Teoría de la Computación y Lenguajes Formales

Lenguajes Regulares - Propiedades

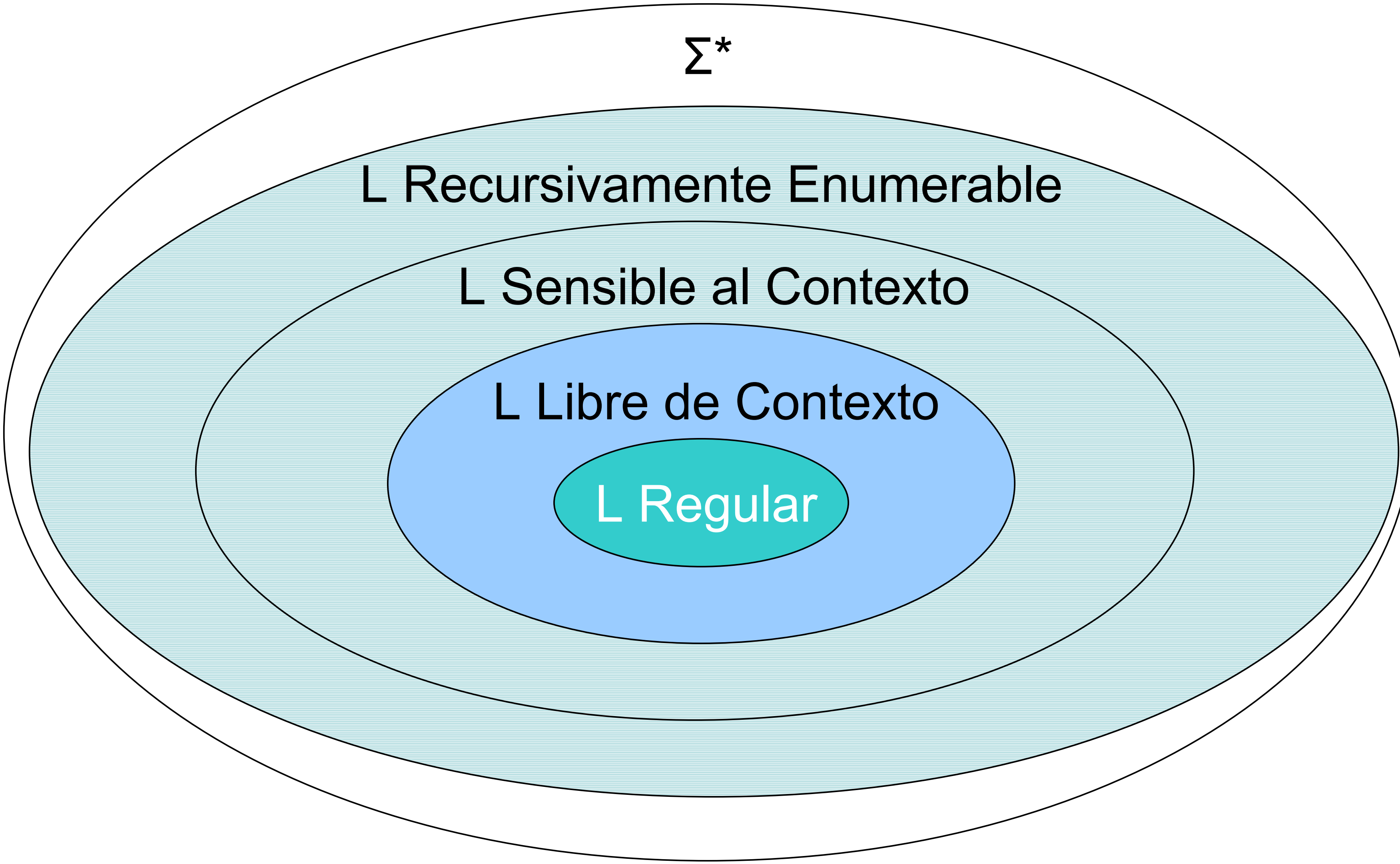
Prof. Hilda Y. Contreras
Departamento de Computación
hyelitza@ula.ve
hildac.teoriadelacomputacion@gmail.com

UNIVERSIDAD DE LOS ANDES

Objetivo

- Lenguajes Regulares (GR, AF y ER)
- Propiedad de regularidad = lema del bombeo para LR
- Propiedades cerradas de los LR
- Algoritmos de decisión de los LR

Lenguajes Regulares



¿Lenguajes Regulares?

Ejemplos de lenguajes:

- Lenguajes de programación
- ADN, ARN
- HTML, XML, OWL, etc.
- Lenguas humanas
- Código Morse
- Música
- Latex ......

Lenguaje Regular

- Jerarquía de Chomsky (Tipo 3)

Tipo	Lenguaje	Máquina	Gramática
0	Recursivamente enumerable	Máquina de Turing	Sin restricciones
1	Dependiente del Contexto	Autómata linealmente acotado	*Gramática dependiente del contexto* $\alpha A\beta \Rightarrow \alpha\gamma\beta$
2	Independiente del Contexto	Autómata de Pila	*Gramática libre de contexto* $A \Rightarrow \gamma$
3	**Lenguaje Regular**	**Autómata finito**	***Gramática Regular*** $A \Rightarrow aB$ $A \Rightarrow a$

Lema del Bombeo

Enunciado por Y. Bar-Hillel, M. Perles, E. Shamir en 1961. Su objetivo es demostrar que un Lenguaje L **NO es LR**

Para demostrar que un Lenguaje L si es LR ➔ Obtener un AF, ER o una GR

Importancia: identificar el tipo de lenguaje para poder usar las herramientas adecuadas para procesarlo.

Lema del Bombeo

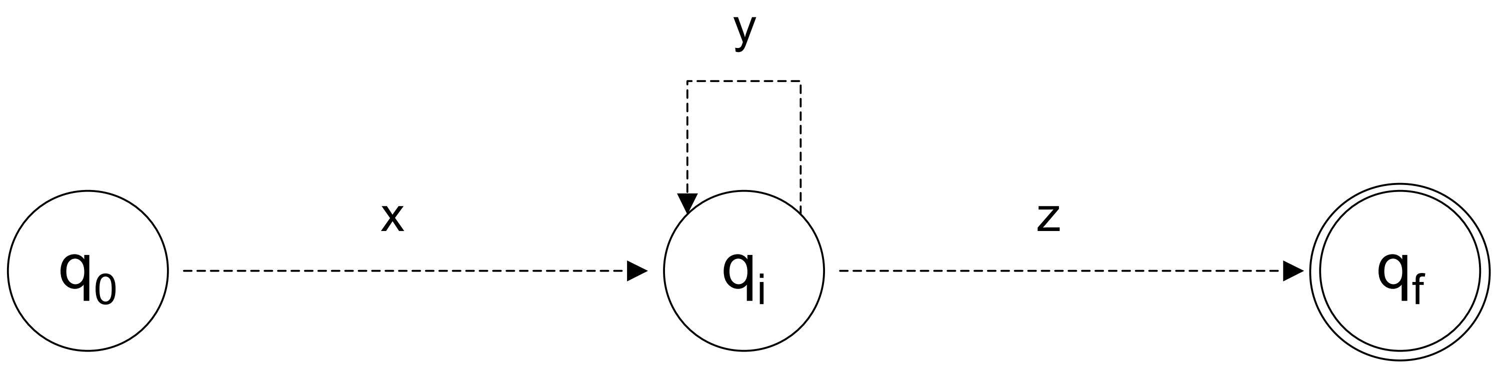
Sea **L** un lenguaje regular sobre **Σ**. Existe un número natural **m** (dependiente del lenguaje L) tal que para todo **w** en **L** se cumple que $|\mathbf{w}| \geq \mathbf{m}$, existen **x**,**y**,**z** en Σ^* tales que $\mathbf{w} = \mathbf{xyz}$ y donde:

1. $|\mathbf{xy}| \geq \mathbf{m}$
2. $|\mathbf{y}| \geq 1$
3. Para todo $\mathbf{i} \geq 0$, $\mathbf{xy^iz}$ en **L**

Condición **necesaria** para que un lenguaje sea regular: todos LR tiene esta propiedad.

Lema del Bombeo

El lema dice que un AFD con un número "finito" de estados (m), genera un lenguaje "infinito" a través un ciclo.



$\delta^\wedge(q_0,x) = q_i$, $\boldsymbol{\delta^\wedge(q_i,y) = q_i}$, $\boldsymbol{\delta^\wedge(q_i,y^i) = q_i}$, $\delta^\wedge(q_i,z) = q_f$

$\rightarrow \delta^\wedge(q_0,xyz) = q_f$, $w = xyz$, $\delta^\wedge(q_0,w) = q_f$

Lema del Bombeo

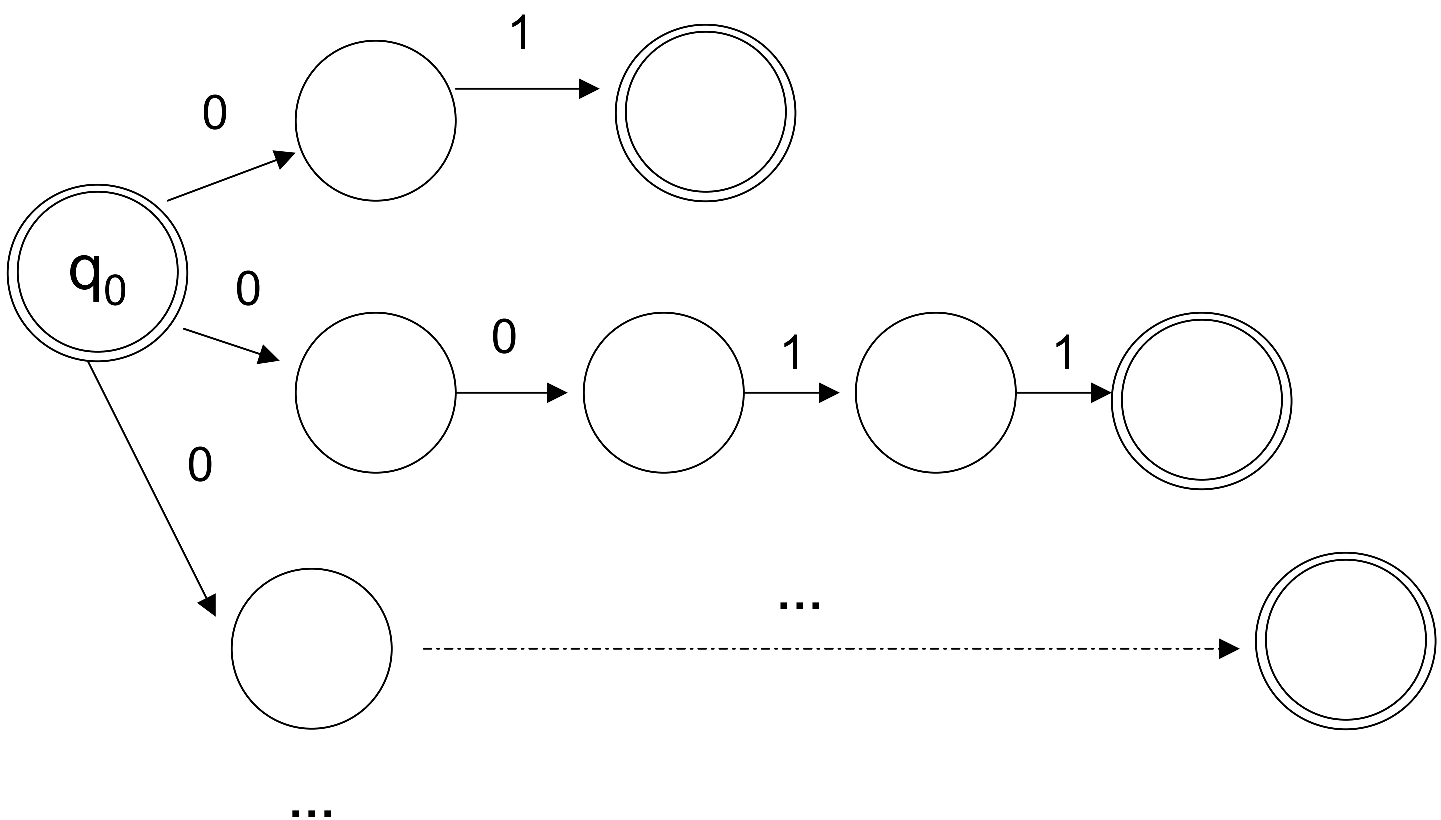
Método de uso (reducción al absurdo):

1. Asumir que L es un lenguaje regular
2. Tomar **m** como el valor de la constante del lema de bombeo
3. Escoger una palabra **w** en L tal que |**w**| ≥ **m**
4. Considerar todas las posibles factorizaciones de **w** (xyz según #2 y #1 del lema)
5. Mostrar que, para todas las factorizaciones posibles, puede encontrarse un valor de **i** tal que $\mathbf{xy^iz}$ no esta L (contradicción #3)

Lema del Bombeo

Por ejemplo: $\Sigma = \{0,1\}$ $L = \{ 0^j 1^i \mid i=j,\ i,j \geq 0 \}$

¿lenguaje regular?



Lema del Bombeo

Por ejemplo: $L = \{ 0^j 1^i \mid i = j, i,j \geq 0 \}$

1. Asumir que L es un lenguaje regular
2. Escoger una palabra **w** en L tal que |**w**| ≥ **m**
   w $= 0^m 1^m$
3. Considerar todas las posibles factorizaciones de **w**
4. Mostrar que, para todas las factorizaciones posibles, puede encontrarse un valor de **i** tal que $\mathbf{xy^iz}$ no esta L

Usar JFLAP para ver demostración: http://www.jflap.com/

Lema del Bombeo

El lema del bombeo sólo puede usarse para mostrar que un lenguaje **NO** es regular (reducción al absurdo), pero no puede usarse para mostrar que un lenguaje si es regular (#3 Para todo $\mathbf{i} \geq 0$, $\mathbf{xy^iz}$ en **L**).

Es una condición necesaria pero no suficiente: Hay Lenguajes libres del contexto (LR ≤ LLC) que satisfacen el lema del bombeo para LR!

Propiedades de clausura

Propiedades cerradas: operaciones aplicadas en un conjunto cuyo resultado pertenece al mismo conjunto

p.e. Suma de enteros 1 **+** 4 = 5

División de enteros 1 **/** 4 = 0,25

Importancia: componer varios lenguajes y obtener otro lenguaje más complejo dentro del mismo tipo

Propiedades de clausura

Permiten resolver lenguajes complejos:

- Cadenas binarias con un número par de ceros **y** con un número impar de unos.
- Cadenas binarias que **no** contienen la subcadena 001.
- Cadenas binarias que tienen un número de ceros múltiplo de tres **menos** la cadena 000.

Propiedades de clausura

Propiedad	LR
U (unión)	S
concatenación	S
Kleene-clausura	S
∩ (intersección)	S
complemento	S

Propiedades de clausura

Unión, concatenación y clausura de Kleene:
Si r y s son expresiones regulares denotando los lenguajes R y S entonces definimos las siguientes operaciones:

Unión: (r + s) es una expresión regular que denota el lenguaje R U S

Concatenación: (rs) es una expresión regular que denota el lenguaje RS

Clausura: r* es una expresión regular que denota el lenguaje R*.

Propiedades de clausura

Intersección: Sean L_1 y L_2, entonces existen dos autómatas A_1 y A_2 tales que $L_1 = L(A_1)$ y $L_2 = L(A_2)$ donde:

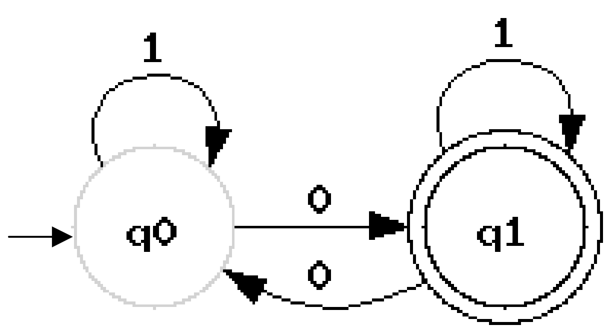
$A_i = (Q_i, \Sigma_i, \delta_i, q_i, F_i)$, i = 1,2

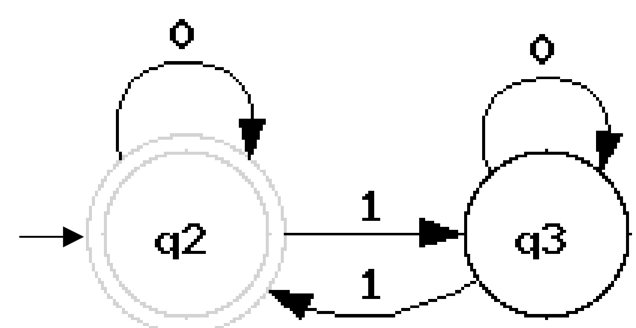
Construimos A = (Q, Σ, δ, q0, F) donde:

- $Q = Q_1 \times Q_2$
- $q_0 = [q_1, q_2]$
- $F = F_1 \times F_2$
- $\delta([p_1, p_2], a) = [\delta_1(p_1, a), \delta_2(p_2, a)]$, para todo p_1 en Q_1, Para todo p_2 en Q_2, Para todo a en Σ

Propiedades de clausura

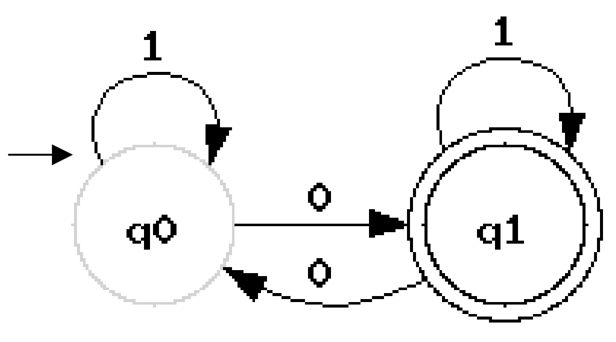
Por ejemplo: el lenguaje de todas las cadenas binarias que tienen un número impar de 0s **y** número par de 1s

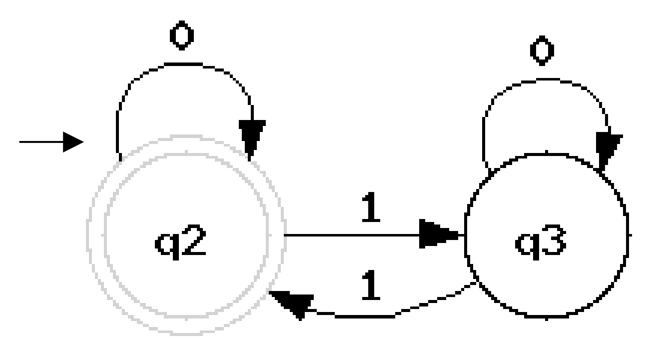




Propiedades de clausura

Producto δ	0	1
[q0,q2]	**[q1, q2]**	**[q0, q3]**
[q1,q2]	**[q0, q2]**	**[q1, q3]**
[q0,q3]	**[q1, q3]**	**[q0, q2]**
[q1,q3]	**[q0, q3]**	**[q1, q2]**

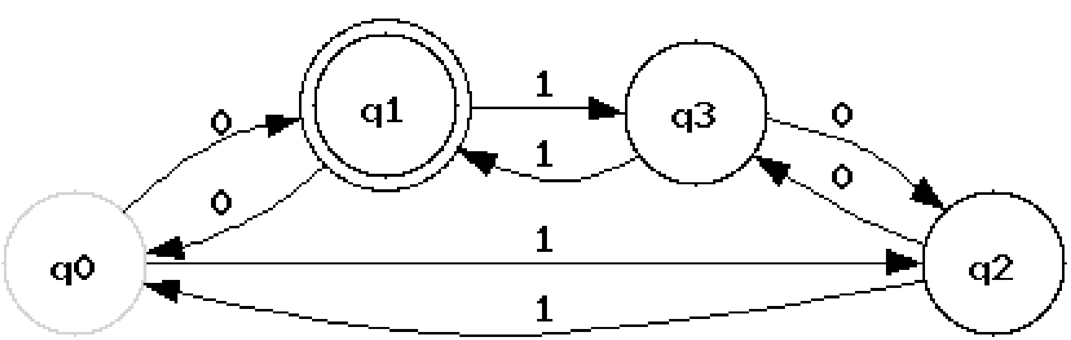




Propiedades de clausura

El lenguaje de todas las cadenas binarias que tienen un número impar de 0s **y** un número par de 1s: Intersección (Construcción de producto)

http://obi-wan.esi.uclm.es:4080/apps/selfa/



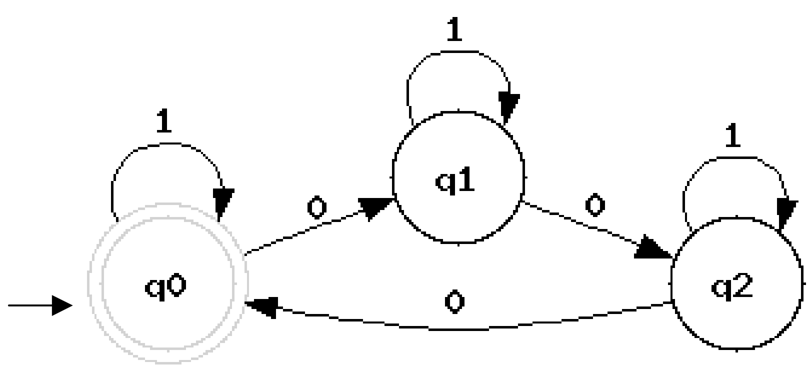
Propiedades de clausura

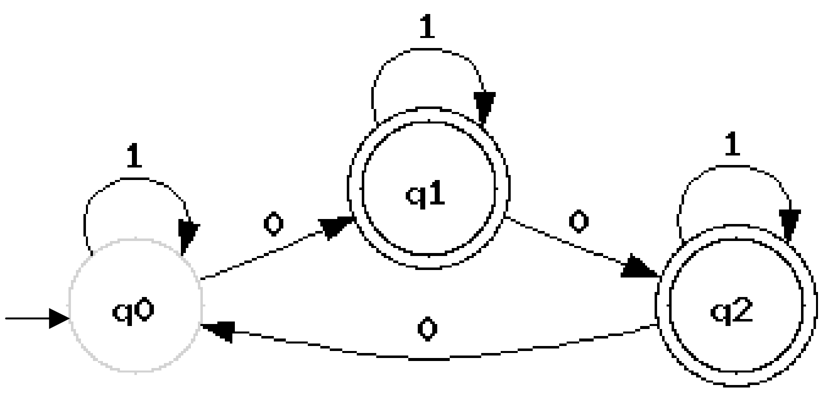
Complemento:

- Sea L_1 un lenguaje regular entonces existe un autómata completo A tal que $L_1 = L(A)$ donde
  $A = (Q,\Sigma,\delta,q_0,F)$
- El autómata $A^c = (Q, \Sigma,\delta, q_0,Q - F)$

Propiedades de clausura

Por ejemplo: el lenguaje de todas las cadenas binarias que tienen NO tienen un número de 0s múltiplo de 3.





Propiedades de clausura

Propiedad	**LR**
Reflejo	S
Morfismo	S
Morfismo^{-1}	S
Diferencia	S

Propiedades de clausura

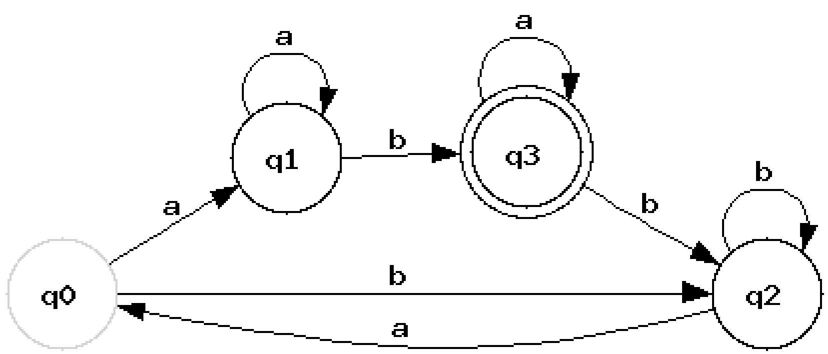
Reflejo o inverso: Sean L existe un autómata A tal que $L = L(A)$, $A = (Q,\Sigma,\delta,q,F)$

Construimos $A^R = (Q_R, \Sigma, \delta_R, q_{0R}, \{q_f\})$ donde:

- $Q_R = Q_1$
- Si $|F| > 1$ puede modificarse el autómata para que posea un único estado final.
- Construimos $A = (Q_R; \Sigma ; \delta_R; qf ; q0)$ donde:
- Si $\delta(p,a) = q$, entonces $\delta_R(q; a) = p$

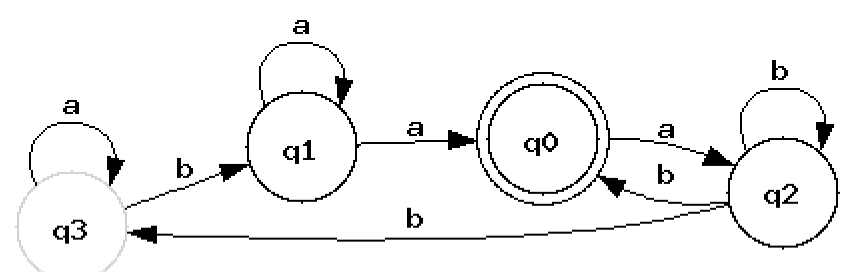
Propiedades de clausura

Reflejo o inverso: p.e



(ab*b)*+(ab*ba*ba*a)*

(bb*a)*aa*ba*



Propiedades de clausura

Morfismo:

Sea $h : \Sigma \rightarrow \Delta^*$, Existe L en Σ^* y un autómata A tal que $L = L(A)$ y donde $A = (Q,\Sigma,\delta,q_0,F)$

Construimos $A' = (Q,\Sigma,\delta',q_0,F)$ donde:

- $\delta'(p,a) = \delta(p,h(a))$ si $\delta(p,h(a)) \neq \Phi$
- $\delta'(p,a) = \Phi$ en cualquier otro caso

Propiedades de clausura

Diferencia: Sean L_1 y L_2, lenguajes regulares, entonces existen dos autómatas completos A_1 y A_2 tales que $L_1 = L(A_1)$ y $L_2 = L(A_2)$ donde:

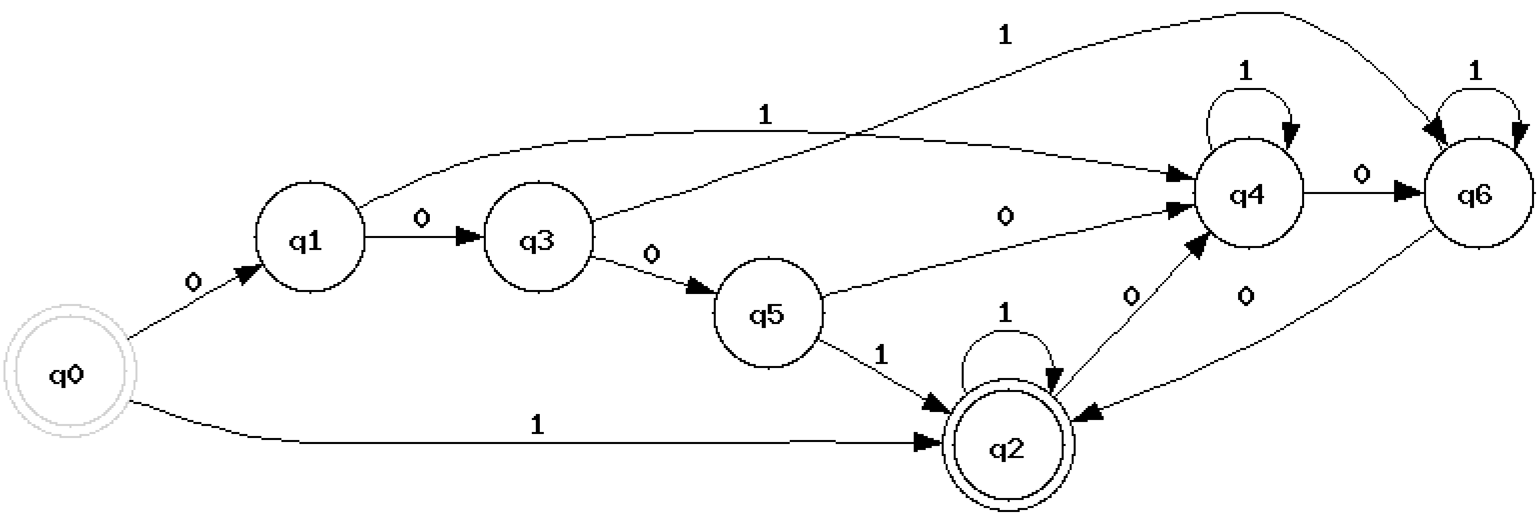
$A_i = (Q_i,\Sigma_i,\delta_i,q_i,F_i)$, $i = 1,2$

$L_1 - L_2 = L_1 \sqcap L_2{}^c$

Propiedades de clausura

Diferencia: $L_1 - L_2 = L_1 \sqcap L_2{}^c$

p.e. Reconozca el lenguaje formado por las cadenas binarias con un número de ceros múltiplos de 3 menos la cadena 000.



Algoritmos de decisión

Problemas	LR
Equidad	D
Inclusión	D
Membresía	D
Vacuidad	D
Finitud	D

Algoritmos de decisión

Algoritmos de decisión: procedimientos aplicados a toda instancia del **problema**, efectivo y que siempre termina dando un resultado

Notas:

- Algoritmos de decisión ≠ Problema de decisión
- **Problema** aplicados a lenguajes regulares
- El lenguaje común es infinito

Algoritmos de decisión

Finitud: L_1 es finito ?.

Por el lema del bombeo ➔ si existen ciclos es infinito.

AF: Algoritmo para hallar ciclos en un digrafo desde el estado inicial a algún final.

ER: ¿si existe clausura de Kleene?

Algoritmos de decisión

Vacuidad: L_1 es vacio?.

AF: por accesabilidad de un digrafo, existe un camino desde el estado inicial a algunos de los estados finales.

ER: propiedades algebraicas de ER.

- R + S → R + Φ = R , R y S deben ser Φ
- RS → RΦ = Φ, R o S deben ser Φ
- R* → nunca es Φ porque la clausura contiene a la cadena vacía λ

Algoritmos de decisión

Membresía: Dado w en Σ^*, si w esta en L_1?.

AF: recorrido de un digrafo o simular un AFD

ER: convertir a un AFN-λ, convertir a un AFD y simular

Algoritmos de decisión

Equidad: L_1 y L_2 son iguales?

Existe un algoritmo para determinar si dos AFD aceptan el mismo lenguaje.

Demostración.- Sean M_1 y M_2 dos AFD, entonces de forma algorítmica se puede construir al AFD M que acepte el lenguaje:

$L(M) = (L(M_1) \Pi L(M_2)^c)$ U $(L(M_1)^c \Pi L(M_2))$

Entonces si $L(M) \neq \Phi$ (vacuidad).

Algoritmos de decisión

Equidad: L_1 y L_2 son iguales?

Teorema de Myhill-Nerode. Minimización de Autómatas

Sea L es subconjunto de A* un lenguaje arbitrario. Asociado a este lenguaje L se puede definir una relación de equivalencia RL en el conjunto A, de la siguiente forma:

Si x, y estan en A*, entonces (xRLy) si y solo si (Para todo z en A*; (xz en L, si i solo si, yz en L))

Esta relación de equivalencia dividirá el conjunto A* en clases de equivalencia. El número de clases de equivalencia se llama índice de la relación.

Algoritmos de decisión

Equidad: L_1 y L_2 son iguales?

Teorema de Myhill-Nerode. Minimización de Autómatas

También se puede definir una relación de equivalencia, RM, en A* asociada a AFD $M = (Q;A;\delta;q_0;F)$

Si u,v en A*, entonces uRMv si y solo si $(\delta(q_0;u) = \delta(q_0;v))$

Esta relación de equivalencia divide también el lenguaje A en clases de equivalencia.

Algoritmos de decisión

Teorema de Myhill-Nerode. Minimización de Autómatas

- Si L es subconjunto de A* entonces:
  1. L es aceptado por un autómata finito
  2. L es la unión de algunas de las clases de equivalencia de una relación de equivalencia en A* de índice finito que sea invariante por la derecha.
  3. La relación de equivalencia RL es de índice finito.
- Si L es un conjunto regular y RL la relación de equivalencia asociada, entonces el autómata construido en el teorema anterior es minimal y único salvo isomorfismos.

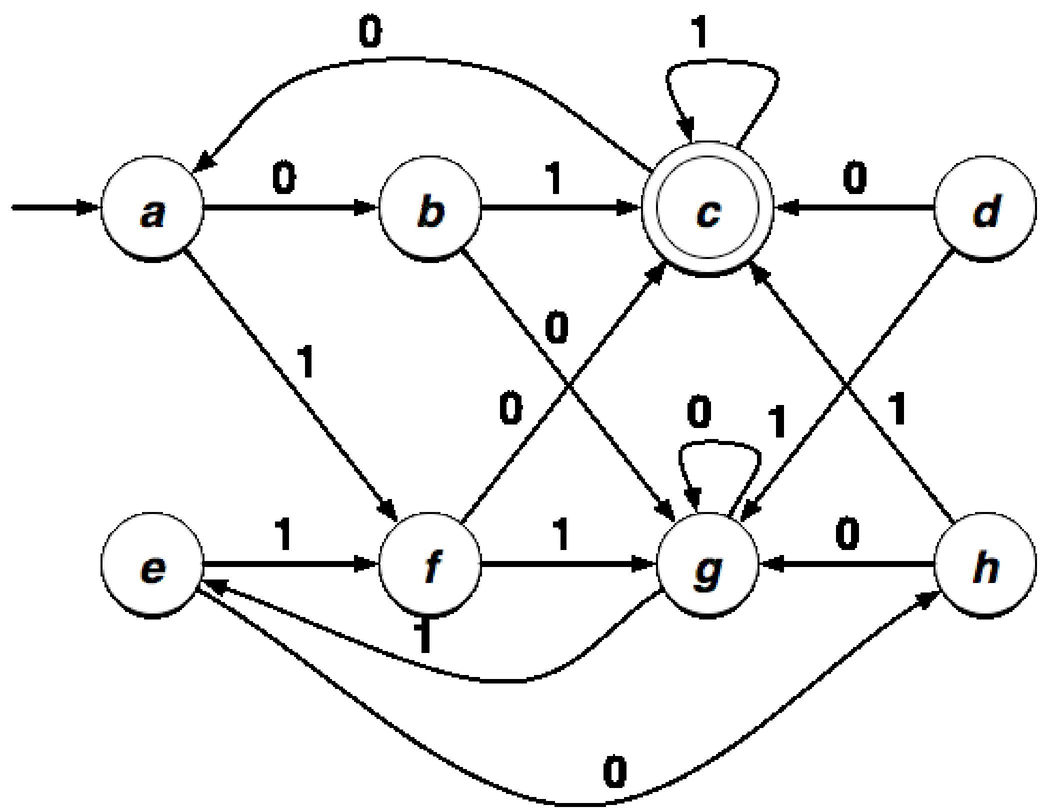
Minimización de Autómatas

Existe un método simple para encontrar el AFD con número mínimo de estados *M'* equivalente a un AFD $M = (Q, A, \delta, q_0, F)$. Sea ≡ la relación de equivalencia de los estados de *M* tal que $p \equiv q$ si y sólo si para cada entrada *x*, $\delta\hat{}(p,x)$ es un estado de aceptación si y sólo si $\delta\hat{}(q,x)$ es un estado de aceptación

Si $p \equiv q$, decimos que p es equivalente a q. Decimos que p es **distinguible** de q si existe un x tal que $\delta\hat{}(p, x)$ en F y $\delta\hat{}(q, x)$ no está en F, o viceversa

Ejemplo

- Sea M el siguiente autómata



Ejemplo

- Se tiene que construir una tabla con una entrada para cada par de estados. Se coloca una *X* en la tabla cada vez que un par de estados son distinguibles. Inicialmente se coloca una *X* en cada entrada correspondiente a un estado final y un estado no final. En el ejemplo, Q - F = {a,b,d,e,f,g,h} y F = {c} colocamos una *X* en las entradas (*a,c*), (*b,c*), (*c,d*), (*c,e*), (*c,f*), (*c,g*) y (*c,h*).
- Para cada par de estados p y q que no se han identificado como distinguibles, consideramos el par de estados (r,s), $r = \delta(p, a)$ y $s = \delta(q, a)$ para cada entrada *a.*

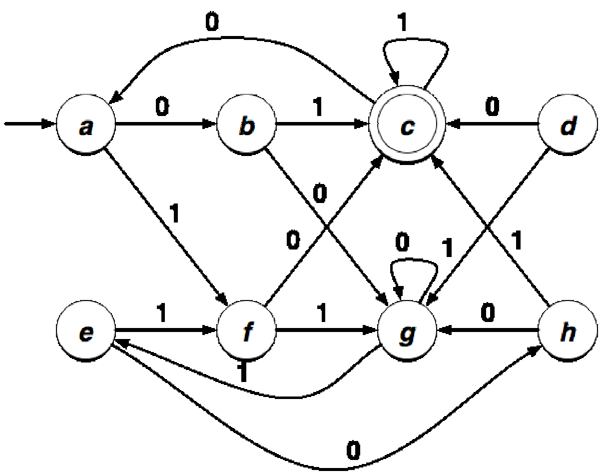
Ejemplo

- Si se demuestra que los estados *s* y *r* son distinguibles para alguna cadena *x* entonces *p* y *q* son distinguibles para cualquier cadena *ax*.
- Así si la entrada (*r, s*) en la tabla tiene una *X*, se coloca una *X* en la entrada (*p, q*).
- Si la entrada (*r, s*) no tiene *X*, entonces el par (*p, q*) es colocado en una lista asociada con la entrada (*r, s*).
- Continuando se tiene que si la entrada (*r, s*) recibe una *X* entonces cada par en la lista asociada con la entrada (*r, s*) también recibe una *X*.

Ejemplo

- En el ejemplo, colocamos una X en la entrada (r, s), porque la entrada ($\delta(b,1)$, $\delta(a,1)$) = (c,f) ya tiene una X. Similarmente, la entrada (a,d) recibe una *X*. Ahora consideramos la entrada (a, e) que con la entrada 0 va a dar el par (b, h), así (a, e) es colocado en la lista asociada con (b,h). Observe que con la entrada 1, *a* y *e* van al mismo estado *f* y por lo tanto no hay cadena con 1 que pueda distinguir *a* de *e*.

Ejemplo: Tabla de estados distinguibles

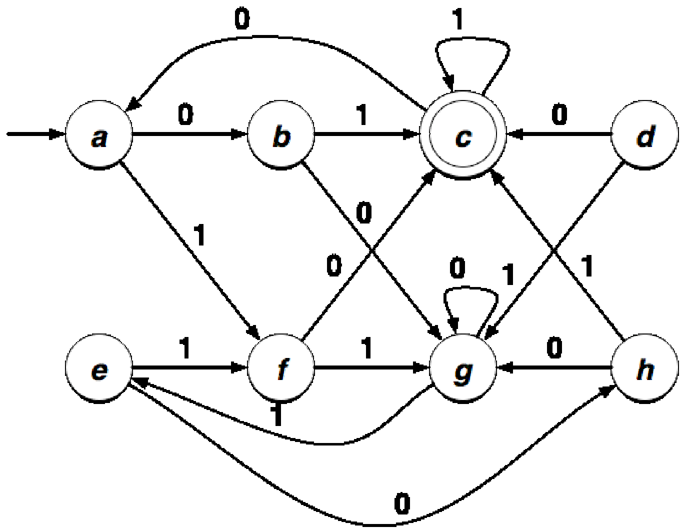


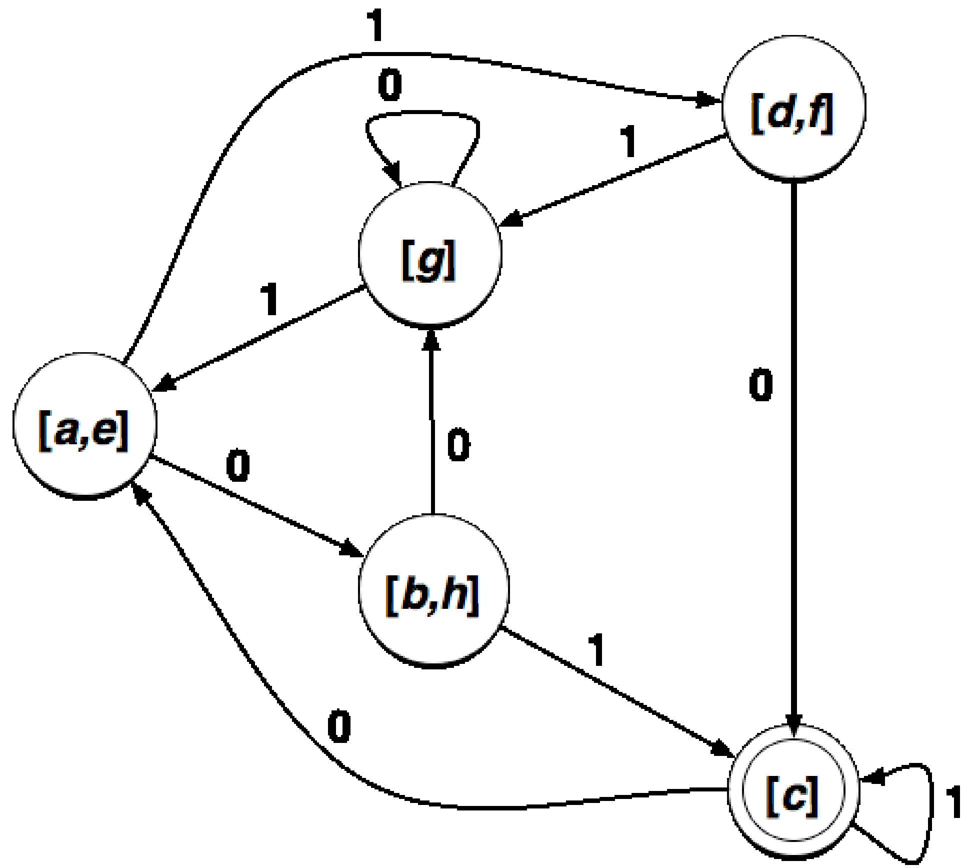
	a	b	c	d	e	f	g
b	X						
c	X	X					
d	X	X	X				
e		X	X	X			
f	X	X	X		X		
g	X	X	X	X	X	X	
h	X		X	X	X	X	X

Ejemplo

- Se concluye que los estados equivalentes son $a \equiv e$, $b \equiv h$, y $d \equiv f$, y el autómata con número de estados es el siguiente:

Ejemplo





Algoritmos de decisión

- Comprobar el resultado de la minimización con la herramienta JFLAP
- Cómo usar el cálculo de particiones o tabla de estados distinguibles para comprobar la equivalencia entre 2 autómatas deterministas