Tema 3: Propiedades de los Conjuntos Regulares

Jose A. Garcia

Universidad de Granada

Contenido

- Lema de bombeo y aplicaciones:
  - Demostrar que un lenguaje no es regular.
- Operaciones con conjuntos regulares: complementario, intersección, diferencia, homomorfismos.
- Algoritmos para autómatas:
  - Lenguaje vacío-no vacío
  - Lenguaje finito-infinito
  - Igualdad de lenguajes de dos autómatas
- Minimización de autómatas: estados indistinguibles.

Lema de Bombeo

Sea L un conjunto regular, entonces *existe* un $n \in \mathbb{N}$ tal que $\forall z \in L$, *si* $|z| \geq n$, entonces z *se puede* expresar de la forma $z = uvw$ donde

1. $|uv| \leq n$
2. $|v| \geq 1$
3. $(\forall i \geq 0) \quad uv^i w \in L$

además n puede ser el número de estados de cualquier autómata que acepte el lenguaje L.

Comentarios

- Es un lema.

Comentarios

- Es un lema.
- Es útil para demostrar que un determinado lenguaje no es regular.

Comentarios

- Es un lema.
- Es útil para demostrar que un determinado lenguaje no es regular.
- Es una condición necesaria para los conjuntos regulares.

Comentarios

- Es un lema.
- Es útil para demostrar que un determinado lenguaje no es regular.
- Es una condición necesaria para los conjuntos regulares.
- No es una buena guía para descubrir si un lenguaje es o no regular.

Demostración

Supongamos $z \in L, \quad z = a_1 a_2 \ldots a_m$ con $m \geq n$.

Demostración

Supongamos $z \in L, \quad z = a_1 a_2 \ldots a_m$ con $m \geq n$.

Sea $z' = a_1 a_2 \ldots a_n$ la palabra formada por los n primeros símbolos de z.

Consideremos el vector de estados $(q_{i_0}, q_{i_1}, \ldots, q_{i_n})$ donde q_{i_0} es el estado inicial, q_0, y $q_{i_j} = \delta(q_{i_{j-1}}, a_j)$

Demostración

Supongamos $z \in L$, $\quad z = a_1 a_2 \dots a_m$ con $m \geq n$.

Sea $z' = a_1 a_2 \dots a_n$ la palabra formada por los n primeros símbolos de z.

Consideremos el vector de estados $(q_{i_0}, q_{i_1}, \dots, q_{i_n})$ donde q_{i_0} es el estado inicial, q_0, y $q_{i_j} = \delta(q_{i_{j-1}}, a_j)$

Hay n estados distintos y el vector es de longitud $n+1$: algún estado se debe de repetir. Supongamos que se repiten $q_{i_k} = q_{i_l}$.

Demostración

Supongamos $z \in L$, $\quad z = a_1 a_2 \ldots a_m$ con $m \geq n$.

Sea $z' = a_1 a_2 \ldots a_n$ la palabra formada por los n primeros símbolos de z.

Consideremos el vector de estados $(q_{i_0}, q_{i_1}, \ldots, q_{i_n})$ donde q_{i_0} es el estado inicial, q_0, y $q_{i_j} = \delta(q_{i_{j-1}}, a_j)$

Hay n estados distintos y el vector es de longitud $n+1$: algún estado se debe de repetir. Supongamos que se repiten $q_{i_k} = q_{i_l}$.

La descomposición es

$u = a_1 \ldots a_k$, $v = a_{k+1} \ldots a_l$, $w = a_{l+1} \ldots a_m$

Demostración

Supongamos $z \in L$, $\quad z = a_1 a_2 \dots a_m$ con $m \geq n$.

Sea $z' = a_1 a_2 \dots a_n$ la palabra formada por los n primeros símbolos de z.

Consideremos el vector de estados $(q_{i_0}, q_{i_1}, \dots, q_{i_n})$ donde q_{i_0} es el estado inicial, q_0, y $q_{i_j} = \delta(q_{i_{j-1}}, a_j)$

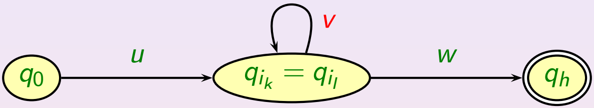
Hay n estados distintos y el vector es de longitud $n+1$: algún estado se debe de repetir. Supongamos que se repiten $q_{i_k} = q_{i_l}$.

La descomposición es

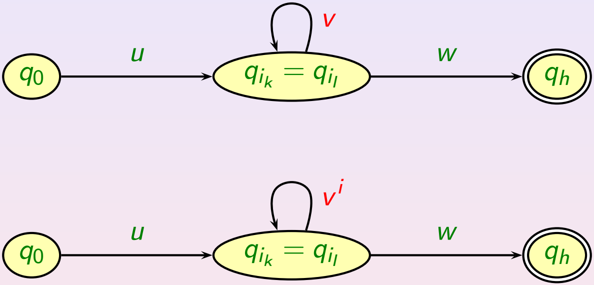
$u = a_1 \dots a_k$, $v = a_{k+1} \dots a_l$, $w = a_{l+1} \dots a_m$

Tenemos que se verifica $uv^i w \in L$, ya que leyendo v pasamos por un ciclo: de un estado a él mismo

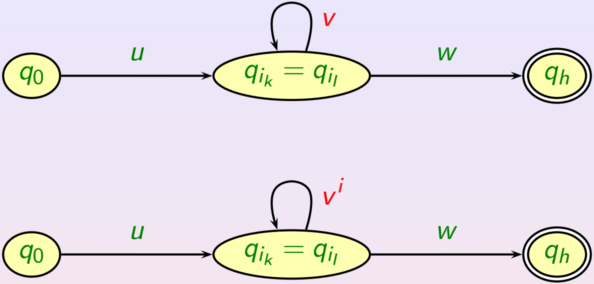
En el Diagrama de Transición



En el Diagrama de Transición



En el Diagrama de Transición



Pertenecen al lenguaje porque llega al mismo estado final que z.

$|uv| \leq n$ porque el ciclo se produce como máximo al leer n símbolos.

$|v| \geq 1$ porque en el vector de estados siempre se leía un símbolo para pasar al siguiente.

Un lenguaje no es regular

$\forall n \in \mathbb{N}$, existe una palabra $z \in L$, con $|z| \geq n$ tal que para toda descomposición

$$z = uvw$$

Si se verifica

- $|uv| \leq n$
- $|v| \geq 1$

entonces

$$\exists i \in \mathbb{N}, \text{tal que } uv^i w \notin L$$

$\{0^j 1^j : j \geq 0\}$ no es regular

$\forall n \in \mathbb{N}$, existe una palabra $z \in L$, con $|z| \geq n$,
tal que para toda descomposición $z = uvw$. Si se verifica

- $|uv| \leq n$
- $|v| \geq 1$

entonces

$$\exists i \in \mathbb{N}, \text{ tal que } uv^i w \notin L$$

$\{0^j1^j : j \geq 0\}$ no es regular

$\forall n \in \mathbb{N}$, existe una palabra $z \in L$, con $|z| \geq n$, $\quad z = 0^n1^n$
tal que para toda descomposición $z = uvw$. Si se verifica

- $|uv| \leq n$
- $|v| \geq 1$

entonces

$$\exists i \in \mathbb{N}, \text{ tal que } uv^iw \notin L$$

$\{0^j 1^j : j \geq 0\}$ no es regular

$\forall n \in \mathbb{N}$, existe una palabra $z \in L$, con $|z| \geq n$, $\quad z = 0^n 1^n$
tal que para toda descomposición $z = uvw$. Si se verifica

- $|uv| \leq n$
- $|v| \geq 1$

entonces tenemos que $u = 0^k, v = 0^l, w = 0^{n-k-l} 1^n$, con $l \geq 1$.

$$\exists i \in \mathbb{N}, \text{ tal que } uv^i w \notin L$$

$\{0^j 1^j : j \geq 0\}$ no es regular

$\forall n \in \mathbb{N}$, existe una palabra $z \in L$, con $|z| \geq n$, $\quad z = 0^n 1^n$
tal que para toda descomposición $z = uvw$. Si se verifica

- $|uv| \leq n$
- $|v| \geq 1$

entonces tenemos que $u = 0^k, v = 0^l, w = 0^{n-k-l}1^n$, con $l \geq 1$.

$$\exists i \in \mathbb{N},\ \text{tal que } uv^i w \notin L$$

Haciendo $i = 2, uv^2 w = 0^k 0^{2l} 0^{n-k-l} 1^n = 0^{n+l} 1^n \notin L$.

$\{0^{j^2} : j \geq 0\}$ no es regular

$\forall n \in \mathbb{N}$, existe una palabra $z \in L$, con $|z| \geq n$,
tal que para toda descomposición $z = uvw$. Si se verifica

- $|uv| \leq n$
- $|v| \geq 1$

entonces
$\exists i \in \mathbb{N}$, tal que $uv^i w \notin L$

$\{0^{j^2} : j \geq 0\}$ no es regular

$\forall n \in \mathbb{N}$, existe una palabra $z \in L$, con $|z| \geq n$, $\quad z = 0^{n^2}$
tal que para toda descomposición $z = uvw$. Si se verifica

- $|uv| \leq n$
- $|v| \geq 1$

entonces
$\exists i \in \mathbb{N}$, tal que $uv^i w \notin L$

$\{0^{j^2} : j \geq 0\}$ no es regular

$\forall n \in \mathbb{N}$, existe una palabra $z \in L$, con $|z| \geq n$, $z = 0^{n^2}$
tal que para toda descomposición $z = uvw$. Si se verifica

- $|uv| \leq n$
- $|v| \geq 1$

entonces tenemos que $u = 0^k, v = 0^l, w = 0^{n^2-l-k}$, con $l \geq 1, l \leq n$.
$\exists i \in \mathbb{N}$, tal que $uv^iw \notin L$

$\{0^{j^2} : j \geq 0\}$ no es regular

$\forall n \in \mathbb{N}$, existe una palabra $z \in L$, con $|z| \geq n$, $z = 0^{n^2}$
tal que para toda descomposición $z = uvw$. Si se verifica

- $|uv| \leq n$
- $|v| \geq 1$

entonces tenemos que $u = 0^k, v = 0^l, w = 0^{n^2-l-k}$, con $l \geq 1, l \leq n$.
$\exists i \in \mathbb{N}$, tal que $uv^iw \notin L$
Haciendo $i = 2, uv^2w = 0^k0^{2l}0^{n^2-l-k} = 0^{n^2+l}$
Como $(n+1)^2 - n^2 = n^2 + 2n + 1 - n^2 = 2n + 1 > n \geq l$, tenemos que $n^2 < n^2 + l < (n+1)^2$ y $uv^2w = 0^{n^2+l} \notin L$

$\{u \in \{0,1\}^* : u = u^{-1}\}$ no es regular

$\forall n \in \mathbb{N}$, existe una palabra $z \in L$, con $|z| \geq n$,
tal que para toda descomposición $z = uvw$. Si se verifica

- $|uv| \leq n$
- $|v| \geq 1$

entonces

$$\exists i \in \mathbb{N}, \text{ tal que } uv^iw \notin L$$

$\{u \in \{0,1\}^* : u = u^{-1}\}$ no es regular

$\forall n \in \mathbb{N}$, existe una palabra $z \in L$, con $|z| \geq n$, $\quad z = 0^n 1^n 0^n$
tal que para toda descomposición $z = uvw$. Si se verifica

- $|uv| \leq n$
- $|v| \geq 1$

entonces

$$\exists i \in \mathbb{N}, \text{ tal que } uv^i w \notin L$$

$\{u \in \{0,1\}^* : u = u^{-1}\}$ no es regular

$\forall n \in \mathbb{N}$, existe una palabra $z \in L$, con $|z| \geq n$, $\quad z = 0^n 1^n 0^n$
tal que para toda descomposición $z = uvw$. Si se verifica

- $|uv| \leq n$
- $|v| \geq 1$

entonces tenemos que $u = 0^k, v = 0^l, w = 0^{n-k-l}1^n 0^n$, con $l \geq 1$.

$$\exists i \in \mathbb{N}, \text{ tal que } uv^i w \notin L$$

$\{u \in \{0,1\}^* : u = u^{-1}\}$ no es regular

$\forall n \in \mathbb{N}$, existe una palabra $z \in L$, con $|z| \geq n$, $\quad z = 0^n 1^n 0^n$
tal que para toda descomposición $z = uvw$. Si se verifica

- $|uv| \leq n$
- $|v| \geq 1$

entonces tenemos que $u = 0^k, v = 0^l, w = 0^{n-k-l}1^n 0^n$, con $l \geq 1$.

$$\exists i \in \mathbb{N}, \text{ tal que } uv^i w \notin L$$

Haciendo $i = 2, uv^2 w = 0^k 0^{2l} 0^{n-k-l} 1^n 0^n = 0^{n+l} 1^n 0^n \notin L$

Contraejemplo

$L = \{a^l b^j c^k \ : \ (l = 0) \vee (j = k)\}$
Se verifica la condición del lema de bombeo para $n = 2$.
Si $z \in L$ y $|z| \geq 2$ entonces $z = a^l b^j c^k$ con $l = 0$ ó $j = k$.
Una descomposición de z se puede obtener de la siguiente forma:

- $u = \varepsilon$
- v es el primer símbolo de z
- w es z menos su primer símbolo

Contraej. Cont.

Caben dos posibilidades:

a) $l = 0$. En este caso $z = b^j c^k$ Se verifican las tres condiciones

exigidas en el lema de bombeo.

1. $|uv| = 1 \leq n = 2$
2. $|v| = 1 \geq 1$
3. Si $i \geq 0$ entonces $uv^i w$ sigue siendo una sucesión de b seguida de una sucesión de c y por tanto una palabra de L.

Contraej. Cont.

b) $l \geq 1$. En ese caso $z = a^l b^j c^j (l \geq 1)$, y también aquí se verifican las tres condiciones:

1. $|uv| = 1 \leq n = 2$
2. $|v| = 1 \geq 1$
3. Si $i \geq 0$ entonces $uv^i w$ sigue siendo una cacesión de a seguida de una sucesión de b y otra de c, en la que la cantidad de b es igual que la cantidad de c, y por tanto, una palabra de L.

Conjuntos Regulares: Operaciones

Ya conocemos las siguientes propiedades,

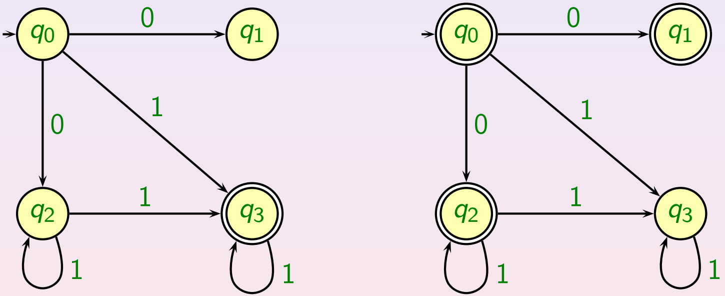
- *Unión:* Si L_1 y L_2 son conjuntos regulares, entonces $L_1 \cup L_2$ es regular.
- *Concatenación:* Si L_1 y L_2 son regulares, entonces $L_1 L_2$ es regular.
- *Clausura de Kleene:* Si L es regular, entonces L^* es regular.

Complementario

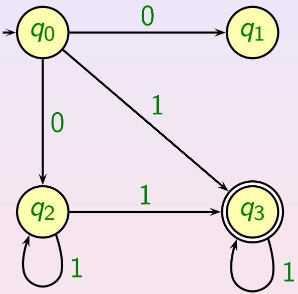
Si $L \subseteq A^*$ es un lenguaje regular entonces $\overline{L} = A^* - L$ es regular.

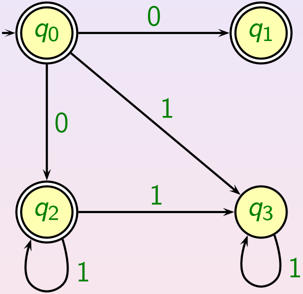
Basta con considerar que si $M = (Q, A, \delta, q_0, F)$ es un autómata finito determista que acepta el lenguaje L, entonces $M' = (Q, A, \delta, q_0, Q - F)$ acepta el lenguaje complementario $A^* - L$.

Esta operación no es válida en autómatas no deterministas



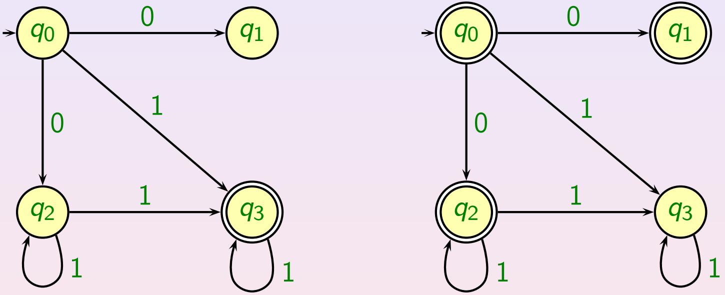
Esta operación no es válida en autómatas no deterministas





011 es aceptada en ambos autómatas

Esta operación no es válida en autómatas no deterministas



011 es aceptada en ambos autómatas
100 no es aceptada en ninguno de los autómatas

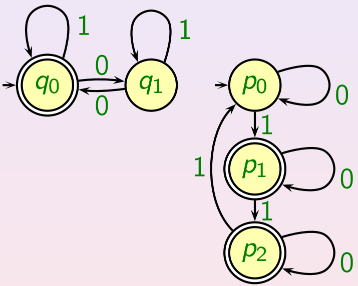
Intersección

Si L_1 y L_2 son dos lenguajes regulares sobre el alfabeto A, entonces $L_1 \cap L_2$ es regular.

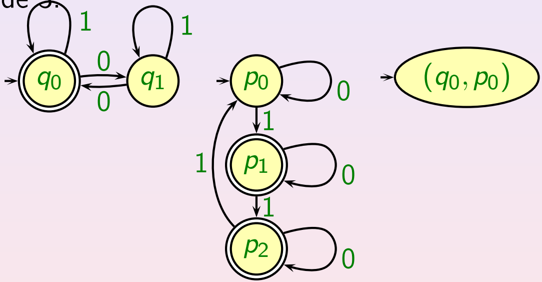
Es inmediato ya que $L_1 \cap L_2 = \overline{(\overline{L_1} \cup \overline{L_2})}$.
Existe también una demostración constructiva. Si $M_1 = (Q_1, A, \delta_1, q_0^1, F_1)$ es un autómata finito determinístico que acepta L_1, y $M_2 = (Q_2, A, \delta_2, q_0^2, F_2)$ es un autómata que acepta L_2, entonces
$M = (Q_1 \times Q_2, A, \delta, (q_0^1, q_0^2), F_1 \times F_2)$
donde $\delta((q_i, q_j), a) = (\delta_1(q_i, a), \delta_2(q_j, a))$, acepta el lenguaje $L_1 \cap L_2$.

Ejemplo

Construir el autómata que acepta las palabras con un número de ceros que es múltiplo de 2 y un número de unos que no sea múltiplo de 3.

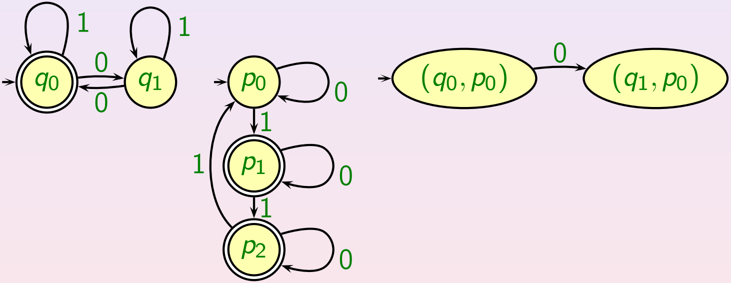


Construir el autómata que acepta las palabras con un número de ceros que es múltiplo de 2 y un número de unos que no sea múltiplo de 3.



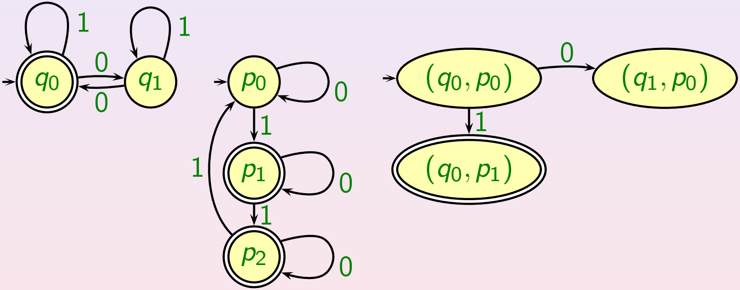
Ejemplo

Construir el autómata que acepta las palabras con un número de ceros que es múltiplo de 2 y un número de unos que no sea múltiplo de 3.



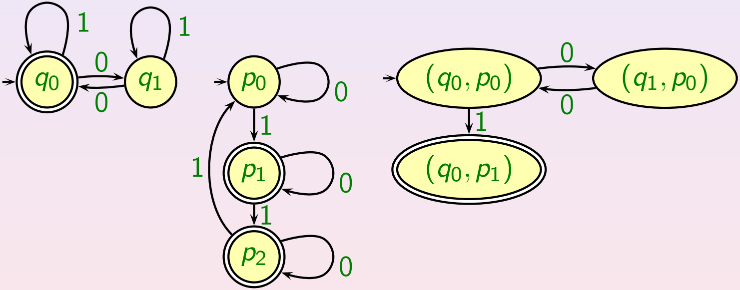
Ejemplo

Construir el autómata que acepta las palabras con un número de ceros que es múltiplo de 2 y un número de unos que no sea múltiplo de 3.



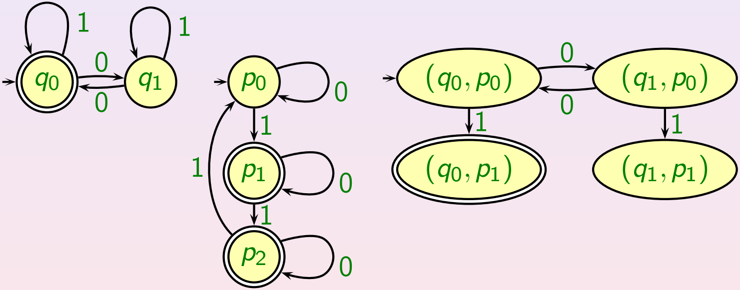
Ejemplo

Construir el autómata que acepta las palabras con un número de ceros que es múltiplo de 2 y un número de unos que no sea múltiplo de 3.



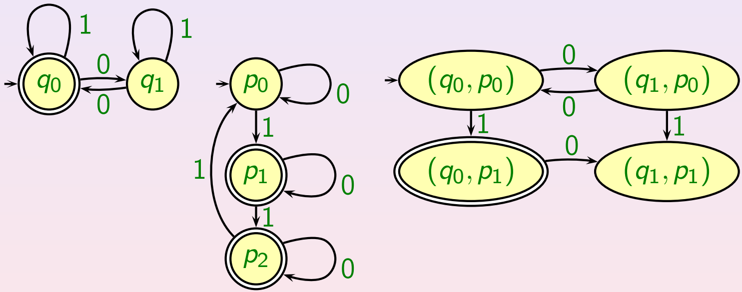
Ejemplo

Construir el autómata que acepta las palabras con un número de ceros que es múltiplo de 2 y un número de unos que no sea múltiplo de 3.

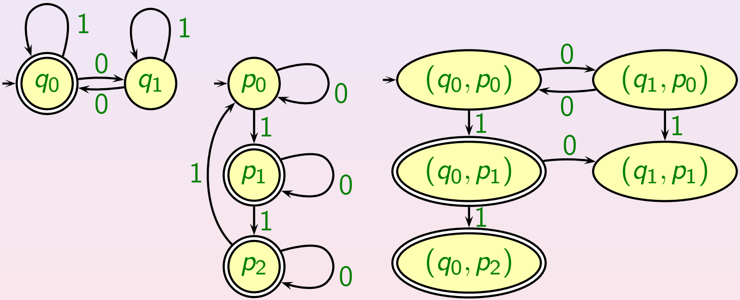


Ejemplo

Construir el autómata que acepta las palabras con un número de ceros que es múltiplo de 2 y un número de unos que no sea múltiplo de 3.

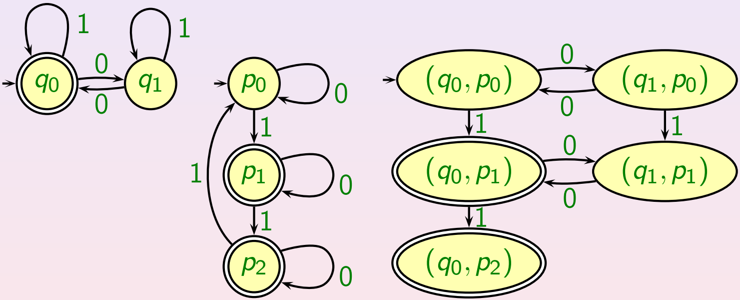


Construir el autómata que acepta las palabras con un número de ceros que es múltiplo de 2 y un número de unos que no sea múltiplo de 3.



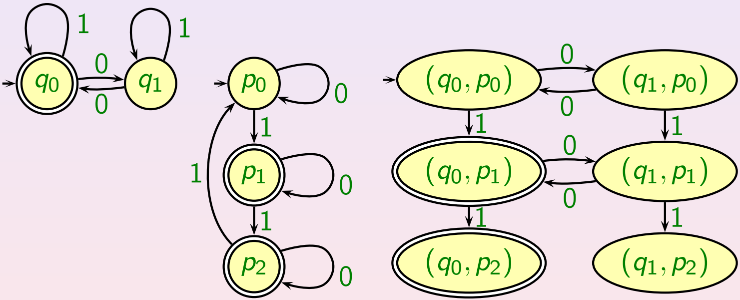
Ejemplo

Construir el autómata que acepta las palabras con un número de ceros que es múltiplo de 2 y un número de unos que no sea múltiplo de 3.



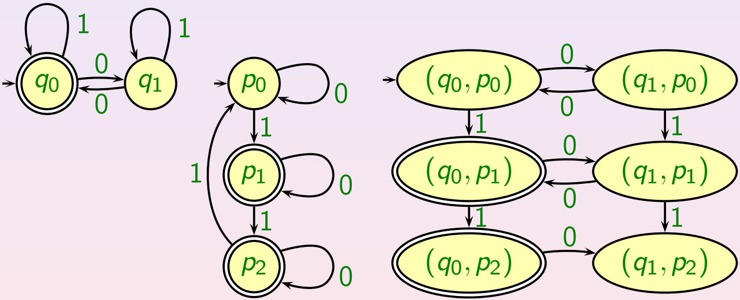
Ejemplo

Construir el autómata que acepta las palabras con un número de ceros que es múltiplo de 2 y un número de unos que no sea múltiplo de 3.

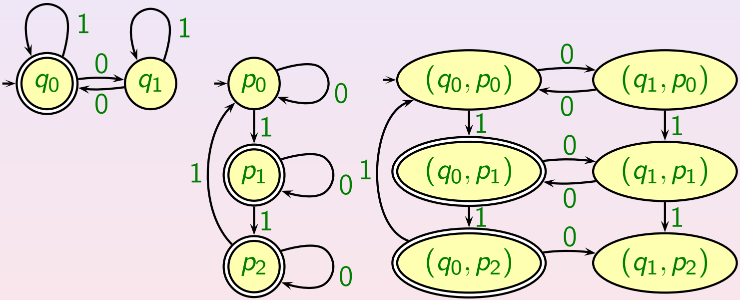


Ejemplo

Construir el autómata que acepta las palabras con un número de ceros que es múltiplo de 2 y un número de unos que no sea múltiplo de 3.

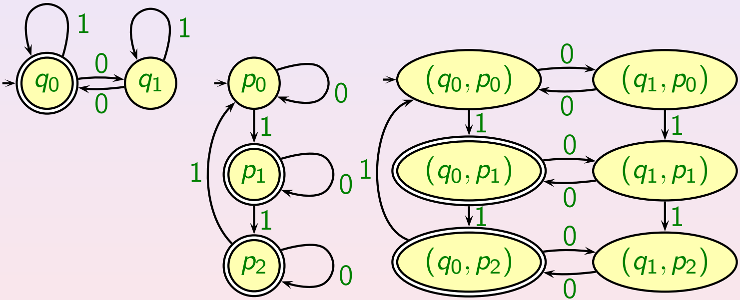


Construir el autómata que acepta las palabras con un número de ceros que es múltiplo de 2 y un número de unos que no sea múltiplo de 3.



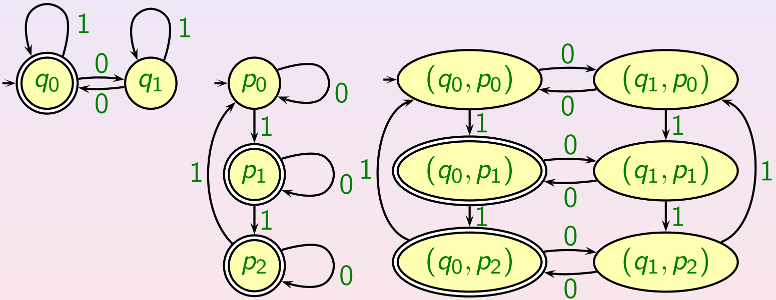
Ejemplo

Construir el autómata que acepta las palabras con un número de ceros que es múltiplo de 2 y un número de unos que no sea múltiplo de 3.

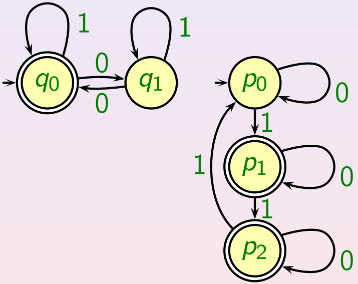


Ejemplo

Construir el autómata que acepta las palabras con un número de ceros que es múltiplo de 2 y un número de unos que no sea múltiplo de 3.

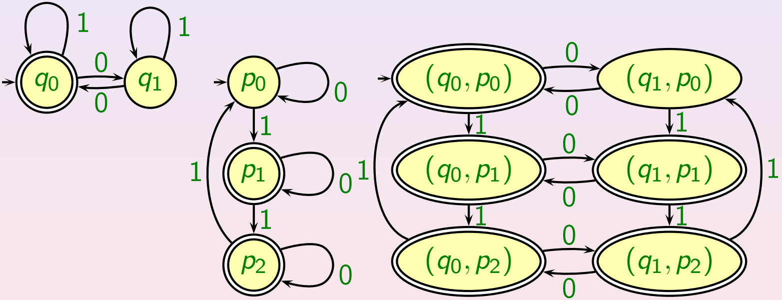


Construir el autómata que acepta las palabras con un número de ceros que es múltiplo de 2 o un número de unos que no sea múltiplo de 3.



Unión

Construir el autómata que acepta las palabras con un número de ceros que es múltiplo de 2 o un número de unos que no sea múltiplo de 3.



Homomorfismo

Si A y B son alfabetos y $f : A^* \longrightarrow B^*$ un homomorfismo entre ellos, **entonces** si $L \subseteq A^*$ es un lenguaje regular, $f(L) = \{u \in B^* : \exists v \in A^* \text{ verificando } f(v) = u\}$ es también un lenguaje regular.

Basta con comprobar que se puede conseguir una expresión regular para $f(L)$ partiendo de una expresión regular para L: Basta con substituir cada símbolo, a, de L, por la correspondiente palabra $f(a)$.

Ejemplo

Si $A = \{0,1,2,3,4,5,6,7,8,9\}$ y $B = \{0,1\}$ y f es el homomorfismo dado por

$f(0) = 0000, \quad f(1) = 0001, \quad f(2) = 0010, \quad f(3) = 0011$
$f(4) = 0100, \quad f(5) = 0101, \quad f(6) = 0110, \quad f(7) = 0111$
$f(8) = 1000, \quad f(9) = 1001$

Si $L \subseteq A^*$ es el lenguaje regular dado por la expresión regular $(1+2)^*9$,
$f(L)$ viene dado por la expresión regular $(0001+0010)^*1001$.

Aplicación

$L = \{a^i b^j c^k \,:\, (i = 0) \vee (j = k)\}$ no es regular.
La haremos por demostración al absurdo, considerando que L es regular y llegando a una contradicción.
Entonces $L' = L \cap L_1$ donde $L_1 = \{a^i b^j c^k \,:\, i > 0, j, k \geq 0\}$ es regular.
$L' = L \cap L_1 = \{a^i b^j c^k \,:\, (i > 0) \wedge (j = k)\}$

Demostración

$L' = \{a^i b^j c^k : (i > 0) \wedge (j = k)\}$ es regular.
Si L' es regular, y f es el homomorfismo entre $\{a, b, c\}^*$ y $\{0, 1\}^*$ dado por

$$f(a) = \varepsilon, \quad f(b) = 0, \quad f(c) = 1$$

entonces $f(L')$ es regular.
Pero $f(L') = \{0^k 1^k : k \geq 0\}$ y sabemos que este conjunto no es regular.
Por la tanto hemos encontrado una contradicción y la hipótesis de que L es regular es falsa.

Homomorfismo Inverso

Si A y B son alfabetos y $f : A^* \longrightarrow B^*$ es un homomorfismo, entonces si $L \subseteq B^*$ es un conjunto regular, también lo es $f^{-1}(L) = \{u \in A : f(u) \in L\}$.

Supongamos que $M = (Q, B, \delta, q_0, F)$ que acepta el lenguaje L
Entonces el autómata $\overline{M} = (Q, A, \overline{\delta}, q_0, F)$ donde

$$\overline{\delta}(q, a) = \delta'(q, f(a)),$$

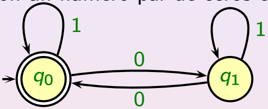
acepta el lenguaje $f^{-1}(L)$.

Ejemplo

En el homomorfismo f entre $A = \{0,1,2,3,4,5,6,7,8,9\}$ y $B = \{0,1\}$ dado por:

$f(0) = 0000$, $\quad f(1) = 0001$, $\quad f(2) = 0010$, $\quad f(3) = 0011$, $\quad f(4) = 0100$,
$f(5) = 0101$, $\quad f(6) = 0110$, $\quad f(7) = 0111$, $\quad f(8) = 1000$, $\quad f(9) = 1001$

El conjunto de las palabras con un número par de ceros es:

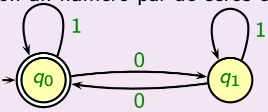


Ejemplo

En el homomorfismo f entre $A = \{0,1,2,3,4,5,6,7,8,9\}$ y $B = \{0,1\}$ dado por:

$f(0) = 0000$, $f(1) = 0001$, $f(2) = 0010$, $f(3) = 0011$, $f(4) = 0100$,
$f(5) = 0101$, $f(6) = 0110$, $f(7) = 0111$, $f(8) = 1000$, $f(9) = 1001$

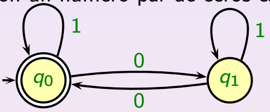
El conjunto de las palabras con un número par de ceros es:



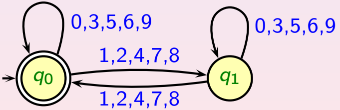
y el conjunto de palabras $u \in A^*$ tales que $f(u)$ tiene un número par de ceros es regular con autómata:

Ejemplo

En el homomorfismo f entre $A = \{0,1,2,3,4,5,6,7,8,9\}$ y $B = \{0,1\}$ dado por:
$f(0) = 0000$, $f(1) = 0001$, $f(2) = 0010$, $f(3) = 0011$, $f(4) = 0100$,
$f(5) = 0101$, $f(6) = 0110$, $f(7) = 0111$, $f(8) = 1000$, $f(9) = 1001$
El conjunto de las palabras con un número par de ceros es:



y el conjunto de palabras $u \in A^*$ tales que $f(u)$ tiene un número par de ceros es regular con autómata:



Aplicación

Si $A = B = \{0, 1\}$ y f es el homomorfismo dado por

$$f(0) = 00, \qquad f(1) = 11$$

entonces el lenguaje $L = \{0^{2k}1^{2k} : k \geq 0\}$ no es regular, porque si lo fuese su imagen inversa, $f^{-1}(L) = \{0^k 1^k : k \geq 0\}$ sería también regular y no lo es.

Cociente

Teorema

Si R es un conjunto regular y L un lenguaje culquiera, entonces el cociente de lenguajes $R/L = \{u : \exists v \in L \text{ verificando } uv \in R\}$ es un conjunto regular.

Sea $M = (Q, A, \delta, q_0, F)$ un autómata finito determinístico que acepta el lenguaje R.
Entonces R/L es aceptado por el autómata

$$M' = (Q, A, \delta, q_0, F')$$

donde $F' = \{q \in Q : \exists y \in L \text{ tal que } \delta'(q, y) \in F\}$

Teorema

Si R es un conjunto regular y L un lenguaje culquiera, entonces el cociente de lenguajes $R/L = \{u : \exists v \in L \text{ verificando } uv \in R\}$ es un conjunto regular.

Sea $M = (Q, A, \delta, q_0, F)$ un autómata finito determinístico que acepta el lenguaje R.
Entonces R/L es aceptado por el autómata

$$M' = (Q, A, \delta, q_0, F')$$

donde $F' = \{q \in Q : \exists y \in L \text{ tal que } \delta'(q, y) \in F\}$

¡¡Esta demostración no es constructiva!!

El lenguaje aceptado por un autómata es vacío:

El lenguaje aceptado por un autómata es vacío:
Basta eliminar estados inaccesibles (mediante un recorrido por el grafo a partir del estado inicial) y comprobar si quedan estados finales.

El lenguaje aceptado por un autómata es vacío:
Basta eliminar estados inaccesibles (mediante un recorrido por el grafo a partir del estado inicial) y comprobar si quedan estados finales.
El lenguaje aceptado por un autómata es finito o infinito:

El lenguaje aceptado por un autómata es vacío:
Basta eliminar estados inaccesibles (mediante un recorrido por el grafo a partir del estado inicial) y comprobar si quedan estados finales.

El lenguaje aceptado por un autómata es finito o infinito:
Se suponen eliminados los estados inaccesibles y se eliminan los estados de error o estados desde los que no se pueden llegar a estado finales.
Se puede hacer recorriendo el grafo en sentido contrario a los arcos y empezando en los estados finales.
Se comprueba si en el grafo resultante quedan ciclos.

Algoritmos: Igualdad

Dados dos autómatas finitos M_1 y M_2 comprobar si aceptan el mismo lenguaje:

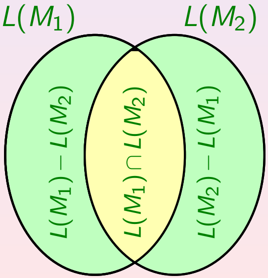
Algoritmos: Igualdad

Dados dos autómatas finitos M_1 y M_2 comprobar si aceptan el mismo lenguaje:

Basta con construir el autómata que acepta el lenguaje

$$(L(M_1) - L(M_2)) \cup (L(M_2) - L(M_1)) =$$
$$(L(M_1) \cap \overline{L(M_2)}) \cup (L(M_2) \cap \overline{L(M_1)})$$

Después se comprueba si el lenguaje aceptado por este autómata es vacío.



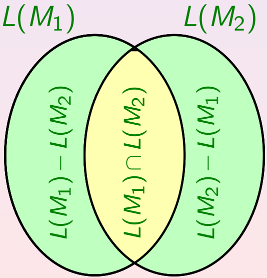
Algoritmos: Igualdad

Dados dos autómatas finitos M_1 y M_2 comprobar si aceptan el mismo lenguaje:

Basta con construir el autómata que acepta el lenguaje

$$(L(M_1) - L(M_2)) \cup (L(M_2) - L(M_1)) =$$
$$(L(M_1) \cap \overline{L(M_2)}) \cup (L(M_2) \cap \overline{L(M_1)})$$

Después se comprueba si el lenguaje aceptado por este autómata es vacío.



La mejor forma de hacer este autómata es con el autómata producto, haciendo finales las parejas de estados en las que un estado es final y el otro no.

Autómata Minimal

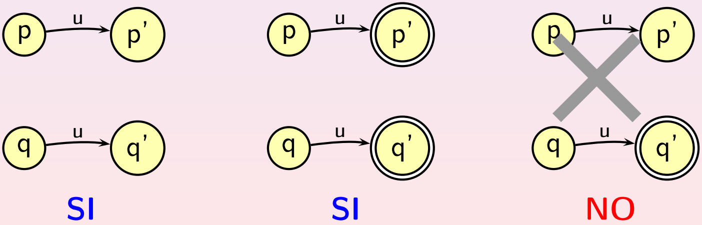
Un autómata finito determinista M se dice minimal si no exite otro autómata con menos estados que él y que acepte el mismo lenguaje.

- La primera condición para que un autómata sea minimal es que no tenga estados inaccesibles.
- Vamos a ver primero una condición para que un autómata sin estados indistinguibles sea minimal: que no tenga estados indistinguibles.
- A continuación veremos un algoritmo que, dado un autómata M calcula un autómata minimal que acepta el mismo lenguaje.

Un concepto básico para minimizar autómatas es el de estados indistinguibles.

Si $M = (Q, A, \delta, q_0, F)$ es un autómata finito determinista y p, q son dos estados de Q, decimos que p y q son indistinguibles si y solo si se cumple que

$$\forall u \in A^*, \quad (\delta'(p,u) \in F \Leftrightarrow \delta'(q,u) \in F)$$



Un autómata sin estados inaccesibles es **minimal** si y solo si no tiene estados indistinguibles.

Demostraremos que un autómata no es minimal $\Leftrightarrow$ tiene estados indistinguibles.

$\Leftarrow$ Si p y q son estados indistinguibles de M, entonces el autómata que se obiene a partir de M uniendo p y q en el mismo estado (las transiciones del nuevo estado son las de uno cualquiera de ellos) acepta el mismo lenguaje y tiene menos estados.
Por tanto, M no sería minimal.

Un autómata sin estados inaccesibles es **minimal** si y solo si no tiene estados indistinguibles.

Demostraremos que un autómata no es minimal $\Leftrightarrow$ tiene estados indistinguibles.

$\Rightarrow$ Si M no es minimal existe otro autómata M' con menos estados que acepta el mismo lenguaje.
Es fácil comprobar que exiten dos palabras $u, v \in A^*$ tales que al leerlas en M se llega a dos estados distintos p y q; y al leerlas en M' se llega al mismo estado p'.
Se puede comprobar que p y q tienen que ser indestinguibles, ya que M y M' aceptan el mismo lenguaje.

Identificación de Estados Indistinguibles

Dos estados p, q son distinguibles de nivel n si y solo si existe una palabra $u \in A^*$ de longitud menor o igual que n tal que en el conjunto $\{\delta'(p,u), \delta'(q,u)\}$ hay un estado final y otro no final.

Dos estados p, q son distinguibles de nivel n si y solo si existe una palabra $u \in A^*$ de longitud menor o igual que n tal que en el conjunto $\{\delta'(p,u), \delta'(q,u)\}$ hay un estado final y otro no final.
Una pareja de estados es distinguible si y solo si es distinguible a nivel n para algún $n \in \mathbb{N}$.

Dos estados p, q son distinguibles de nivel n si y solo si existe una palabra $u \in A^*$ de longitud menor o igual que n tal que en el conjunto $\{\delta'(p,u), \delta'(q,u)\}$ hay un estado final y otro no final.

Una pareja de estados es distinguible si y solo si es distinguible a nivel n para algún $n \in \mathbb{N}$.

Las parejas distinguibles a nivel 0 son las formadas por un estado final y otro no final.

Dos estados p, q son distinguibles de nivel n si y solo si existe una palabra $u \in A^*$ de longitud menor o igual que n tal que en el conjunto $\{\delta'(p,u), \delta'(q,u)\}$ hay un estado final y otro no final.

Una pareja de estados es distinguible si y solo si es distinguible a nivel n para algún $n \in \mathbb{N}$.

Las parejas distinguibles a nivel 0 son las formadas por un estado final y otro no final.

Las parejas $\{p, q\}$ distinguibles a nivel $n+1$ son las que son distinguibles a nivel n más aquellas tales que existe un $a \in A$ tal que $\{\delta(p,a), \delta(q,a)\}$ es distinguible a nivel n.

Identificación de Estados Indistinguibles

Dos estados p,q son distinguibles de nivel n si y solo si existe una palabra $u \in A^*$ de longitud menor o igual que n tal que en el conjunto $\{\delta'(p,u), \delta'(q,u)\}$ hay un estado final y otro no final.

Una pareja de estados es distinguible si y solo si es distinguible a nivel n para algún $n \in \mathbb{N}$.

Las parejas distinguibles a nivel 0 son las formadas por un estado final y otro no final.

Las parejas $\{p,q\}$ distinguibles a nivel $n+1$ son las que son distinguibles a nivel n más aquellas tales que existe un $a \in A$ tal que $\{\delta(p,a), \delta(q,a)\}$ es distinguible a nivel n.

Si para un n las parejas distinguibles a nivel n son las mismas que las distinguibles a nivel $n+1$ entonces para todo $m \geq n$, las parejas distinguibles a nivel m son las mismas que las distinguibles a nivel n.

Identificación de Estados Indistinguibles

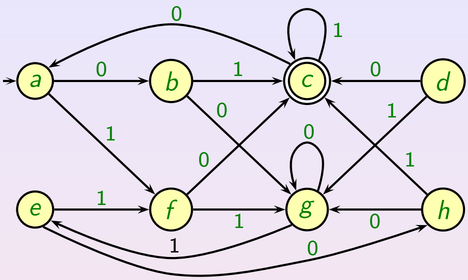
Pareja de estados $\rightarrow$ *variable booleana* $+$ *lista de parejas de estados*

Identificación de Estados Indistinguibles

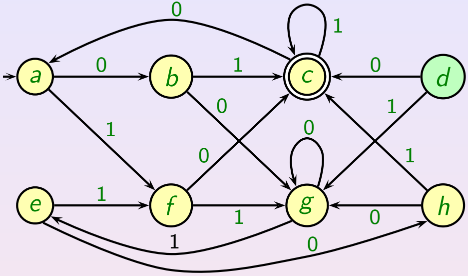
Pareja de estados $\rightarrow$ variable booleana + lista de parejas de estados

1. Eliminar estados inaccesibles.
2. Para cada pareja de estados accesibles $\{q_i, q_j\}$
   3. Si uno de ellos es final y el otro no, hacer la variable booleana asociada igual a true.
4. Para cada pareja de estados accesibles $\{q_i, q_j\}$
   5. Para cada símbolo a del alfabeto de entrada
      6. Calcular los estados q_k y q_l a los que evoluciona el autómata desde q_i y q_j leyendo a
      7. Si $q_k \neq q_l$ entonces
         8. Si la $\{q_k, q_l\}$ está marcada entonces se marca la pareja $\{q_i, q_j\}$ y recursivamente todas las parejas en la lista asociada.
      9. Si la pareja $\{q_k, q_l\}$ no está marcada, se añade la pareja $\{q_i, q_j\}$ a la lista asociada a la pareja $\{q_k, q_l\}$

Ejemplo

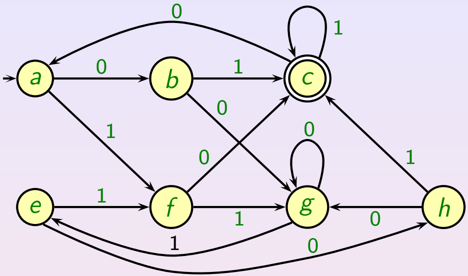


Ejemplo



El estado d es inaccesible

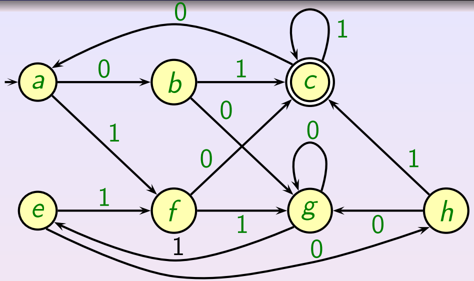
Ejemplo



El estado d es inaccesible

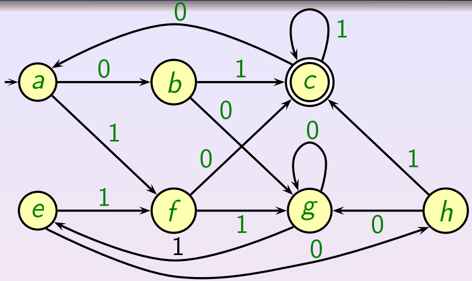
y se elimina

Ejemplo



a b c e f g

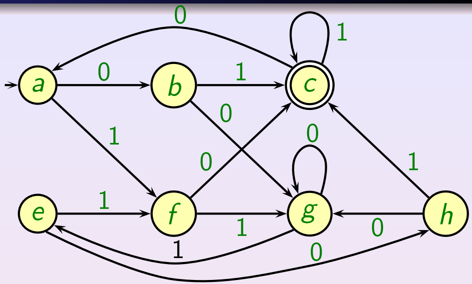
Ejemplo



b						
c						
e						
f						
g						
h						
	a	b	c	e	f	g

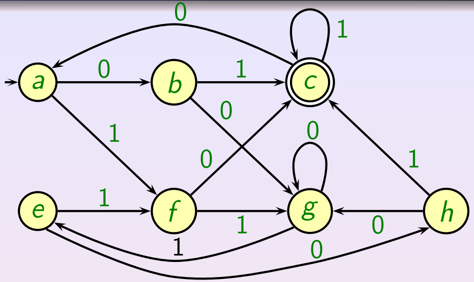
Ejemplo

b						
c						
e						
f						
g						
h						
	a	b	c	e	f	g



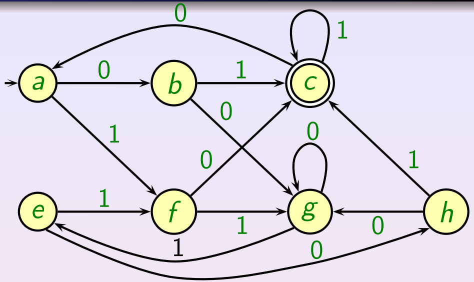
Ejemplo

	a	b	c	e	f	g
b						
c	X	X				
e			X			
f			X			
g			X			
h			X			



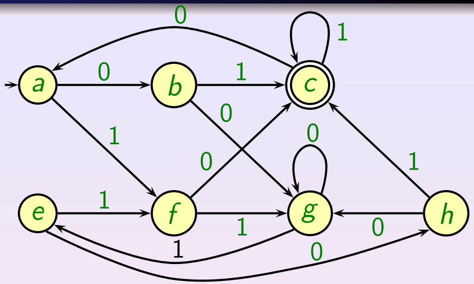
Ejemplo

	a	b	c	e	f	g
b						
c	X	X				
e			X			
f			X			
g			X			
h			X			



Ejemplo

	a	b	c	e	f	g
b						
c	X	X				
e			X			
f			X			
g		●	X			
h			X			

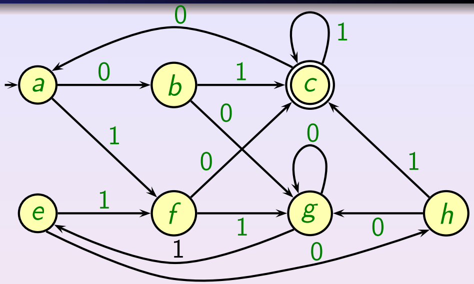


g

b

Ejemplo

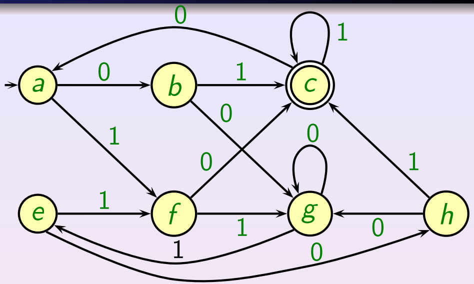
	a	b	c	e	f	g
b						
c	X	X				
e			X			
f			X			
g		(h,a)	X			
h			X			



	0	1
h		c
a		f

Ejemplo

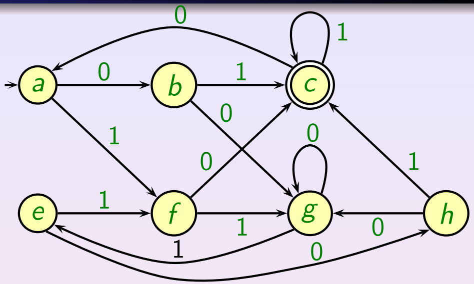
	a	b	c	e	f	g
b						
c	X	X				
e			X			
f			X			
g		(h,a)	X			
h	X		X			



	0	1
h	g	
b	g	

Ejemplo

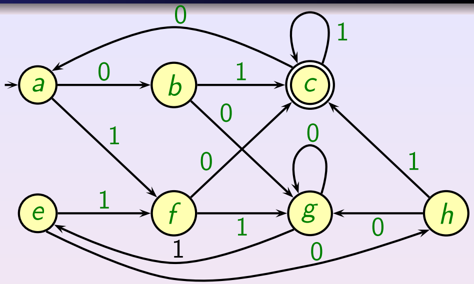
	a	b	c	e	f	g
b						
c	X	X				
e			X			
f			X			
g		(h,a)	X			
h	X		X			



	0	1
h		c
b		c

Ejemplo

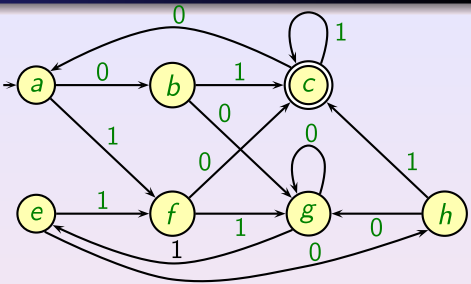
	a	b	c	e	f	g
b						
c	X	X				
e			X			
f			X			
g		(h,a)	X			
h	X		X			



	0	1
h	g	
e	h	

Ejemplo

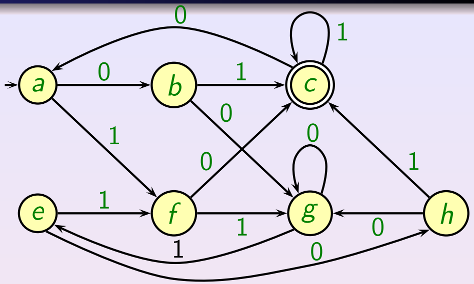
	a	b	c	e	f	g
b						
c	X	X				
e			X			
f			X			
g		(h,a)	X			
h	X		X			(h,e)



	0	1
h		c
e		f

Ejemplo

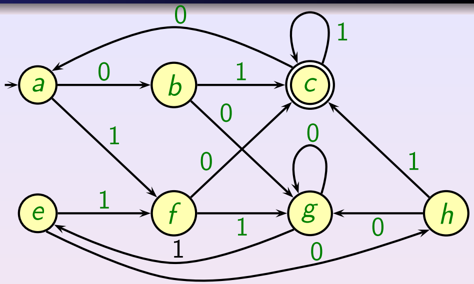
	a	b	c	e	f	g
b						
c	X	X				
e			X			
f			X			
g		(h,a)	X			
h	X		X	X		(h,e)



	0	1
h	g	
f	c	

Ejemplo

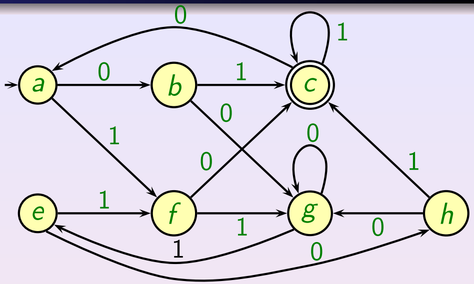
	a	b	c	e	f	g
b						
c	X	X				
e			X			
f			X			
g		(h,a)	X			
h	X		X	X	X	(h,e)



	0	1
h	g	
g	g	

Ejemplo

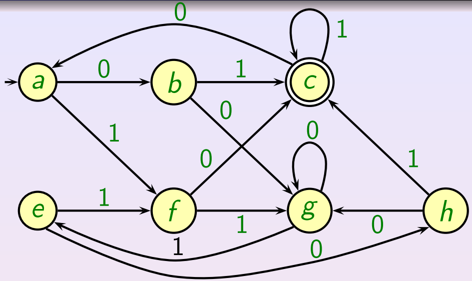
	a	b	c	e	f	g
b						
c	X	X				
e			X			
f			X			
g		(h,a)	X			
h	X		X	X	X	(h,e)



	0	1
h		c
g		e

Ejemplo

	a	b	c	e	f	g
b						
c	X	X				
e			X			
f			X			
g		(h,a)	X			
h	X		X	X	X	X

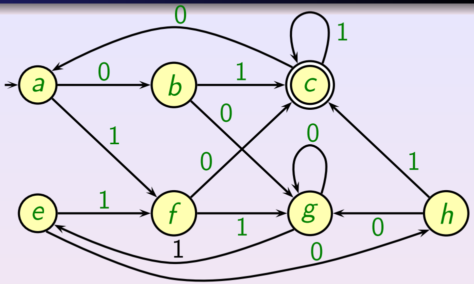


(h,e)

	0	1

Ejemplo

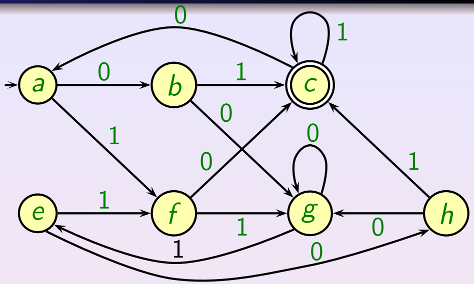
	a	b	c	e	f	g
b						
c	X	X				
e			X			
f			X			
g		(h,a)	X			
h	X		X	X	X	X



	0	1
g	g	
a	b	

Ejemplo

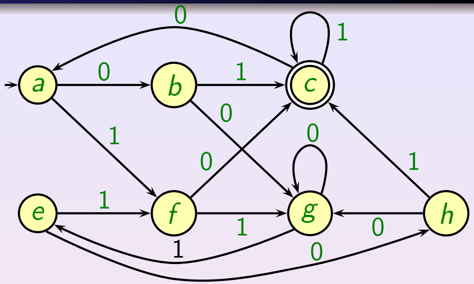
	a	b	c	e	f	g
b						
c	X	X				
e			X			
f			X			
g		(h,a) (g,a)	X			
h	X		X	X	X	X



	0	1
g		e
a		f

Ejemplo

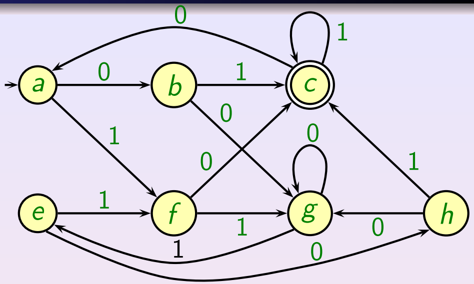
	a	b	c	e	f	g
b						
c	✗	✗				
e			✗			
f			✗	(g,a)		
g		(h,a) (g,a)	✗			
h	✗		✗	✗	✗	✗



	0	1
g	g	
b	g	

Ejemplo

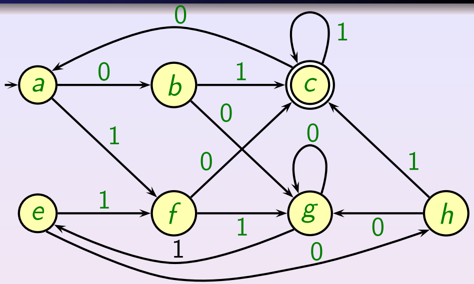
	a	b	c	e	f	g
b						
c	X	X				
e			X			
f			X	(g,a)		
g		(h,a) (g,a)	X			
h	X		X	X	X	X



	0	1
g		e
b		c

Ejemplo

	a	b	c	e	f	g
b						
c	X	X				
e			X			
f			X	(g,a)		
g		X	X			
h	X		X	X	X	X

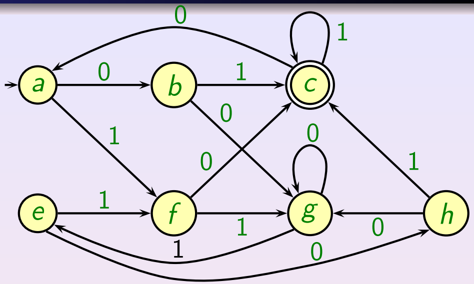


(h,a) **(g,a)**

	0	1

Ejemplo

	a	b	c	e	f	g
b						
c	✗	✗				
e			✗			
f			✗	(g,a)		
g		✗	✗			
h	✗		✗	✗	✗	✗

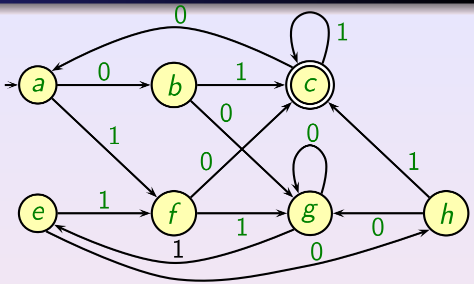


(g,a)

	0	1

Ejemplo

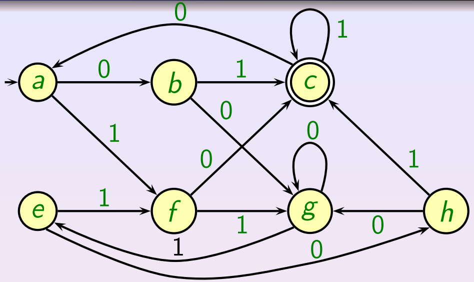
	a	b	c	e	f	g
b						
c	X	X				
e			X			
f			X	(g,a)		
g	X	X	X			
h	X		X	X	X	X



	0	1
g	g	
e	h	

Ejemplo

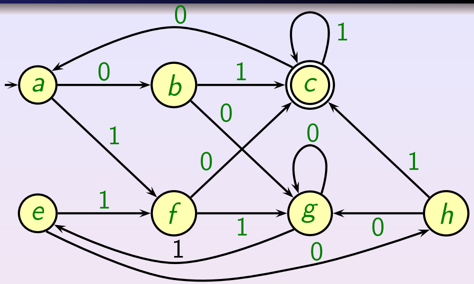
	a	b	c	e	f	g
b						
c	X	X				
e			X			
f			X	(g,a)		
g	X	X	X	X		
h	X		X	X	X	X



	0	1
g	g	
f	c	

Ejemplo

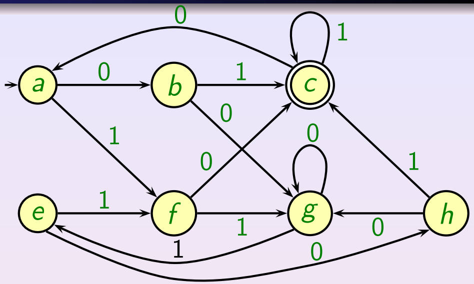
b						
c	X	X				
e			X			
f			X	(g,a)		
g	X	X	X	X	X	
h	X		X	X	X	X
	a	b	c	e	f	g



	0	1
f	c	
a	b	

Ejemplo

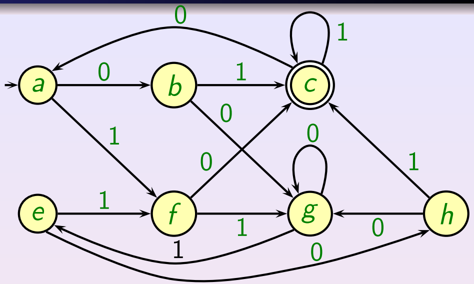
	a	b	c	e	f	g
b						
c	X	X				
e			X			
f	X		X	(g,a)		
g	X	X	X	X	X	
h	X		X	X	X	X



	0	1
f	c	
b	g	

Ejemplo

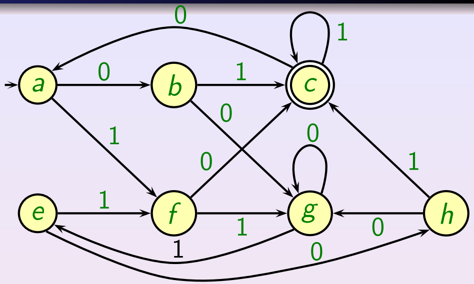
	a	b	c	e	f	g
b						
c	X	X				
e			X			
f	X	X	X	(g,a)		
g	X	X	X	X	X	
h	X		X	X	X	X



	0	1
f	c	
e	h	

Ejemplo

	a	b	c	e	f	g
b						
c	X	X				
e			X			
f	X	X	X	X		
g	X	X	X	X	X	
h	X		X	X	X	X

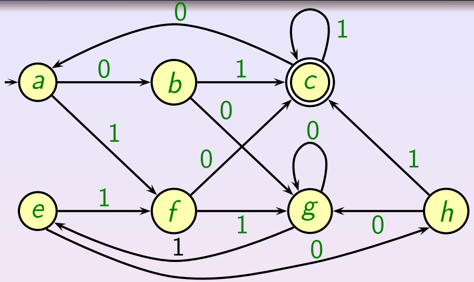


(g,a)

	0	1

Ejemplo

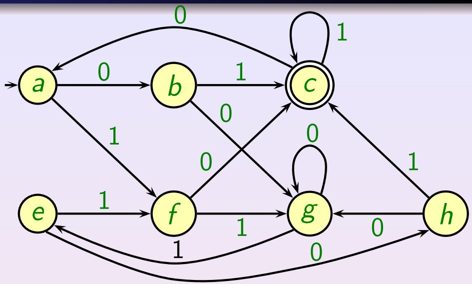
	a	b	c	e	f	g
b						
c	X	X				
e			X			
f	X	X	X	X		
g	X	X	X	X	X	
h	X	•	X	X	X	X



	0	1
e	h	
a	b	

Ejemplo

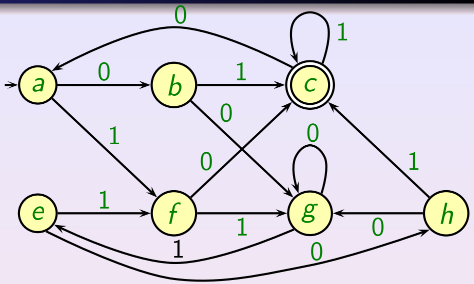
	a	b	c	e	f	g
b						
c	X	X				
e			X			
f	X	X	X	X		
g	X	X	X	X	X	
h	X	(e,a)	X	X	X	X



	0	1
e		f
a		f

Ejemplo

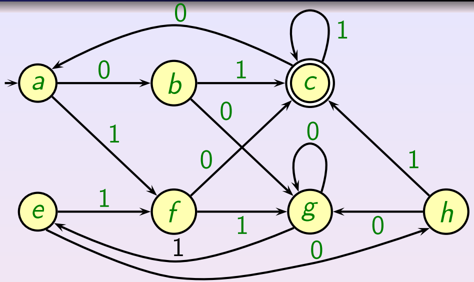
	a	b	c	e	f	g
b						
c	X	X				
e			X			
f	X	X	X	X		
g	X	X	X	X	X	
h	X	(e,a)	X	X	X	X



	0	1
e	h	
b	g	

Ejemplo

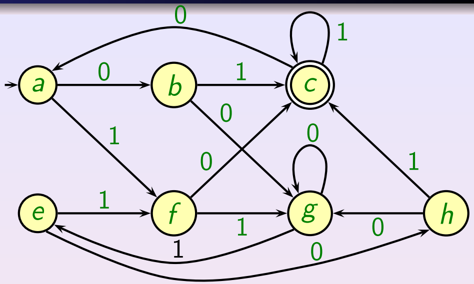
	a	b	c	e	f	g
b						
c	X	X				
e		X	X			
f	X	X	X	X		
g	X	X	X	X	X	
h	X	(e,a)	X	X	X	X



	0	1
b	g	
a	b	

Ejemplo

	a	b	c	e	f	g
b	X					
c	X	X				
e		X	X			
f	X	X	X	X		
g	X	X	X	X	X	
h	X	(e,a)	X	X	X	X



$b \equiv h, \quad a \equiv e$

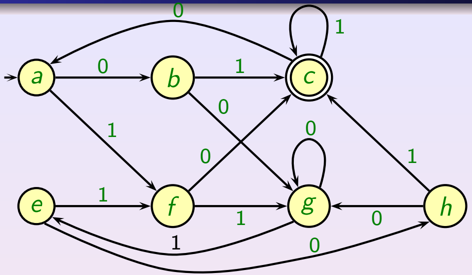
	0	1

Construcción del Autómata Minimal

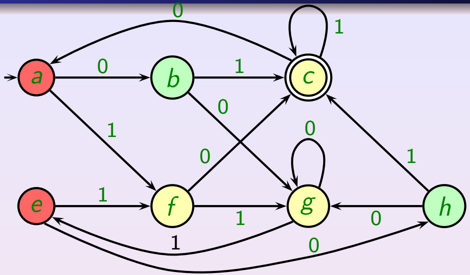
El autómata minimal se construye identificando los estados indistinguibles.
Si el autómata original es $M = (Q, A, \delta, q_0, F)$,
R es la relación de equivalencia de indistinguibilidad entre estados
$[q]$ la clase de equivalencia asociada al estado q,
El nuevo autómata, $M_m = (Q_m, A, \delta_m, q_0^m, F_m)$ tiene los siguientes elementos,

- $Q_m = \{[q] : q \text{ es accesible desde } q_0\}$
- $F_m = \{[q] : q \in F\}$
- $\delta_m([q], a) = [\delta(q, a)]$
- $q_0^m = [q_0]$

Ejemplo: Autómata Minimal



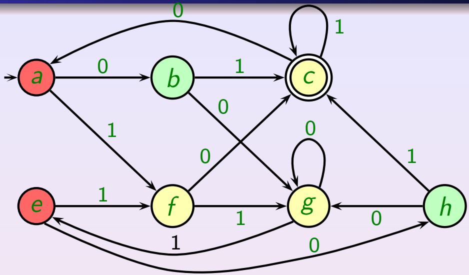
Ejemplo: Autómata Minimal



$$b \equiv h$$

$$a \equiv e$$

Ejemplo: Autómata Minimal



$b \equiv h$

$a \equiv e$