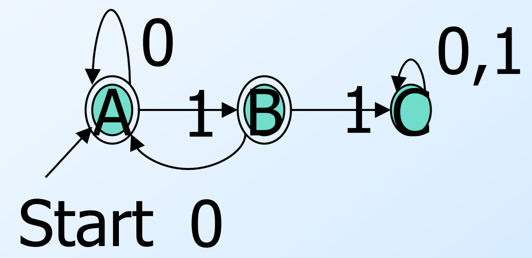
Importante:
Cuando la parte "si" no se cumple.

Paso inductivo



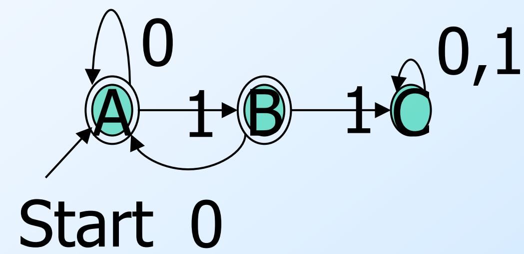
- Asumir que (1) y (2) valen para cadenas de longitud < |w|, (donde |w| es al menos 1).
- Como w no es la cadena vacía ε, podemos escribir w = xa, donde *a* es el símbolo final de w, y *x* es un cadena de longitud |w|-1.
- La hipótesis vale para *x*.

Paso inductivo – (2)



- Debemos probar (1) y (2) para w = xa.
- (1) para w: si δ(A, w) = A, entonces w no posee 1's consecutivos y no termina en 1.
- Como δ(A, w) = A, δ(A, x) debe ser A ó B, y *a* debe ser 0 (véa el autómata).
- Por hipótesis inductiva, x no tiene 11's.
- Por tanto, w no posee 11's y no termina en 1.

Paso inductivo – (3)

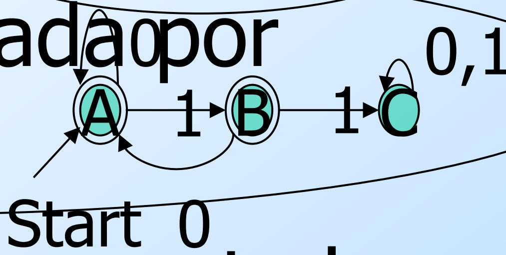


- Debemos mostrar (2) si w = xa: Si δ(A, w) = B, entonces w no tiene 11's y termina en 1.
- Como δ(A, w) = B, δ(A, x) debe ser A, y *a* debe ser 1 (véa el autómata).
- Por hipótesis inductiva, x no tiene 11's y no termina en 1.
- Portanto, w no posee 11's y termina en 1.

Parte 2: $T \subseteq S$

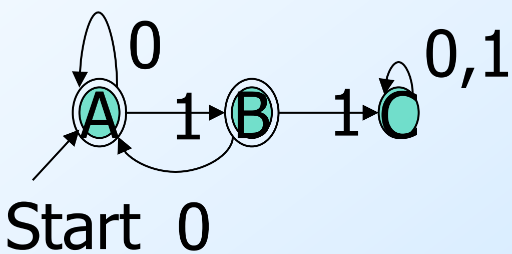
X

- Ahora debemos mostrar: si w no tiene 11's, entonces w es aceptada por



Y

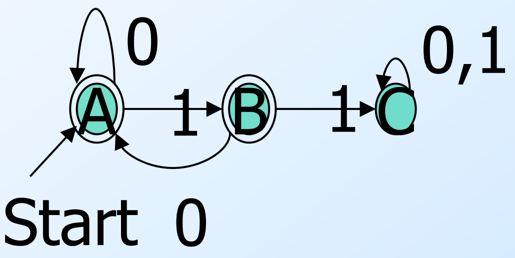
- *Contrapositiva* : si w no es aceptada por



entonces w tiene 11's.

Nota: la contrapositiva de "si X, entonces Y" es equivalente a "si no Y, entonces no X."

Contrapositiva

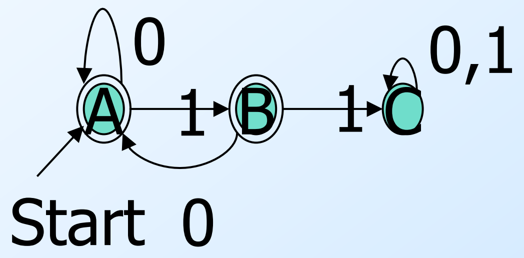


- Como sólo hay una transición desde cada estado y cada entrada, entonces una cadena w lleva al autómata a un único estado posible.

  (determinismo).

- La única forma de no aceptar w es que llegemos al estado C.

Contrapositiva – (2)



- La única manera de obtener C, es decir, δ(A,w) = C, es que si w = x1y, *x* vaya al estado B, y *y* es la cola de w que sigue después de llegar al estado C por vez primera.
- Si δ(A,x) = B, entonces x = z1, para alguna cadena z.
- Portanto, w = z11y, y posee 11's.

Lenguajes regulares

- Un lenguaje L es *regular* si se puede representar por una expresión regular.
- Alternativamente: L es regular si L=L(M) para algún autómata finito determinista M.
  - Nota: autómata M sólo debe aceptar las cadenas en L, no otras.
- (Existen lenguajes no regulares)

Ejemplo: Un lenguaje no regular

$L_1 = \{0^n 1^n \mid n \geq 1\}$

- Nota: a^n denota *n* a's consecutivas.
  - *e.g.*, $0^4 = 0000$, $1^7 = 1111111$
- L_1 = "conjunto de cadenas de 0's y 1's, que consisten de *n* 0's seguidas de *n* 1's, donde $n \geq 1$.
- Así, $L_1 = \{01, 0011, 000111, \ldots\}$

Otro Ejemplo

L_2 = {w ϵ {(,)}* : w es *balanceada* }

- Paréntesis balanceados se refiere a secuencias de paréntesis que pueden aparecer en una expresión aritmética (escrita correctamente).
- *e.g.*: (), ()(), (()), (()()),…

Muchos lenguajes son regulares

- Aparecen en contextos diversos, poseen propiedades importantes.
- Ejemplo: las cadenas que representan un número en notación de *punto flotante* en su Java, Phyton, C++, … es un lenguaje regular.

Ejemplo: Un lenguaje regular

L_3 = { w en {0,1}*: w vista como un número entero en representación binaria es divisible por 23}

- El autómata:
  - 23 estados, etiquetados 0, 1,…,22.
  - Corresponden a las 23 clases de residuos módulo 23.
  - Inicio y único estado final es 0.

Transiciones para L_3

- Si una cadena w representa el entero i, asumimos que $\delta(0, w) = i \bmod 23$.
- Entonces w0 representa el entero 2i, así $\delta(i \bmod 23, 0) = (2i) \bmod 23$.
- Similarmente: w1 representa 2i+1, así $\delta(i \bmod 23, 1) = (2i+1) \bmod 23$.
- Ejemplos: $\delta(15,0) = 30 \bmod 23 = 7$;

  $\delta(11,1) = 23 \bmod 23 = 0$.

Otro Ejemplo

L_4 = { w en {0,1}*: w, vista como el reverso de la representación binaria de un entero i es divisible por 23}

- Ejemplo: 01110100 está en L_4, porque su reversa, 00101110 es 46 en binario.
- Difícil construir el autómata AFD.
- Pero… existe un teorema que dice que el reverso de un lenguaje regular, es regular.