Autómatas finitos determinísticos y Lenguajes regulares

Universidad Nacional de Quilmes

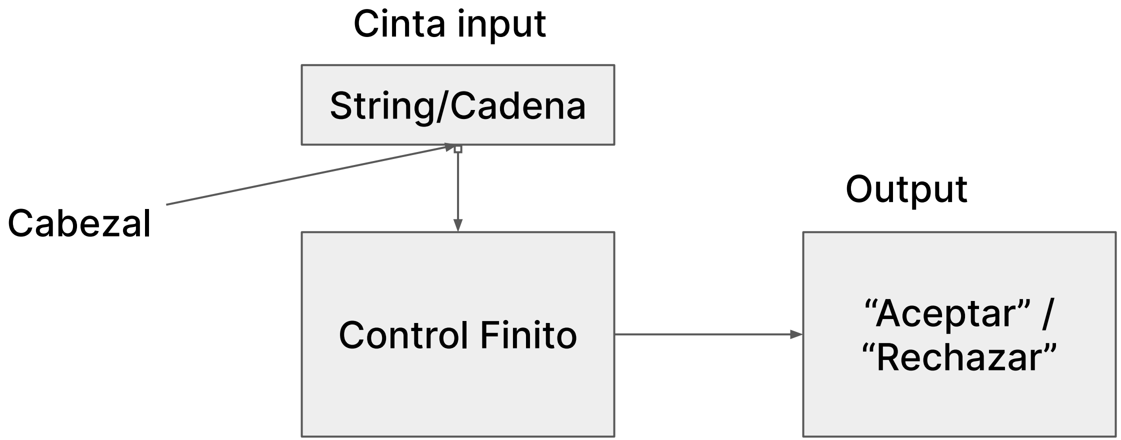
Emanuel Alvaredo, Federico Lochbaum

Autómata finito determinístico

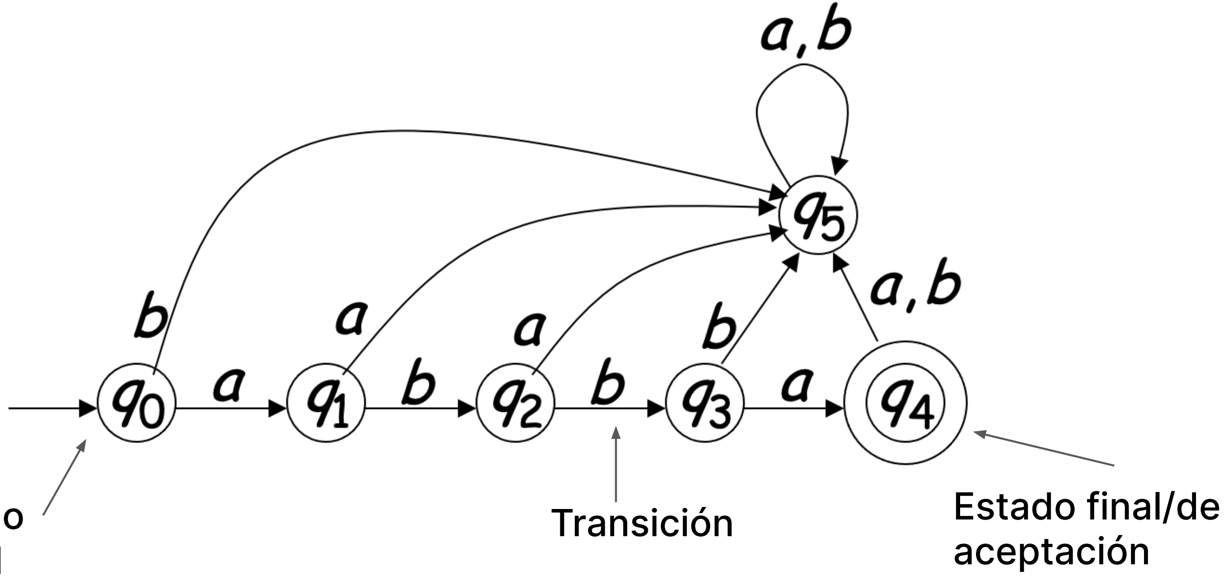
Máquina de estados: En cada momento, la máquina se encuentra en un único estado

Estímulos/Interacciones: Para cada estímulo y en cada estado hay una única respuesta

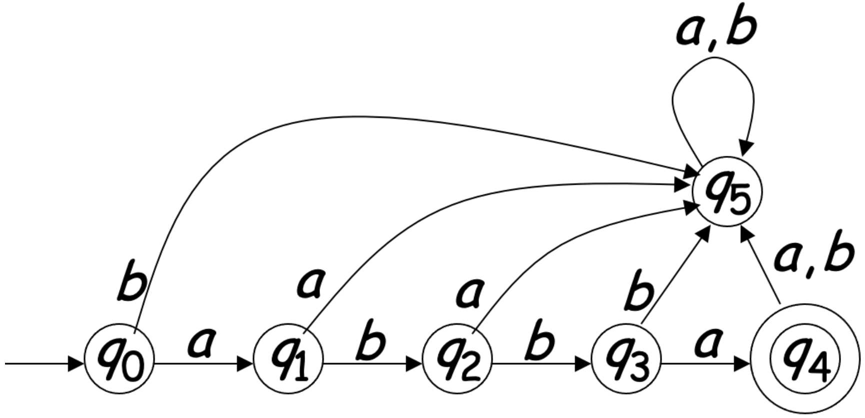
AFD versión “reconocedor”



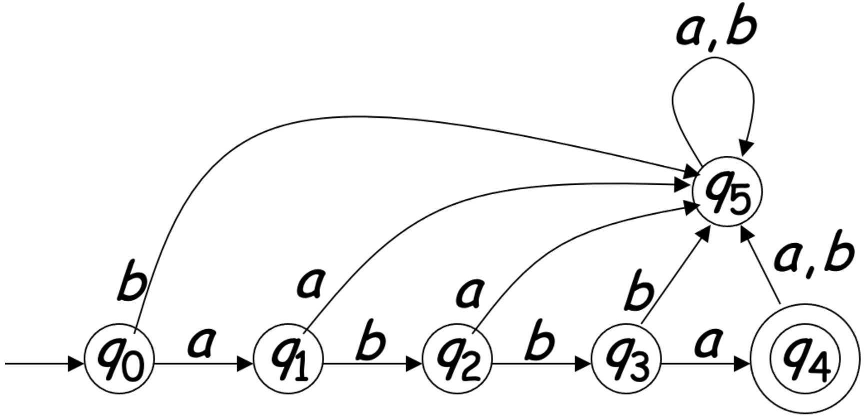
Grafo de transición



Cual es el alfabeto (Σ) ?



Cual es el alfabeto (Σ) ? $\Sigma = \{a, b\}$

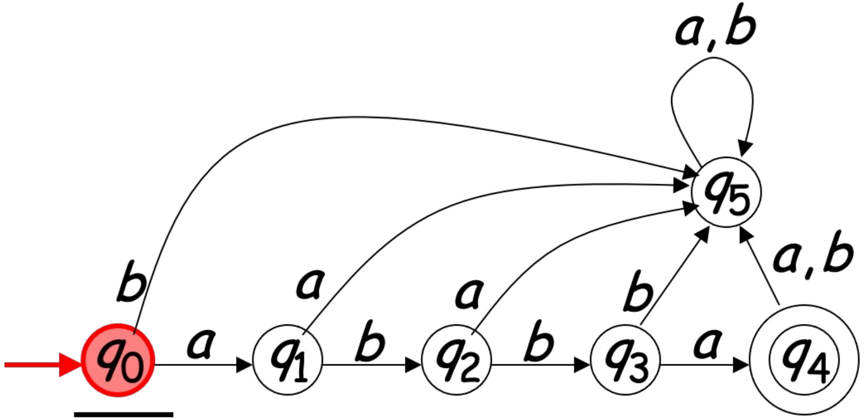


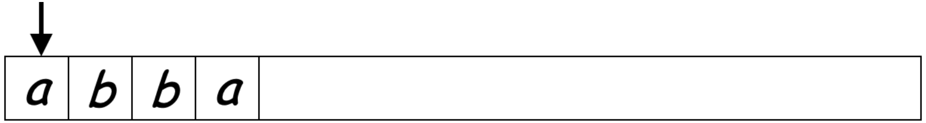
Nota: Para **todo** estado, existe una **única** transición para **cada** símbolo en el alfabeto

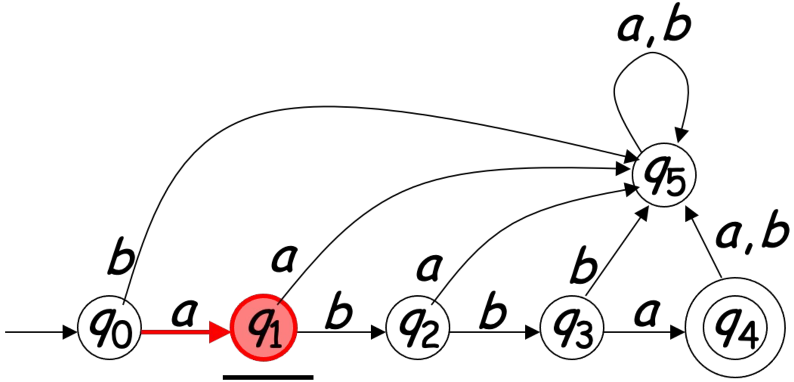
Caso de aceptación

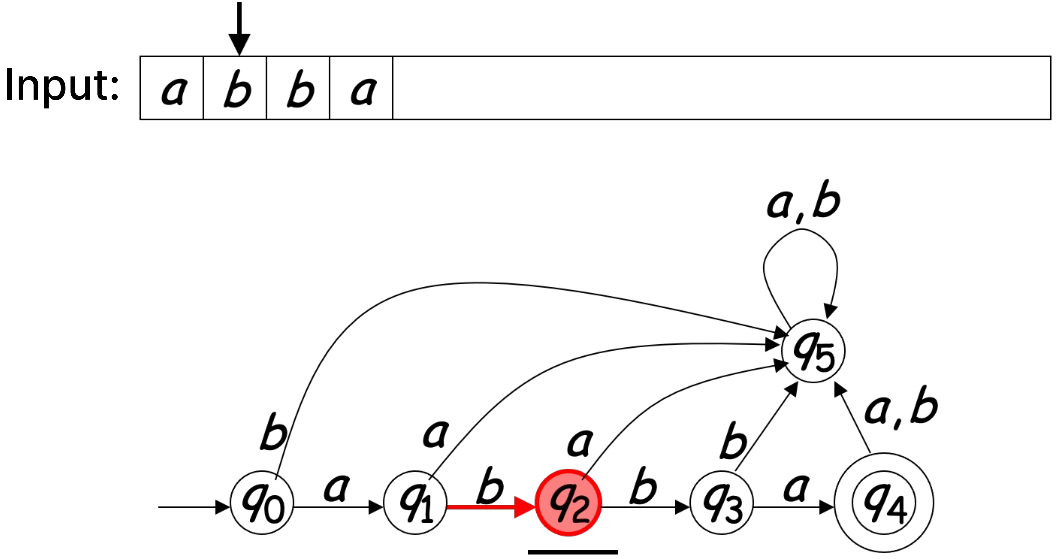
Cabezal ↓

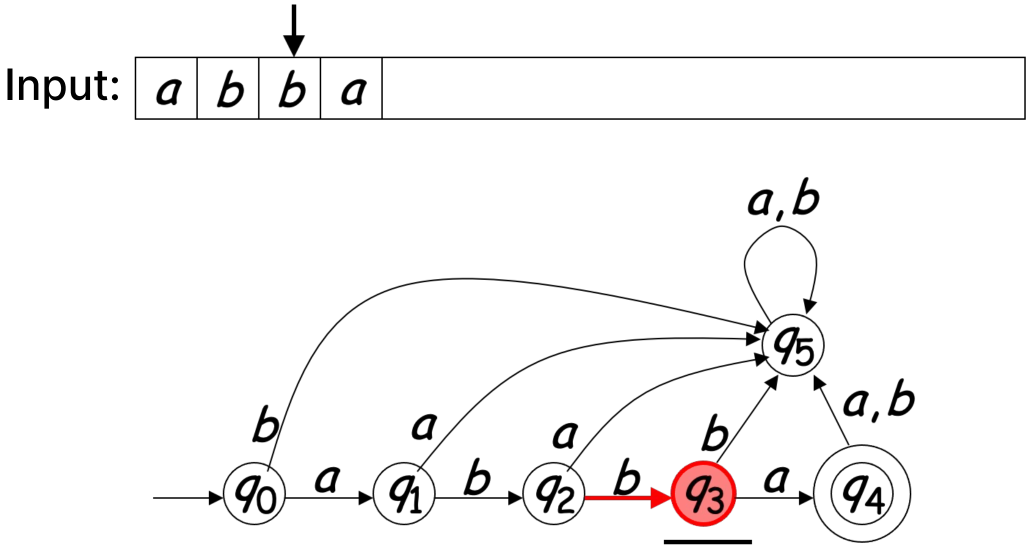
Input: | a | b | b | a | |



Input: 

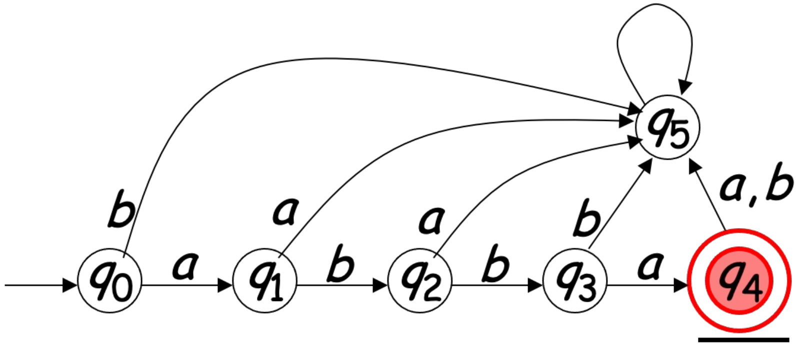



Input:
a
b
b
a
a,b
b
a
a
b
a,b
q0
a
q1
b
q2
b
q3
a
q4
q5



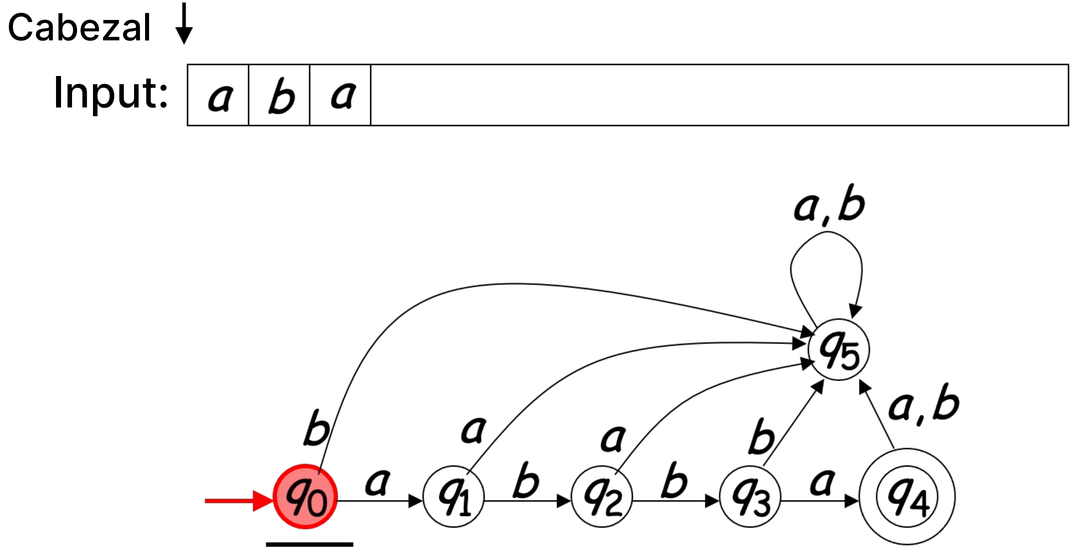
Input:
a
b
b
a
a,b
b
a
a
b
a,b
a
b
b
a
q_0
q_1
q_2
q_3
q_4
q_5

Input: a b b a

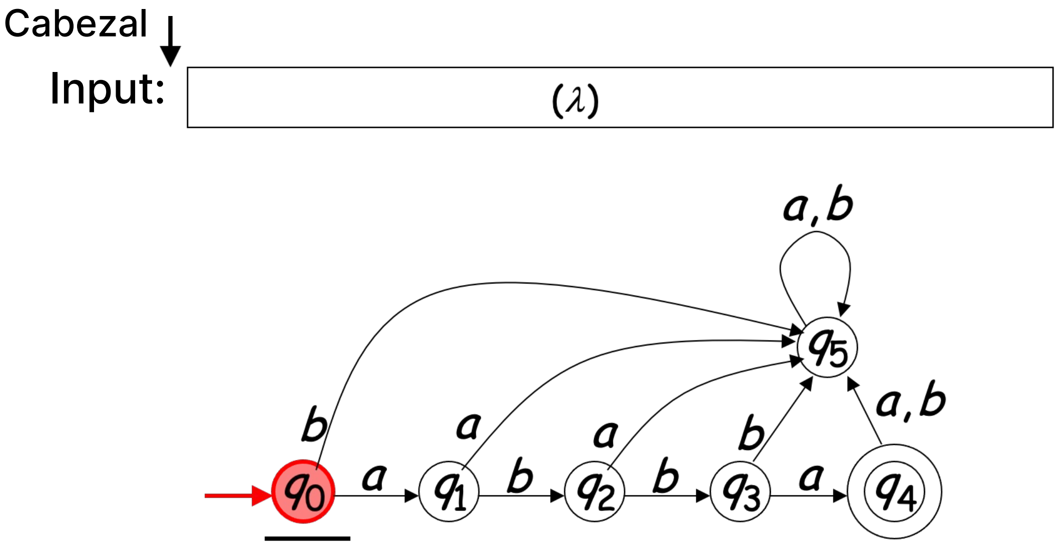


El string “abba” es aceptado ya que finaliza con el estado final q4

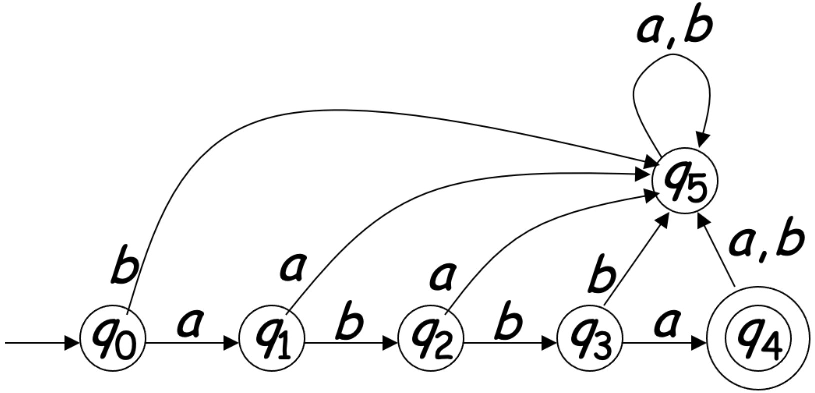
Caso de rechazo



Otro caso de rechazo

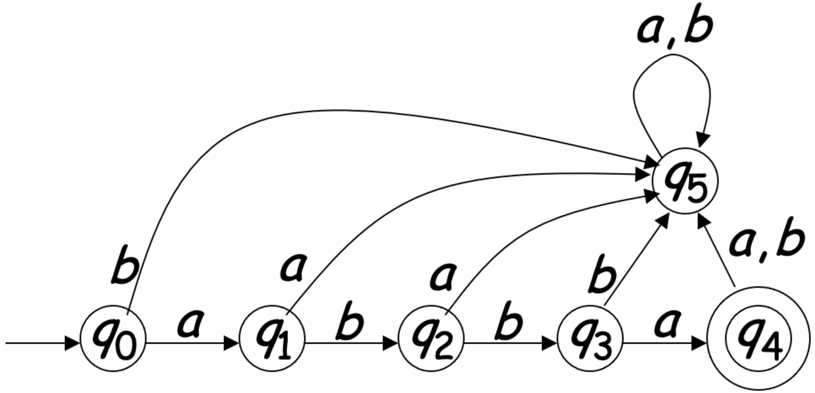


Cual es el lenguaje aceptado por este autómata ?



Cual es el lenguaje aceptado por este autómata ?

$$L = \{abba\}$$



Resumiendo...

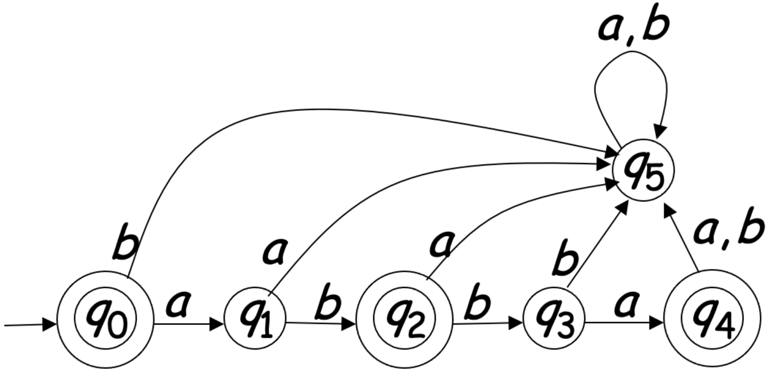
Para **aceptar** una cadena, el AFD consumió todo el string donde el último estado era el estado final

Para **rechazar** una cadena, el AFD consumió todo el string donde el último estado NO era el estado final

Vamos con otro

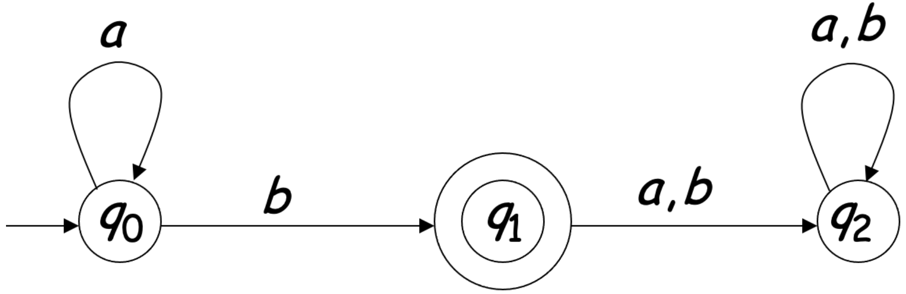
Qué pasa ahora con las cadenas aceptadas ?

$$L = \{\lambda, ab, abba\}$$



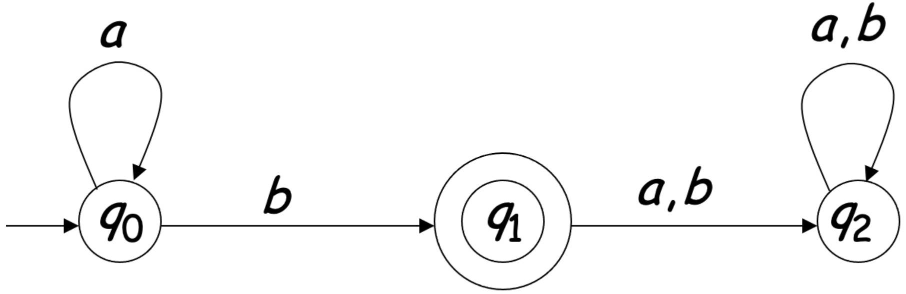
Y otro...

Cuál es el lenguaje ?



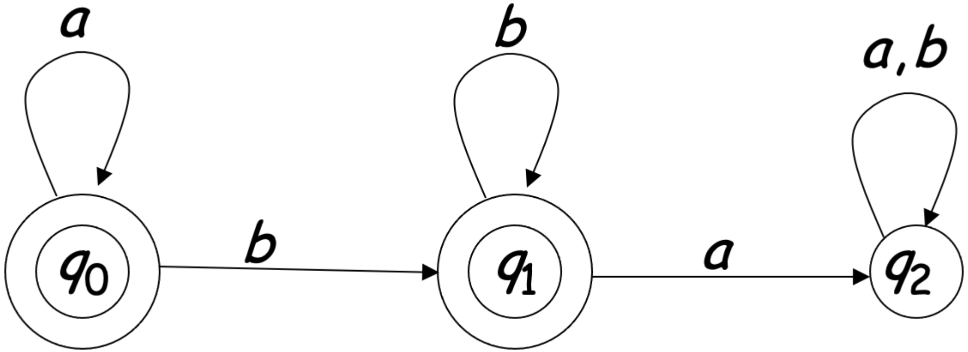
Cuál es el lenguaje ?

$$L = \{a^n b : n \geq 0\}$$



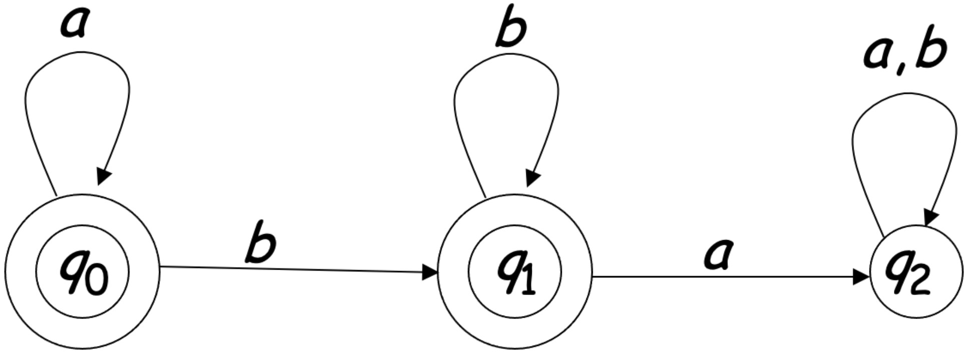
Último?

Cuál es el lenguaje ?



Cuál es el lenguaje ?

$$L = \{a^n b^m : n, m \geq 0\}$$



Definición formal de AFD

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

***Q* : Conjunto de estados**

***Σ* : Alfabeto de input**

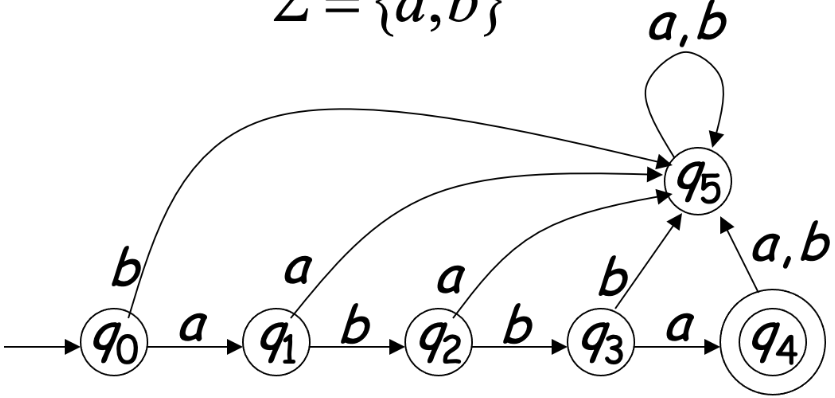
***δ* : Función de transición**

q_0 : Estado inicial ($q_0 \in Q$)

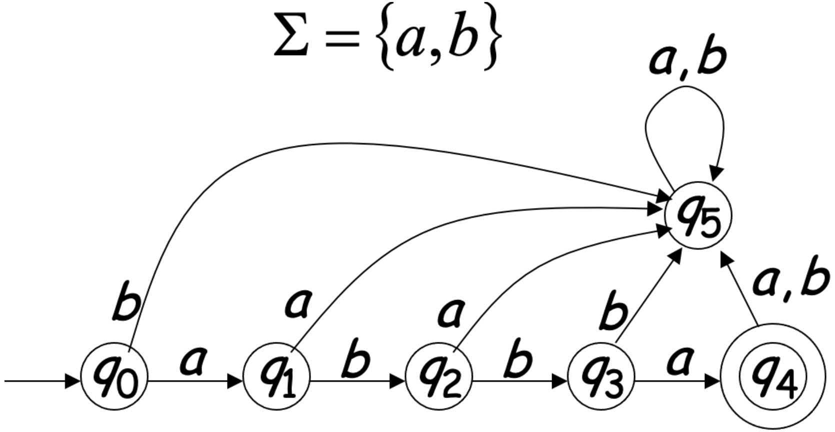
F : Conjunto de estados finales ($F \subset Q$)

Volvemos al ejemplo

$$\Sigma = \{a, b\}$$



Volvemos al ejemplo



M = (*Q*, *Σ*, *δ*, *q0*, *F*)

Q *: { q_0, q_1, q_2, q_3, q_4, q_5 }*

Σ *: { a, b }*

q_o *: q_0*

F *: { q_4 }*

δ *: ?*

Función de transición

$$\delta : Q \times \Sigma \rightarrow Q$$

La función de transición describe el resultado de una transición desde el estado q con símbolo símbolo x

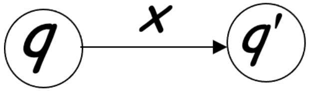
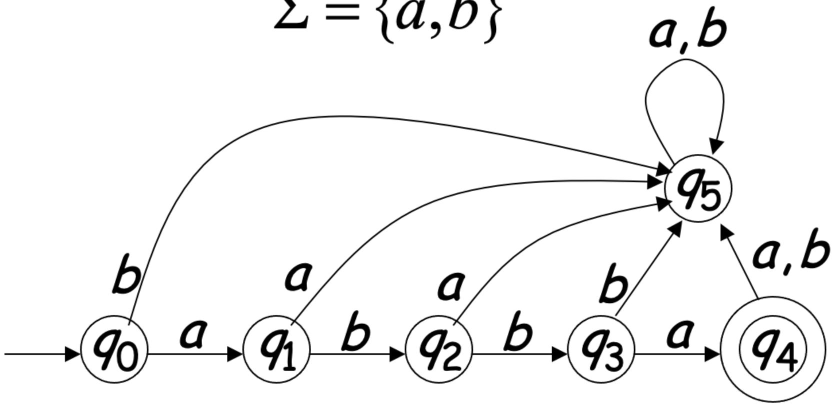


Tabla de transición

$$\Sigma = \{a, b\}$$



δ	a	b
q_0	q_1	q_5
q_1	q_5	q_2
q_2	q_5	q_3
q_3	q_4	q_5
q_4	q_5	q_5
q_5	q_5	q_5

Función de transición extendida

$$\delta^* : Q \times \Sigma^* \to Q$$

La función de transición extendida describe el estado resultante luego de leer el string w a partir del estado q

$$\delta^*(q, w) = q'$$

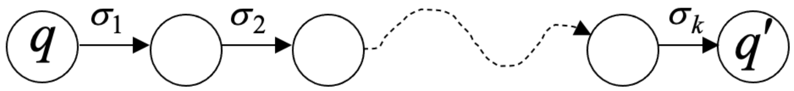
Caso especial: Para cualquier estado q, $\delta^*(q, \lambda) = q$

Definición inductiva

$$\delta^*(q,\lambda) = q$$
$$\delta^*(q,aw) = \delta^*(\delta(q,a),w)$$
$$q \in Q, a \in \Sigma, w \in \Sigma^*$$

$\delta^*(q,w) = q'$ **implica que existe un camino de transiciones**

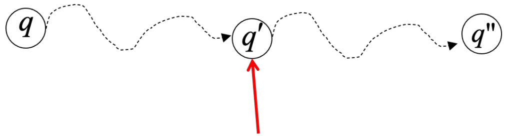
$$w = \sigma_1 \sigma_2 \cdots \sigma_k$$



Teorema!

Teo (asoc, δ):

$$\delta^*(q, vw) = \delta^*\left(\delta^*(q, v), w\right) \quad v \in \Sigma^*, w \in \Sigma^*$$



$$\delta^*(q, v) = q'$$

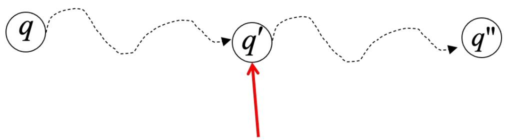
Demo (Por inducción sobre la estructura de v) :

Ideas?

Teorema!

Teo (asoc, δ):

$$\delta^*(q,vw)=\delta^*\left(\delta^*(q,v),w\right) \quad v\in\Sigma^*, w\in\Sigma^*$$



$$\delta^*(q,v)=q'$$

Demo (Por inducción sobre la estructura de v) :

(i) *Caso base*: $\quad v=\lambda,$

$$\delta^*(q,vw)=\delta^*(q,\lambda w)\overset{def\ de\ \lambda}{=}\delta^*(q,w)\overset{def\ de\ \delta^*}{=}\delta^*\left(\delta^*(q,\lambda),w\right)$$

(ii) *Hipótesis inductiva*: Vale el enunciado *para* todo q' con v' de tamaño menor o igual a n

(iii) $v=av'$ (v de tamaño $n+1$)

HI

$$\delta^*(q,vw)=\delta^*(q,av'w)\overset{def\ de\ \delta^*}{=}\delta^*(\delta(q,a),v'w)\overset{\delta(q,a)=q'}{=}\delta^*(q',v'w)\overset{(ii)}{=}$$

$$\delta^*\left(\delta^*(q',v'),w\right)\overset{\delta(q,a)=q'}{=}\delta^*\left(\delta^*(\delta(q,a),v'),w\right)\overset{def\ de\ \delta^*}{=}\delta^*\left(\delta^*(q,av'),w\right)=$$

$$\delta^*\left(\delta^*(q,v),w\right)$$

Lenguaje “aceptado” por un AFD

Sea M un AFD, denotamos L(M) como el lenguaje aceptado por M

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

Decimos que L’ es aceptado por M si L(M) = L’

$$L(M) = \left\{ w \in \Sigma^* : \delta^*(q_0, w) \in F \right\}$$

Lenguaje “rechazado” por un AFD

Sea M un AFD, denotamos $\overline{L(M)}$ como el lenguaje rechazado por M

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

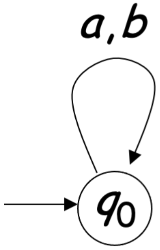
Decimos que L’ es rechazado por M si $\overline{L(M)}$ = L’

$$\overline{L(M)} = \{ w \in \Sigma^* : \delta^*(q_0, w) \notin F \}$$

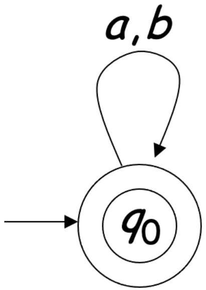
Un break ? :)

Ejemplitos

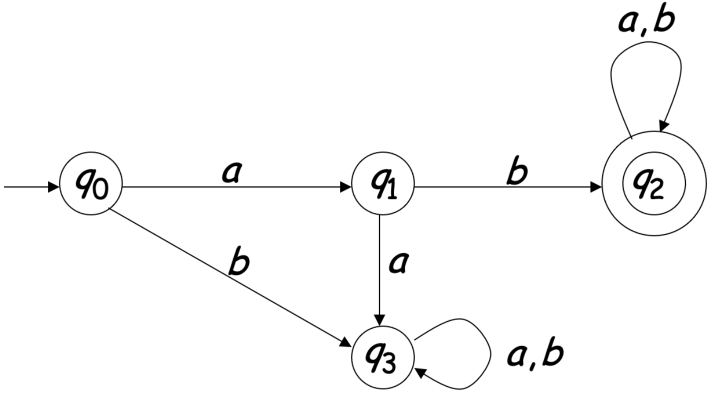
$\Sigma = \{a, b\}$



$L(M) = \{\ \}$



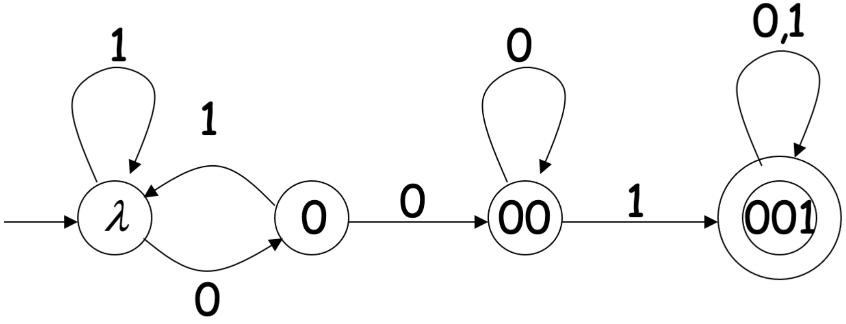
$L(M) = \Sigma^*$



$L(M)$ **= ?**

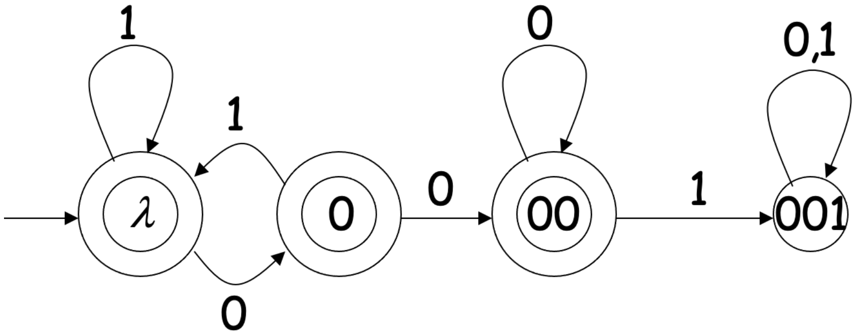
Más ejemplitos

L(M) = { todos los strings binarios que contienen el substring 001 }



Más ejemplitos

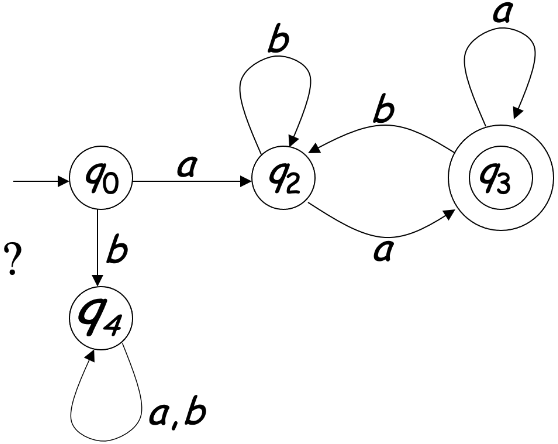
L(M) = { todos los strings binarios sin el substring 001 }



M satisface una especificación L ?

$$L = \left\{awa : w \in \{a,b\}^*\right\}$$

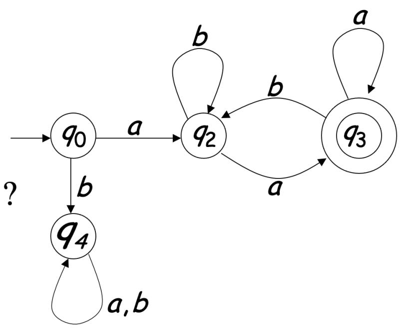
¿$L = L(M)$?



M satisface una especificación L ?

$$L = \{awa : w \in \{a,b\}^*\}$$
$$L(M) = \{x \in \Sigma^* : \delta^*(q_0, x) \in F\} =$$
$$= \{x \in \Sigma^* : \delta^*(q_0, x) = q_3\}$$
$$¿L = L(M)?$$

$$¿L = L(M)?$$



Veremos que $L \subseteq L(M)$

Sea $x \in L$, entonces x es de la forma awa con $w \in \{a.b\}^*$

$$\delta^*(q_0, awa) = \delta^*(\delta(q_0, a), wa) = \delta^*(q_2, wa) \overset{asoc}{=} \delta^*(\delta^*(q_2, w), a)$$

En este punto *para* $\delta^*(q_2, w)$ hay dos casos posibles :

$\delta^*(q_2, w) = q_3$ o bien $\delta^*(q_2, w) = q_2$, *entonces*

dado que $\delta^*(\delta^*(q_2, w), a) = \delta(\delta^*(q_2, w), a)$ y que

$\delta(q_3, a) = \delta(q_2, a) = q_3$, *tenemos que*

$\delta^*(q_0, x) = \delta^*(q_0, awa) = \delta^*(q_2, wa) = q_3$ *esto es*, $x \in L(M)$

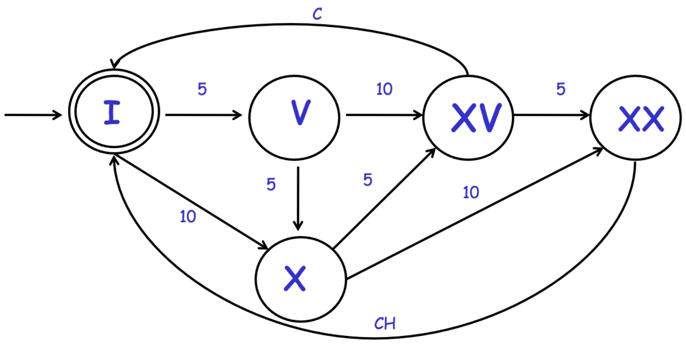
La máquina expendedora

- Ranura para monedas (acepta monedas de 5 y 10 centavos)
- Bandeja expendedora (permite retirar chicles y caramelos)
- Botones “C” (para pedir un caramelo) y “CH”(para pedir un chicle)
- Parlante (hace “click” cada vez que se inserta 5 centavos y “click-click” cuando se inserta 10 centavos)
- Un caramelo cuesta 15 ctvs y un chicle 20ctvs.

Formalizamos las secuencias de entrada

- Olvidamos las salidas y los posibles errores para capturar la lógica básica del problema.
- Obtenemos una versión totalmente abstracta del problema: Un conjunto de secuencias correctas L0 y su clausura L_0^*

La máquina expendedora (AFD)

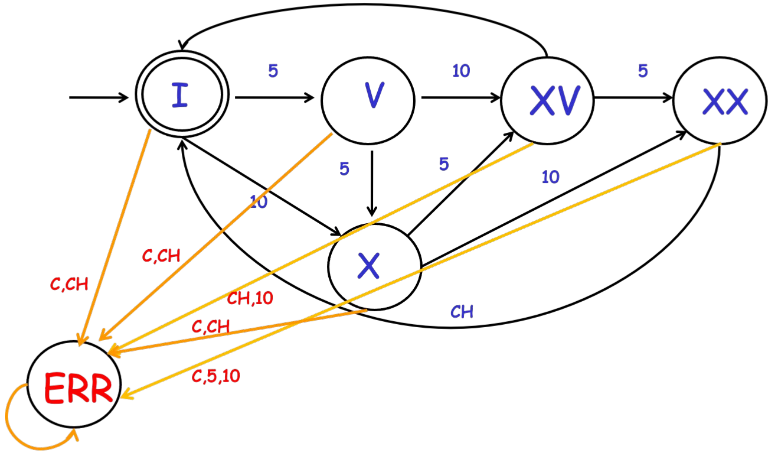


$Q = \{ I, V, X, XV, XX \}$ $\Sigma = \{ 5, 10, C, CH \}$ $q_0 = I$ $F = \{ I \}$

La máquina expendedora (AFD)

Debemos completarlo!

L_0={ λ,510C,555C,105C,5105CH,5555CH,1055CH, 5510CH, 1010CH }



Q= {I, V, X, XV, XX, ERR} Σ = {5, 10, C, CH} q0= I F= {I}

Nuevo estado ERR (estado trampa) y función de transición: δ(I, s) = δ(V, s)= δ(X, s) = ERR, para s ∈ {C, CH}, δ (XV, s) = ERR, para s ∈ {CH, 10}, δ(XX, s) = ERR, para s ∈ {C, 5, 10} δ(ERR, s) =ERR, para s ∈ Σ

Teorema: El lenguaje L reconocido por el autómata M es L_0^*

L_0={ λ,510C,555C,105C,5105CH,5555CH,1055CH, 5510CH, 1010CH }

Probamos que $L(M) = L_0^*$?

Por **def** de igualdad de conjuntos, debemos mostrar que

1. $L(M) \subset L_0^*$
2. $L_0^* \subset L(M)$

Probamos 2) es decir, ($\mathbf{L_0^* \subset L(M)}$), 1) queda como tarea

Sabemos que para cualquier lenguaje L, M tiene la forma $M = (Q, \Sigma, \delta, q_0, F)$ tenemos que

$$\mathbf{L(M) = \{ w^* \mid \delta^*(q_0, w) \in F\}}$$

Entonces si mostramos que **para todo** i, **para todo** w $\in L_0^i$, $\mathbf{\delta^*(I, w)}$

Por inducción en i

- Caso base $\mathbf{L_0^i \subset L(M)}$ (i = 0)

  Trivial, veamos caso por caso que cada uno de las nueve cadenas (que podría ser w) de L_0={ λ, 510C, 555C, 105C, 5105CH, 5555CH, 1055CH, 5510CH, 1010CH } que son aceptadas por M

- Caso inductivo $\mathbf{L_0^{i+1} \subset L(M)}$ (i >= 0)

  Sabemos que $L_0^{i+1} = L_0^i L_0$

  HI: Suponemos que $L_0^i \subset L(M)$, **para todo w ∈ $\mathbf{L_0^i}$**, sucede que δ*(I,w) = I

Entonces probar $\mathbf{L_0^{i+1} \subset L(M)}$ equivale a probar que para todo $\mathbf{w \in L_0^{i+1}}$, sucede que $\boldsymbol{\delta^*(I,w) = I}$

Sea $w \in L_0^{i+1}$ entonces $w = xy$ con $\boldsymbol{x \in L_0^i}$ e $\boldsymbol{y \in L_0}$

Entonces

Por HI, $L_0^i \subset L(M)$ sabemos que $\delta^*(I, x) = I$, además sabemos por caso base que $\delta^*(I, y) = I$ y por asociación tenemos que

- $\delta^*(I, w) = \delta^*(I, xy)$ **POR DEF**
- $\delta^*(I, xy) = \delta^*(\delta^*(I, x), y)$ **POR ASSOC**
- $\delta^*(\delta^*(I, x), y) = \delta^*(I, y)$ **POR HI + CASO BASE**

Entonces $\mathbf{L_0^* \subset L(M)}$

Lenguajes regulares

Def: Un lenguaje L es ***regular*** si existe un AFD M que lo acepta, es decir, que L(M) = L

Los lenguajes aceptados por todos los AFDs forman la familia de los lenguajes regulares

Ejemplos de lenguajes regulares

$\{abba\}$ $\{\lambda, ab, abba\}$ $\{a^n b^m : n, m \geq 0\}$

$\{a^n b : n \geq 0\}$ $\{awa : w \in \{a,b\}^*\}$

{strings en {a,b}* con prefijo ab }

{strings binarios sin substring 001 }

$\{x : x \in \{1\}^* \text{ y } x \text{ es par}\}$

$\{\ \}$ $\{\lambda\}$ $\{a,b\}^*$

Y si alguien se lo pregunta.... si! existen lenguajes que **NO** son regulares!

Eso significa que **NO** existe un **AFD** que acepte estos lenguajes

$$L = \{a^n b^n : n \geq 0\}$$

Lo demostramos en la próxima clase! :)