TEMA 3. AUTÓMATAS FINITOS:

Un autómata finito es un modelo computacional que realiza cómputos de forma automática sobre una entrada para producir una salida. Se representan mediante la quíntupla: $A = (\Sigma, Q, f, q_0, F)$ donde:

Σ : es un alfabeto, llamado "alfabeto de entrada"
Q : es un conjunto finito no vacio llamado conjunto de estados.
f : es el conjunto de funciones de transición ($f: Q \times \Sigma \rightarrow Q$)
q_0 : es el estado inicial del autómata ($q_0 \in Q$)
F: es el conjunto no vacío de "estados finales" ($F \subset Q$)

AUTÓMATA FINITO DETERMINISTA (AFD): es un autómata en el que cada estado $q \in Q$ en que se encuentre el autómata, y con cualquier símbolo $a \in \Sigma$ del alfabeto de entrada, existe siempre una transición posible $f(q, a)$.

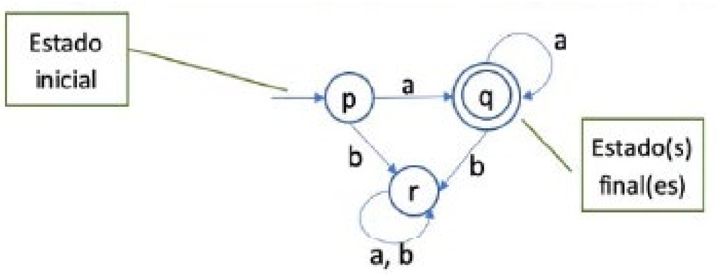
Un autómata finito puede ser representado por un diagrama o tabla de transición.
$AFD = < \Sigma = \{a, b\}, Q = p, q, r\}, f, q_0 = p, F = \{q\} >$

$f(p, a) = q$ $\quad f(p, b) = r$ $\quad f(q, a) = q$
$f(q, b) = r$ $\quad f(r, a) = r$ $\quad f(r, b) = r$

Estado inicial

Estado(s) final(es)

	a	b
→p	**q**	**r**
(q)	**q**	**r**
r	**r**	**r**



Lenguaje que coincide con el autómata anterior es $L(A) = \{a^n / n \geq 1\}$

<u>Reconocimiento de un Lenguaje por parte de un AFD</u>: una palabra es reconocida o aceptada por un autómata si partiendo del estado inicial se llega al estado final y la concatenación de las etiquetas de las transiciones realizadas conforman la palabra.

Se llama lenguaje asociado o reconocido por un autómata al conjunto de todas las palabras aceptadas por este.

$$L = \{ x \mid x \in \Sigma^* \ \& \ f(q_0, x) \in F \}$$

Para ampliar la definición de la función de transición para que actúe sobre todas las palabras ($Q \times \Sigma^* \rightarrow Q$), añadimos:

- $f(q, \lambda) = q$ para todo $q \in Q$
- $f(q, ax) = f(f(q, a), x) \quad \forall a \in \Sigma, \quad x \in \Sigma^*, \quad Q \in \Sigma$

Definiciones extra:

- Si $F = \emptyset$, $L = \emptyset$ - Si $F = Q$, $L = \Sigma^*$
- Para que $\lambda \in L(A)$ se tiene que cumplir que $q_0 \in F$

ESTADOS ACCESIBLES: Un estado es accesible si existen transiciones mediante las cuales se llegue a ese estado.
Todo estado es accesible a si mismo dado que $f(q, \lambda) = q$

LEMA: Sea $<\Sigma, Q, f, q_0, F>$ un Autómata Finito Determinista tal que $|Q| = n$. El estado p es accesible desde el estado q $\Leftrightarrow$ $x \in \Sigma^*$, $|x| < n$, tal que $f(q, x) = p$.

Demostración

- $\Leftarrow$ Elemental según la definición de accesibilidad
- $\Rightarrow$ Sea p accesible desde q $\Rightarrow \exists\, y \ni \Sigma^* \,/\, f(q, y) = p$
  - si $|y| < n$ está demostrado
  - si $|y| \geq n$ entonces el autómata pasará por $1+|y|$ estados (contando el punto de partida y final, desde "p" a "q").

    Como $|Q|=n \wedge |y| \geq n \Rightarrow 1+|y|>n \Rightarrow$ el autómata finito tiene que pasar más de una vez por un estado, al menos.

    Supongamos q....r.....r.....p
    u v w

    $y=uvw \Rightarrow$ **se puede pasar de "q" a "p" con la palabra y_1=uw**

    $|y_1| < |y|$

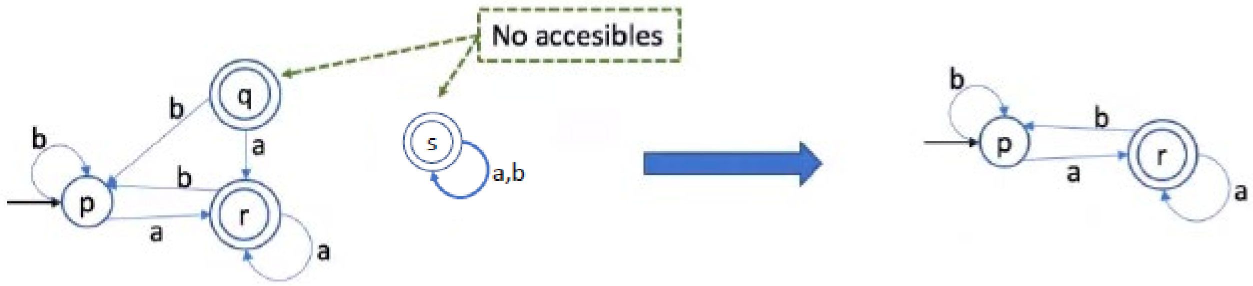
Caso y_1**=uw**

- Si $|y_1| < n$ está demostrado
- Si $|y_1| \geq n$ se repite el proceso $y_2 = u'w'$
  - $|y_2| < |y_1| < |y|$
  - En algún momento llegamos a $|y_i| < n$

Colorario: Sea $<\Sigma, Q, f, q_0, F>$ un Autómata Finito Determinista tal que $|Q| = n$. Entonces, el lenguaje aceptado por el autómata es no vacío, si y sólo si el autómata acepta al menos una palabra $x \in \Sigma^*$ tal que $|x| < n$.

AUTÓMATAS CONEXOS: Un autómata es conexo si todos los estados de dicho autómata son accesibles desde el estado inicial tras un nº finito de transiciones.

Dado un autómata no conexo, se puede obtener un autómata que si es conexo y que reconozca el mismo lenguaje al eliminar todos los estados que no sean accesibles desde el estado inicial.



ESTADOS EQUIVALENTES: sea AFD = $< \Sigma, Q, f, q_0, F >$, decimos que dos estados $p, q \in Q$ son equivalentes (pEq) si para toda palabra $x \in \Sigma^*$, se verifica que:
$f(p, x) \in F \iff f(q, x) \in F$

También se puede decir que son equivalentes en función de su longitud "k" si para toda palabra $x \in \Sigma^*$ $|x| \leq k$, se verifica que $f(p, x) \in F \iff f(q, x) \in F$

Si $pEq \rightarrow pE_nq$ para todo n, como consecuencia $pE_kq \rightarrow pE_nq$ para todo $n \leq k$

Las relaciones de E y E_k en Q son relaciones de equivalencia, como inducen sobre Q a una partición de clase existen los conjuntos cociente Q/E, Q/E_k.

P_E = {conjuntos de clases de equivalencia}
$P_E = \{[q] / q \in Q\}$ donde $[q] = \{p / p \in Q \wedge pEq\}$ $\qquad Q/E = P_E \qquad Q/E_k = P_k$

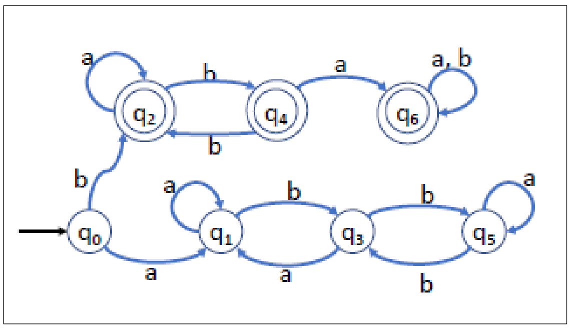
Propiedades:

1) $pEq \rightarrow pE_kq \quad \forall k$
2) $pE_kq \rightarrow pE_kq \quad \forall n \leq k$
3) $pEq \rightarrow f(p, x)\ E\ f(q, x) \qquad \forall x \in \Sigma^*$
4) pE_kq no implica $f(p, x)\ E_k\ f(q, x) \qquad \forall x \in \Sigma^*$
5) $pE_{k+1}q \iff pE_kq \wedge f(p, e)\ E_k\ f(q, e) \qquad \forall e \in \Sigma$
6) $P_k = P_{k+1} \rightarrow P_{k+i} = P_k \quad \forall i \geq 0$
7) $P_k = P_{k+1} \rightarrow P_E = P_k$
8) Sea $|Q| = n$, $n > 1$ entonces $\exists j \leq n - 2 / P_j = P_{j+1}$

Algoritmo para construir Q/E (Conjunto Cociente):

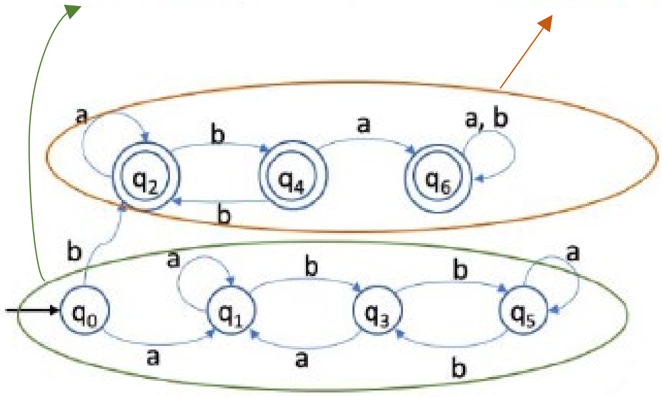
1. Construir Q/E_0, usando la siguiente regla:
   - p y q están en la misma clase $\iff$ son estados finales o no lo son
   - es decir $\mathbf{P_0 = \{F, Q\text{-}F\}}$
2. **Obtener $\mathbf{P_{K+1}}$**
   - Supongamos que $P_n = \{C1, C2, \dots Cm\}$ para construir P_{n+1} se usa la regla:
     - "p" y "q" están en la misma clase de $P_{n+1} \iff$ p y q pertenecen a la misma clase C_j $(1 \leq j \leq m)$ en y se verifica que $\forall a \in \Sigma$; $f(p, a)$ y $f(q, a)$ están en la misma C_i $(1 \leq i \leq m)$ de P_n
3. Si $\mathbf{P_K = P_{K+1}}$ **entonces** $\mathbf{P_K = P_E}$. En caso contrario, aplicar de nuevo el **paso 2**.

Ejercicio: Dado el AFD calcular PE o Q/E:

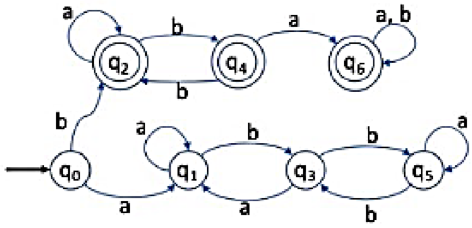


Paso 1: Definimos P_0

- $P_0 = \{ A_1$(no finales) $= [q0, q1, q3, q5]$, A_2 (finales) $= [q2, q4, q6]\}$



Calcular P_1



$P_0 = \{ A_1 = [q0, q1, q3, q5]$,
$A_2 = [q2, q4, q6]\}$

Para los estados de A_1

$f(q0, a) = q1 \in A_1$	$f(q0, b) = q2 \in A_2$
$f(q1, a) = q1 \in A_1$	$f(q1, b) = q3 \in A_1$
$f(q3, a) = q1 \in A_1$	$f(q3, b) = q5 \in A_1$
$f(q5, a) = q5 \in A_1$	$f(q5, b) = q3 \in A_1$

no cumple,

hay que separar q0

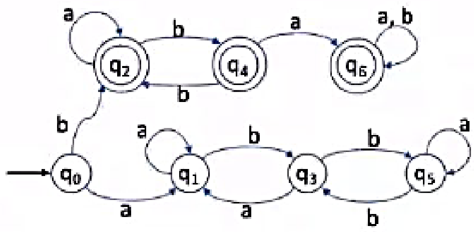
Para los estados de A_2

$f(q2, a) = q2 \in A_2$	$f(q2, b) = q4 \in A_2$
$f(q4, a) = q6 \in A_2$	$f(q4, b) = q2 \in A_2$
$f(q6, a) = q6 \in A_2$	$f(q6, b) = q6 \in A_2$

$P_1 = \{ B_1 = [q1, q3, q5]$
$B_2 = [q0]$
$B_3 = [q2, q4, q6]\}$

Como $P_1 \neq P_0$ Calcular P_2

Calcular P_2



$P_1 = \{ B_1 = [q1, q3, q5]$
$B_2 = [q0]$
$B_3 = [q2, q4, q6]\}$

Para los estados de B_1

$f(q1, a) = q1 \in B_1$	$f(q1, b) = q3 \in B_1$
$f(q3, a) = q1 \in B_1$	$f(q3, b) = q5 \in B_1$
$f(q5, a) = q5 \in B_1$	$f(q5, b) = q3 \in B_1$

Como $P_2 = P_1$ FIN:

$Q/E = P_E = P_1 = \{ B_1 = [q1, q3, q5]$

$B_2 = [q0]$

$B_3 = [q2, q4, q6]\}$

AUTÓMATAS EQUIVALENTES

Sean $A = < \Sigma, Q, f, q_0, F >$ y $A' = < \Sigma', Q', f', q_0', F' >$ dos autómatas, se dice que son equivalentes entre sí, si ambos reconocen el mismo lenguaje.

La condición necesaria y suficiente para que A_1 y A_2 sean equivalentes es que $q_0 E q_0'$.

Para comprobar la equivalencia entre autómatas se utiliza:

Suma Directa $\oplus$:
Dados $A_1 = < \Sigma, Q_1, f_1, q_{01}, F_1 >$ y $A_2 = < \Sigma, Q_2, f_2, q_{02}, F_2 >$ $Q_1 \cap Q_2 = \emptyset$

$$A_1 \oplus A_2 = < \Sigma, Q_1 \cup Q_2, f, q_0, F_1 \cup F_2 >$$

Con f:
$f: (Q_1 \cup Q_2) \times \Sigma \rightarrow Q_1 \cup Q_2$
$f(q, a) = f_1(q, a)$ si $q \in Q_1$
$f(q, a) = f_2(q, a)$ si $q \in Q_2$
$\forall a \in \Sigma$

- Si se representa la **tabla de transición** se pone una debajo de la otra.
- El **diagrama de transición** es el conjunto de los 2 diagramas.

Con q_0 cualquier que sea q_{01} o q_{02}

Teorema

Sean los autómatas $A_1 = <\Sigma, Q_1, f_1, q_{01}, F_1>$ y $A_2 = <\Sigma, Q_2, f_2, q_{02}, F_2>$ tal que

- $Q_1 \cap Q_2, = \emptyset$
- $|Q_1| = n_1$
- $|Q_2| = n_2$

A1 ≡ A2 $\Leftrightarrow$ $\mathbf{q_{01} E q_{02}}$ **en $A_1 \oplus A_2$**, es decir, si aceptan las mismas palabras.

$p \in Q_1$ y $q \in Q_2$ son equivalentes $\Leftrightarrow$

- $f_1(p, x) \in F_1 \Leftrightarrow f_2(q, x) \in F_2 \ \forall x \in \Sigma^*$ con $|x| \leq n_1 + n_2 - 2$

$n_1 + n_2 - 2$ es el menor valor en general que verifica ii)

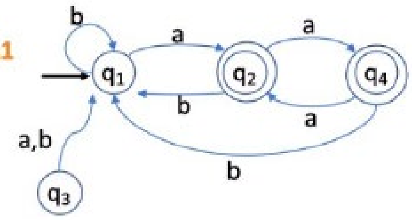
Algoritmo para comprobar la equivalencia entre autómatas con Suma Directa:

1. Se construye $A_1 \oplus A_2$
2. Se obtiene Q/E de la suma directa $(Q_1 \cup Q_2) / E = P_E$
3. $A1 \equiv A2 \Leftrightarrow$ q_{01} y q_{02} están en la misma clase C_i en P_E

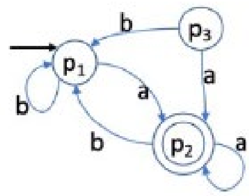
Ejercicio: Dado los AFDs comprobar si son equivalentes mediante la suma directa.

$A_1 \oplus A_2 = \langle \Sigma = \{a, b\}, Q = \{q1, q2, q3, q4, p1, p2, p3\}, f, p1, F = \{q2, q4, p2\} \rangle$

A1



A2



f en forma de tabla de transición

El inicial puede ser q_1 o p_1, elegimos p_1

	a	b
→p1	p2	p1
(p2)	p2	p1
p3	p2	p1
q1	q2	q1
(q2)	q4	q1
q3	q1	q1
(q4)	q2	q1

Definimos P_0

$P_0 = \{ A_1(\text{finales}) = [p2, q2, q4], A_2 (\text{no finales}) = [p1, p3, q1, q3]\}$

	a	b
→p1	p2	p1
(p2)	p2	p1
p3	p2	p1
q1	q2	q1
(q2)	q4	q1
q3	q1	q1
(q4)	q2	q1

Calcular P_1

Para los estados de A_1

$f(p2, a) = p2 \in A_1$ $\quad$ $f(p2, b) = p1 \in A_2$

$f(q2, a) = q4 \in A_1$ $\quad$ $f(q2, b) = q1 \in A_2$

$f(q4, a) = q2 \in A_1$ $\quad$ $f(q4, b) = q1 \in A_2$

Para los estados de A_2

$f(p1, a) = p2 \in A_1$ $\quad$ $f(p1, b) = p1 \in A_2$

$f(p3, a) = p2 \in A_1$ $\quad$ $f(p3, b) = p1 \in A_2$

$f(q1, a) = q2 \in A_1$ $\quad$ $f(q1, b) = q1 \in A_2$

$f(q3, a) = q1 \in A_2$ $\quad$ $f(q3, b) = q1 \in A_2$

no cumple,

hay que separar q_3

$P_1 = \{ B_1 = [p2, q2, q4], B_2 = [p1, p3, q1], B_3 = [q3]\}$

Calcular P_2

Para los estados de B_1

$f(p2, a) = p2 \in B_1$ $\quad$ $f(p2, b) = p1 \in B_2$

$f(q2, a) = q4 \in B_1$ $\quad$ $f(q2, b) = q1 \in B_2$

$f(q4, a) = q2 \in B_1$ $\quad$ $f(q4, b) = q1 \in B_2$

Para los estados de B_2

$f(p1, a) = p2 \in B_1$ $\quad$ $f(p1, b) = p1 \in B_2$

$f(p3, a) = p2 \in B_1$ $\quad$ $f(p3, b) = p1 \in B_2$

$f(q1, a) = q2 \in B_1$ $\quad$ $f(q1, b) = q1 \in B_2$

	a	b
→p1	p2	p1
(p2)	p2	p1
p3	p2	p1
q1	q2	q1
(q2)	q4	q1
q3	q1	q1
(q4)	q2	q1

$P_1 = \{ B_1 = [p2, q2, q4], B_2 = [p1, p3, q1], B_3 = [q3]\}$

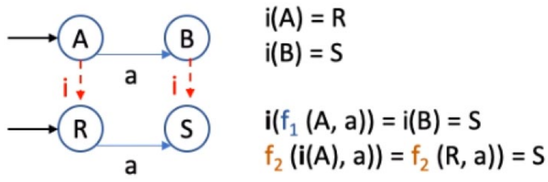
$P_2 = P_1$ luego $P_1 = P/E$

Isomorfismo: dados dos autómatas finitos deterministas A = < Σ, Q, f, q_0, F > y B = < Σ', Q', f', q_0', F' > se dice que son isomorfos si tienen el mismo número de estados (|Q| = |Q'|) y existe una aplicación biyectiva i: Q → Q' que cumple:

- i (q_0) = i (p_0)
  Los estados iniciales son correspondientes

- q ∈ F = i (q) ∈ F'
  Los estados finales se corresponden uno a uno

- i (f_1 (q, a)) = f_2 (i (q), a) ∀a ∈ Σ y ∀q ∈ Q



Si dos autómatas son isomorfos serán también equivalentes pero que sean equivalentes no quiere decir que sean isomorfos.

Autómata Cociente: sea A = < Σ, Q, f, q_0, F > un autómata finito determinista. Se construye a partir de a otro autómata A' = < Σ, Q/E, f', [q_0], F' > donde:

- f' : Q/E x Σ → Q/E   [q], e → f' ([q], e) = [f(q, e)]   e ∈ Σ
- F' = {[q] ∈ Q/E siendo q ∈ F}   es el conjunto de estados finales

A' es el autómata cociente

El Autómata Mínimo de un autómata dado es el autómata del conjunto cociente, A'.

- A' es equivalente a A
- A' es mínimo, si hay otro autómata A'' equivalente a A, entonces A'' tiene un número de estados mayor o igual que el autómata cociente.
- A' es único salvo isomorfismo.

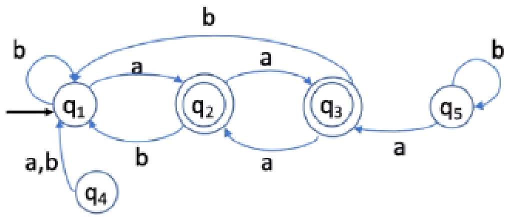
Algoritmo para construir Autómata Mínimo:

1. Se construye el autómata conexo (eliminando estados no accesibles)
2. Se construye P_E = Q/E del autómata conexo
3. A' = <∑, Q', f ', [q_0'], F'> donde:
   - Q' = Q/E
   - [q_0'] = c_0 si q_0 ∈ c_0
   - F' = {c / c contiene al menos un estado de F (es decir, existe un q∈F, tal que q∈c)}

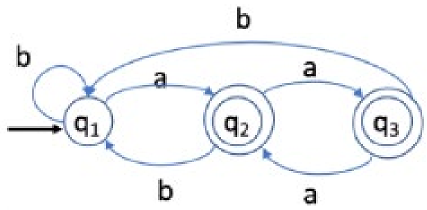
     F' = {[q] ∈Q/E / q ∈F}
   - f '(c_i, e) = c_j si existen q∈c_i, p∈c_j, tales que f(q, a) = p

     f ' ([q], a) = [f(q,a)]

Construir el autómata mínimo para el autómata:



Primero eliminamos los estados no accesibles por lo que el autómata conexo equivalente quedaría:



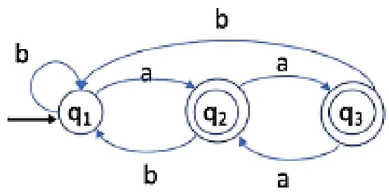
- Se calcula P_E
  1. Definimos P_0
     - $P_0 = \{ A_1 = [q1] , A_2 = [q2, q3]\}$
  2. Calcular P_1

     Para los estados de A_2:

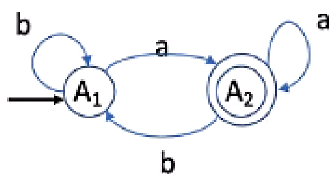
     $f(q_2, a) = q_3 \in A_2$ $\quad$ $f(q_2, b) = q1 \in A_1$

     $f(q_3, a) = q_2 \in A_2$ $\quad$ $f(q_3, b) = q1 \in A_1$

     - Se obtiene que $P_1 = P_0 = P_E$



- Se obtiene $A' = <\Sigma, Q', f', q_0', F'>$ donde:
  - $\Sigma = \{a, b\}$
  - $Q' = Q/E = \{ A_1 , A_2 \}$
  - $q_0' = [q_1] = A_1$
  - $F' = \{A_2\}$
  - f ':
    - $f'(A_1, a) = A_2$ $f'(A_1, b) = A_1$
    - $f'(A_2, a) = A_2$ $f'(A_2, b) = A_1$

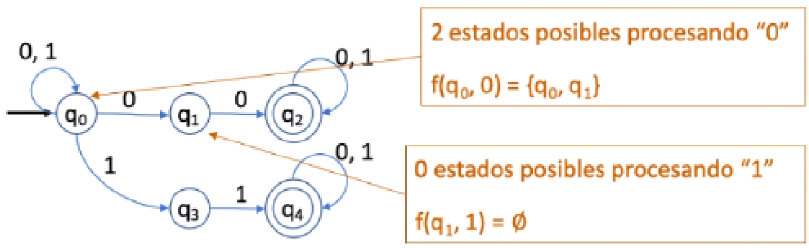


AUTÓMATA FINITO NO DETERMINISTA: Es un autómata que a diferencia de los AFD tiene más de una o no tiene ninguna transición posible para un mismo símbolo y además, acepta la transición para la palabra vacía. Se puede dar que:

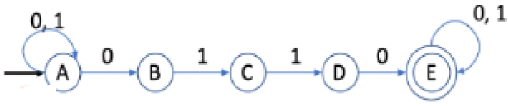
$$f(q, a) = q_1 \ \& \ f(q, a) = q_2 \text{ siendo } q_1 \neq q_2 \qquad f(q, b) = \lambda$$

Un AFND es una séxtupla del tipo A = { Σ, Q, f, q_0, F, T } donde T es el conjunto de transiciones entre estados mediante el símbolo λ. Algunos ejemplos:

- L = { x ∈ {0, 1}* / x contiene 2 ceros o 2 unos consecutivos}



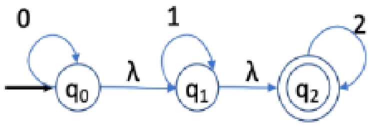
- L = { x ∈ {0, 1}* / x contenga como subpalabra 0110}



0 estados posibles procesando "0"

f(B, 0) = ∅

- L = 0* 1* 2*



- Sea el AFND = {∑ = {a, b}, Q = {q_1, q_2, q_3, q_4}, f , q_1, F ={q_1, q_4}, T} siendo:
  - f:

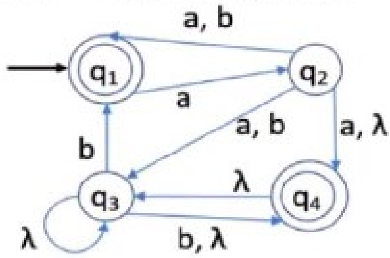
    f(q1, a) = q2 f(q1, b)= ∅

    f(q2, a) = {q1, q3, q4} f(q2, b)= {q1, q3}

    f(q3, a) = ∅ f(q3, b)= {q1, q4}

    f(q4, a) = ∅ f(q4, b)= ∅
  - T = {(q2, q4), (q3, q4), (q3, q3), (q4, q3)}
- Este AFND puede representarse cómo una tabla de transición o cómo diagrama:

	a	b	λ
→(q1)	q2	∅	∅
q2	{q1, q3, q4}	{q1, q3}	q4
q3	∅	{q1, q4}	{q3, q4}
(q4)	∅	∅	q3



Cierre Transitivo (T⁺) : T⁺ = $\sum_{i=1}^{\infty} T^i$; (p, q) ∈ T⁺ si 'q' es accesible desde 'p' por medio de transiciones de λ.

Obtendremos la matriz booleana correspondiente a T⁺ a partir de la de T. Para el ejemplo anterior:

T	q1	q2	q3	q4
q1	0	0	0	0
q2	0	0	0	1
q3	0	0	1	1
q4	0	0	1	0

T = {(q2, q4), (q3, q4), (q3, q3), (q4, q3)}

La matriz de T⁺ resulta ser:

T⁺	q1	q2	q3	q4
q1	0	0	0	0
q2	0	0	1	1
q3	0	0	1	1
q4	0	0	1	1

T⁺ = {**(q2, q3)**(q2, q4), (q3, q4), (q3, q3), (q4, q3), **(q4, q4)**}

Definiciones:

Cierre de Transiciones: λ – clausura o E (q) = {conjunto de todos los estados p / hay un camino de q a p etiquetado con λ, incluyendo q}

f'': extensión de f a palabra: f'' Q x Σ^* → 2Q

a) f'' (q, λ) = λ – clausura o E (q)
b) f'' (q, xa) = E($\cup_{p \in f''(q, x)}$ f (p, a))

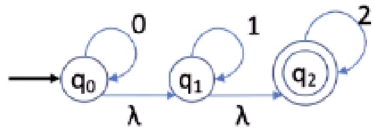
f'': en un conjunto de P de estados: f'' (P, x) = $\cup_{q \in P}$ f''(q, x)

Extensión a la λ – Clausura a un conjunto de estados P:

$$\lambda \text{ – clausura } P = E(P) = \cup_{q \in P} E(q)$$

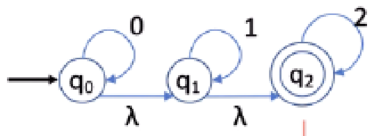
- Sea el AFND = (Σ = {0, 1, 2}, Q = {q_0, q_1, q_2}, f , q_0, F ={q_2}, T={(q_0, q_1), (q_1, q_2)})

- L(A) = 0*1*2*
  - Secuencia para aceptar "002": $q_0q_0q_0q_1q_2q_2$
  - f '' (q_0, 002) = f '' (q_0, 00λλ2) = q_2 (final)



f	0	1	2	λ
→q_0	q_0	Ø	Ø	q_1
q_1	Ø	q_1	Ø	q_2
(q_2)	Ø	Ø	q_2	Ø

- Cálculo de λ – clausuras
  - E(q_0) = λ – clausura de q_0 = {q_0, q_1, q_2} → f '' (q_0, λ) = {q_0, q_1, q_2}
  - E(q_1) = {q_1, q_2}
  - E(q_2) = {q_2}



f	0	1	2	λ
→q_0	q_0	Ø	Ø	q_1
q_1	Ø	q_1	Ø	q_2
(q_2)	Ø	Ø	q_2	Ø

Definición f'' a) f '' (q, λ) = λ -clausura q o E(q)
b) f '' (q, xa) = E($\cup_{p \in f''(q,x)} f(p, a)$)

f '' (q_0, 0) = E (f (f ''(q_0, λ), 0)) =

E(f ({q_0, q_1, q_2}, 0))=

E(f (q_0, 0) ∪ f(q_1, 0) ∪ f(q_2, 0)) =

E({q_0} ∪ Ø ∪ Ø) = E(q_0) =

{q_0, q_1, q_2} (≠ **f(q_0,0)**)

f '' (q_0, 01) = E(f (f ''(q_0, 0), 1)) =

E(f ({q_0, q_1, q_2}), 1) =

E(f (q_0, 1) ∪ f(q_1, 1) ∪ f(q_2, 1)) =

E(Ø ∪ {q_1} ∪ Ø) = E({q_1}) = {q_1, q_2}

Lenguaje aceptado por un AFND: la palabra x es aceptada por un AFND si f''(q0, x) y F tienen al menos un elemento común.
Siendo A un AFND, L (A) es el conjunto de todas las palabras aceptadas por A.

Simulación de un AFND por un AFD: dado un AFND siempre es posible construir otro AFD que acepta el mismo lenguaje. El conjunto de lenguajes aceptados por los AFND es igual al conjunto de lenguajes aceptados por los AFD.

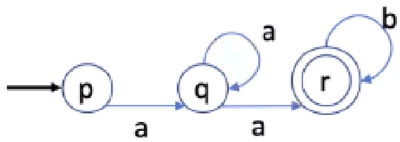
Sea el AFND A = { Σ, Q, f, q_0, F, T } definiremos a partir de él:

AFD B = { Σ, Q', f', q'_0, F' } donde:

- $Q' = 2^Q$ = P (Q) = el conjunto de las partes de Q
- $q'_0 = f''(q0, \lambda)$
- $F' = \{ c \,/\, c \in Q' \wedge \exists q \in c \,/\, q \in F \}$
- $f' = f'(c, a) = \{c' \,/\, c' = \cup_{q \in c} f''(q, a)$

Dado el AFND, calcular el AFD equivalente

AFND = { ∑={a, b}, Q = {p, q, r}, f, p, {r}, T = Ø }



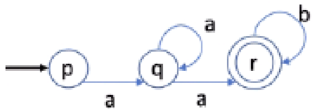
f	a	b
→p	q	
q	q,r	
(r)		r

El AFD buscado viene dado por: { ∑={a, b}, Q', f', f''(p, λ), F' }

P(Q) = {Ø, {p}, {q}, {r}, {p,q}, {p,r}, {q,r}, {p,q,r}} (puede que no se necesiten todos en Q')

$f' = f'(c, a) = \{c' / c' = \cup_{q \in c} f''(q, a)\}$
$f''(q, xa) = E(\cup_{p \in f''(q,x)} f(p, a))$

1. Calculo el nuevo estado inicial f''(p, λ) = {p}
2. Construir f'
   - **f' (p, a)** = f '' (p, a) = E (f(f''(p, λ) ,a)) = E (f (p ,a)) = E (q) = **{q}**
   - **f' (q, a)** = f '' (q, a) = E (f(f''(q, λ) ,a)) = E (f(q ,a)) = E({q,r}) = E(q) ∪ E(q) = **{q, r}**
   - **f' ({q, r}, a)** = f '' (q, a) ∪ f '' (r, a) = E (f (f''(q, λ) ,a)) ∪ E (f (f''(r, λ) ,a)) = E (f (q ,a)) ∪ E (f (r ,a)) = E ({q, r}) ∪ E (Ø) = **{q, r}**
   - **f' ({q, r}, b)** = f '' (q, b) ∪ f '' (r, b) = E (f (f''(q, λ) ,b)) ∪ E (f (f''(r, λ) ,b)) = E (f (q ,b)) ∪ E (f (r ,b)) = E (Ø) ∪ E ({r}) = **{r}**
   - **f' (r, b)** = f '' (r, b) = E (f(f''(r, λ) , b)) = E (f (r , b)) = E (r) = **{r}**



f'	a	b
→{p}	{q}	Ø
{q}	{q,r}	Ø
({q, r})	{q,r}	{r}
({r})	Ø	{r}
Ø	Ø	Ø