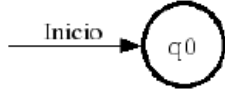
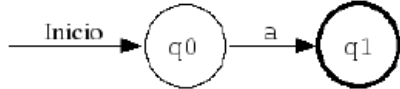
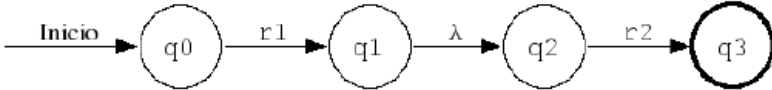
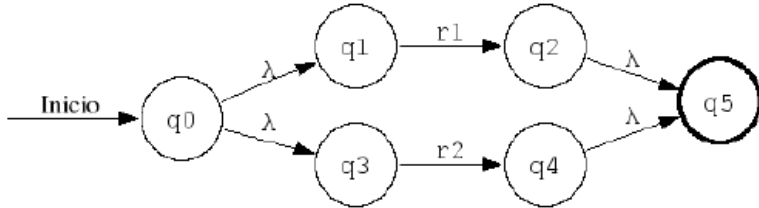
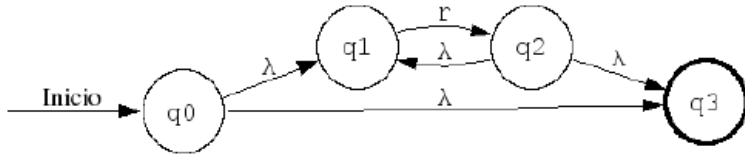


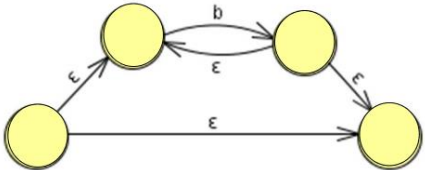
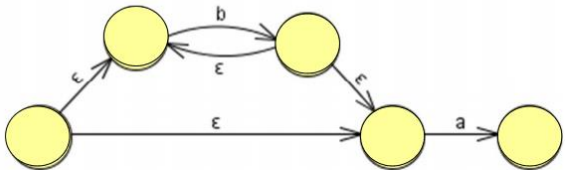
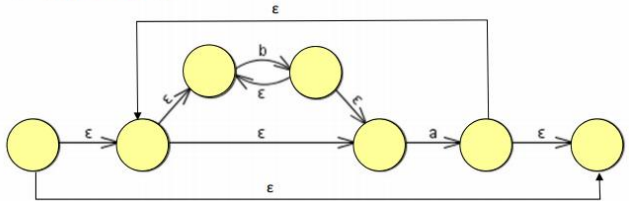
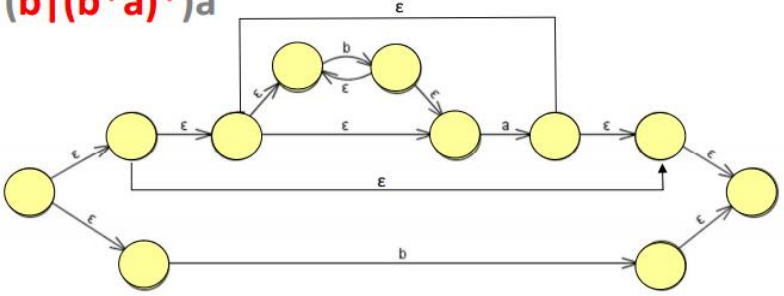
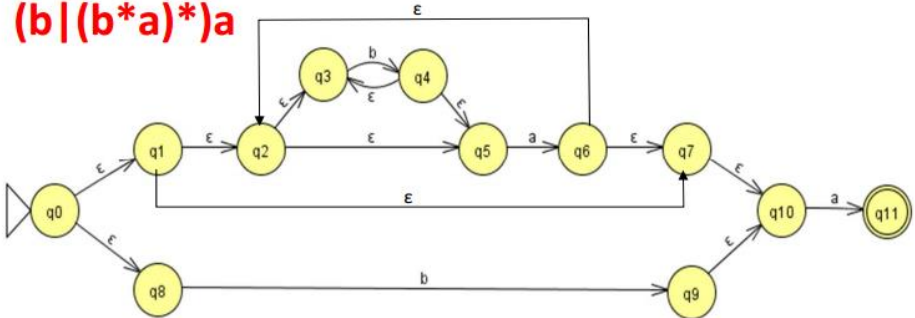
**TALLER AUTOMATAS**  
**UNIVERSIDAD DE CALDAS**  
**CARLOS GUTIERREZ**

El algoritmo para convertir expresiones regulares a AFNDs opera construyendo recursivamente el AFND en base a las expresiones regulares particulares. Los AFND a construir son los siguientes

Expresión Regular	Autómata Finito No Determinístico Generado
$r = \lambda$	
$r = a$	
$r = r_1 r_2$	
$r = r_1   r_2$	
$r = r_1^*$	

**TALLER AUTOMATAS**  
**UNIVERSIDAD DE CALDAS**  
**CARLOS GUTIERREZ**

Ejemplo A partir de la ER  $(b|(b^*a)^*)a$  a construir al AFND

Primero $b^*$	$(b (b^*a)^*)a$ 
Luego la concatenación con a	$(b (b^*a)^*)a$ 
Continuamos con el *	$(b (b^*a)^*)a$ 
Sigue el   o la elección de alternativas	$(b (b^*a)^*)a$ 
Concatenamos el ultimo simbolo	$(b (b^*a)^*)a$ 

## Conversión de un AFND a un AFD

### Definiciones

**Cerradura épsilon:** Dado un AFN definimos la operación cerradura épsilon de un estado X como:

$$C_{\epsilon}(x) = \{x\} \cup \{v \mid v \text{ es alcanzable con transiciones } \epsilon \text{ a partir de } x\}$$

### Operación Mover:

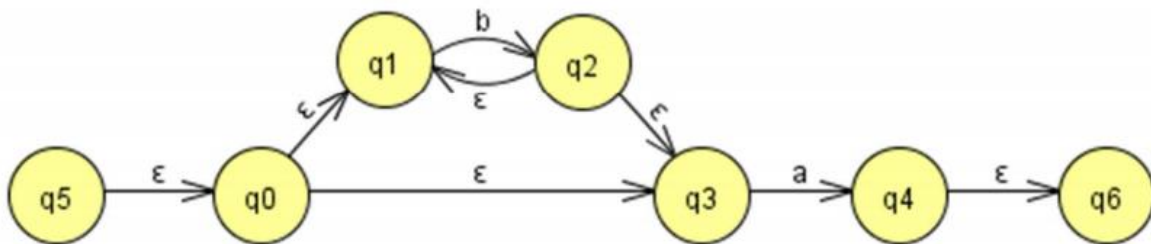
$$\text{Mover}(e, \alpha) = \{\beta \mid \exists \text{ una transición de } e \text{ con } \alpha \text{ hacia el estado } \beta\}$$

### Operación Ir\_A:

$\text{Ir}_A(C, \alpha)$  donde C es un conjunto  $\{e_1, e_2, e_3, \dots, e_n\}$  de estados del AFN y  $\alpha$  es un símbolo del alfabeto del mismo AFN.

$$\text{Ir}_A(C, \alpha) = C_{\epsilon}(\text{Mover}(C, \alpha))$$

### Ejemplo



$$C_{\epsilon}(q_5) = \{q_5\} \cup \{q_0, q_1, q_3\}$$

$$C_{\epsilon}(q_2) = \{q_2\} \cup \{q_3, q_1\}$$

$$C_{\epsilon}(q_1) = \{q_1\} \cup \{\lambda\}$$

$$C_{\epsilon}(\{q_3, q_4\}) = \{q_3, q_4, q_6\}$$

$$\text{Mover}(q_3, a) = \{q_4\}$$

$$\text{Mover}(q_0, a) = \{\lambda\}$$

$$\text{Mover}(\{q_5, q_1\}, b) = \{q_2\}$$

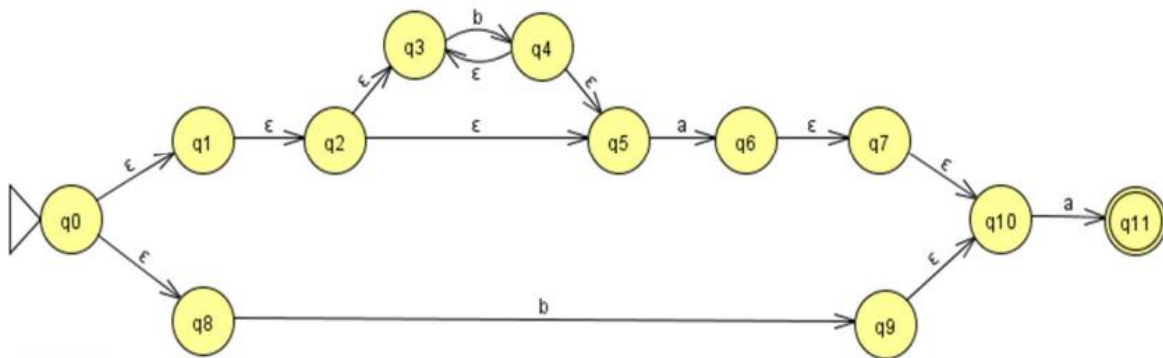
**TALLER AUTOMATAS**  
**UNIVERSIDAD DE CALDAS**  
**CARLOS GUTIERREZ**

**Algoritmo de conversión de un AFN a un AFD**

1. Se calcula la  $C_\epsilon$  del estado inicial del AFND, el resultado será el estado inicial  $S_0$  del AFD ( $S_0$  será el estado inicial del AFD y el primer  $S_i$  del AFD.)
  2. Se calcula para cada  $S_i$  la operación  $Ir\_A$  para cada  $\alpha \in \Sigma$ , la cual arrojará un estado  $S_i$  (Pudiendo repetirse).
  3. Se realiza la operación 2 con todos los estados hasta que ya no surjan estados diferentes.
- El estado inicial del AFD será  $S_0$  y los estados finales serán todos aquellos  $S_i$  que contengan al estado final del AFN original.
  - La función de transición es el resultado de todas las operaciones  $Ir\_A$  sobre los  $S_i$ .

**Ejemplo**

Convertir el autómata finito no determinista de la expresión regular  $(b|b^*a)a$  a un autómata finito determinista.



Se calcula la cerradura épsilon del estado inicial

$$C_\epsilon(q_0) = \{q_0, q_1, q_2, q_3, q_5, q_8\} = A \quad \text{Alfabeto } \{a, b\}$$

Se calcula para cada  $S_i$  la operación  $Ir\_A$  para cada  $\alpha \in \Sigma$

$$Ir\_A(A, a) = C_\epsilon(\text{Mover}(A, a)) = C_\epsilon\{q_6\} = \{q_6, q_7, q_{10}\} = B$$

$$Ir\_A(A, b) = C_\epsilon(\text{Mover}(A, b)) = C_\epsilon\{q_4, q_9\} = \{q_4, q_3, q_5, q_9, q_{10}\} = C$$

$$Ir\_A(B, a) = C_\epsilon(\text{Mover}(B, a)) = C_\epsilon\{q_{11}\} = \{q_{11}\} = D$$

$$Ir\_A(B, b) = C_\epsilon(\text{Mover}(B, b)) = C_\epsilon\{\lambda\} = \{\lambda\}$$

$$Ir\_A(C, a) = C_\epsilon(\text{Mover}(C, a)) = C_\epsilon\{q_6, q_{11}\} = \{q_6, q_7, q_{10}, q_{11}\} = E$$

$$Ir\_A(C, b) = C_\epsilon(\text{Mover}(C, b)) = C_\epsilon\{q_4\} = \{q_4, q_3, q_5\} = F$$

**TALLER AUTOMATAS**  
**UNIVERSIDAD DE CALDAS**  
**CARLOS GUTIERREZ**

$Ir\_A(D,a) = C\_ \epsilon (Mover(D,a)) = C\_ \epsilon \{\lambda\} = \{\lambda\}$   
 $Ir\_A(D,b) = C\_ \epsilon (Mover(D,b)) = C\_ \epsilon \{\lambda\} = \{\lambda\}$

$Ir\_A(E,a) = C\_ \epsilon (Mover(E,a)) = C\_ \epsilon \{q11\} = \{q11\} = D$   
 $Ir\_A(E,b) = C\_ \epsilon (Mover(E,b)) = C\_ \epsilon \{\lambda\} = \{\lambda\}$

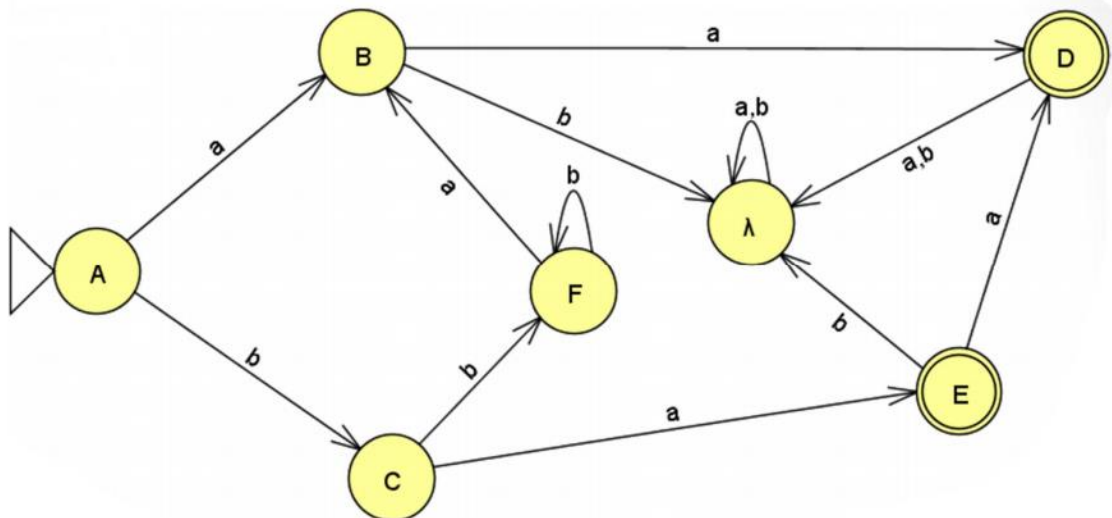
$Ir\_A(F,a) = C\_ \epsilon (Mover(F,a)) = C\_ \epsilon \{q6\} = \{q6,q7,q10\} = B$   
 $Ir\_A(F,b) = C\_ \epsilon (Mover(F,b)) = C\_ \epsilon \{q4\} = \{q4,q3,q5\} = F$

- El estado inicial es A, ya que originalmente contiene a q0.
- Los estados finales son D y E ya que contienen a q11.

Tabla de transiciones

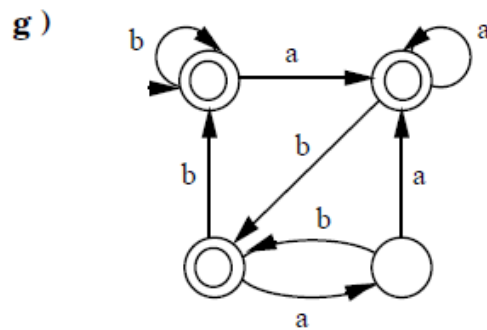
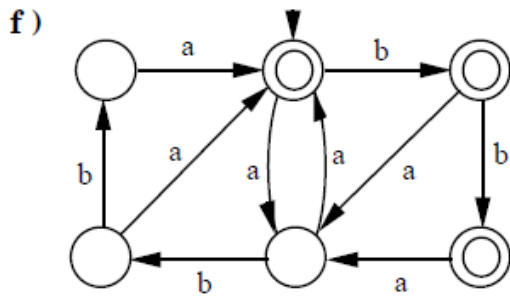
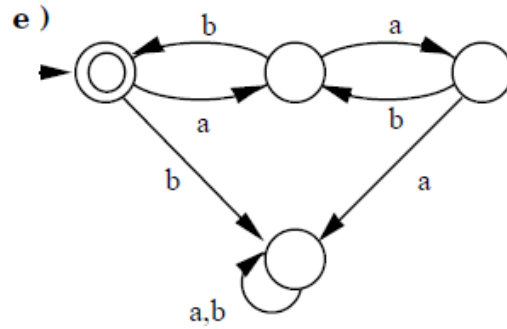
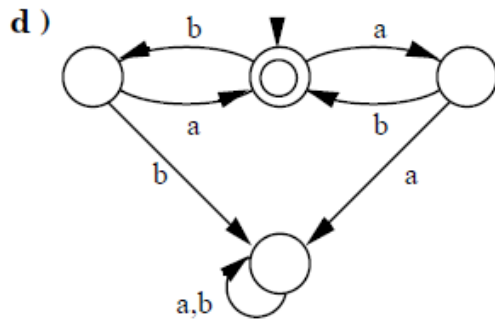
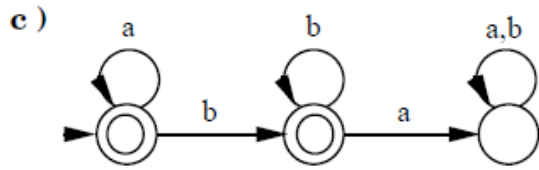
f	a	b
A	B	C
B	D	$\lambda$
C	E	F
D	$\lambda$	$\lambda$
E	D	$\lambda$
F	B	F

**$(b|b^*a)a$**





**TALLER AUTOMATAS**  
**UNIVERSIDAD DE CALDAS**  
**CARLOS GUTIERREZ**



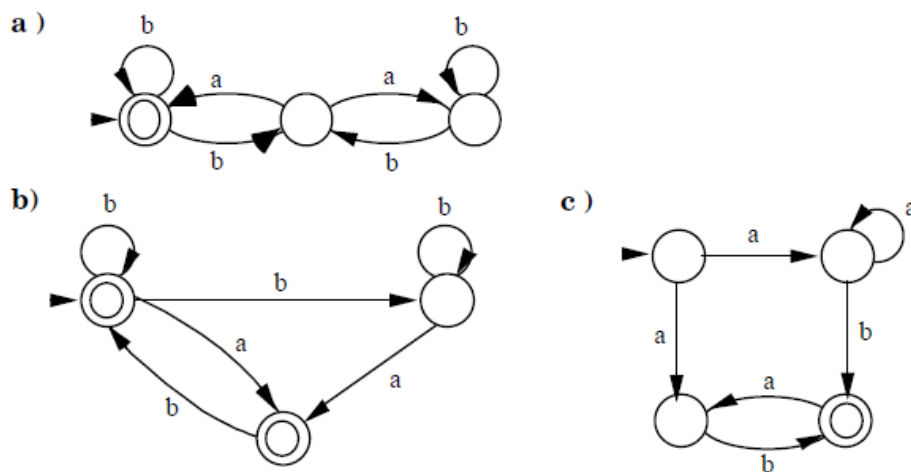
4. Construye AFDs que acepten cada uno de los siguientes lenguajes definidos sobre el alfabeto  $\Sigma = \{a, b\}$ :
- a)  $L = \{x \in \Sigma^* : \text{la longitud de } x \text{ es divisible por } 3\}$
  - b)  $L = \{x \in \Sigma^* : \text{aba no es subpalabra de } x\}$
  - c)  $L = \{x \in \Sigma^* : x \text{ comienza por } a \text{ y termina por } ab\}$
  - d)  $L = \{x \in \Sigma^* : x \text{ tiene un n\u00famero par de } a\text{'s y un n\u00famero par de } b\text{'s}\}$
  - e)  $L = \{x \in \Sigma^* : x \text{ tiene tres } a\text{'s consecutivas}\}$
  - f)  $L = \{x \in \Sigma^* : \text{toda aparici\u00f3n de la subpalabra } aba \text{ en } x, \text{ o bien va seguida de } bb, \text{ o est\u00e1 al final de la palabra}\}$
  - g)  $L = \{x \in \Sigma^* : \text{si } x \text{ empieza por } a \text{ no contiene la subpalabra } aa \text{ y si } x \text{ empieza por } b \text{ contiene la subpalabra } aa\}$
  - h)  $L = \{x \in \Sigma^* : x \text{ tiene un n\u00famero par de apariciones de la cadena } ab\}$
  - i)  $L = \{x \in \Sigma^* : ab \text{ es subpalabra de } x \text{ si y s\u00f3lo si } ba \text{ es subpalabra de } x\}$

**TALLER AUTOMATAS**  
**UNIVERSIDAD DE CALDAS**  
**CARLOS GUTIERREZ**

- j)  $L = \{ x \in \Sigma^* : x \text{ está formada por la concatenación de un número arbitrario de cadenas de la forma } yyR, \text{ con } |y|=2 \}$
- k)  $L = \{ x \in \Sigma^* : x \text{ no contiene ningún prefijo en el que la diferencia entre el número de a's y b's sea mayor que tres (a favor de cualquiera de ellos)} \}$
- l) Construye un AFD que acepte el siguiente lenguaje:  
 $L = \{ x \in \{a, b\}^* : x \text{ no contiene la subpalabra } bab \text{ ni el sufijo } aba \}$

*Sobre AFNDs (autómatas finitos no deterministas):*

7. Busca tres palabras aceptadas y tres palabras rechazadas por cada uno de los siguientes autómatas no deterministas, mostrando todos los cómputos que las procesan. ¿Sabrías qué lenguaje acepta cada uno de ellos?



Construye AF que acepten los siguientes lenguajes sobre el alfabeto

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

- a)  $L = \{ x \in \Sigma^* : x \text{ tiene algún par de a's separadas por una cadena de símbolos de longitud } 4*i, \text{ con } i \geq 0 \}$
- b)  $L = \{ x \in \Sigma^* : |x| \geq 5 \text{ y el quinto símbolo contado desde el final es } a \}$
- c)  $L = \{ x \in \Sigma^* : \text{ni } aa \text{ ni } bb \text{ son subcadenas de } x \}$
- d)  $L = \{ x \in \Sigma^* : x \text{ tiene } ab \text{ y } ba \text{ como subcadena} \}$
- e)  $L = \{ x \in \Sigma^* : ccc \text{ es sufijo de } x \text{ y en ninguna otra posición de } x \text{ pueden encontrarse dos símbolos iguales seguidos} \}$
- f)  $L = \{ x \in \Sigma^* : x \text{ tiene tres símbolos iguales seguidos} \}$
- g)  $L = \{ x \in \Sigma^* : \text{los símbolos de } x \text{ de posición múltiplo de tres son c's} \}$



**TALLER AUTOMATAS**  
**UNIVERSIDAD DE CALDAS**  
**CARLOS GUTIERREZ**

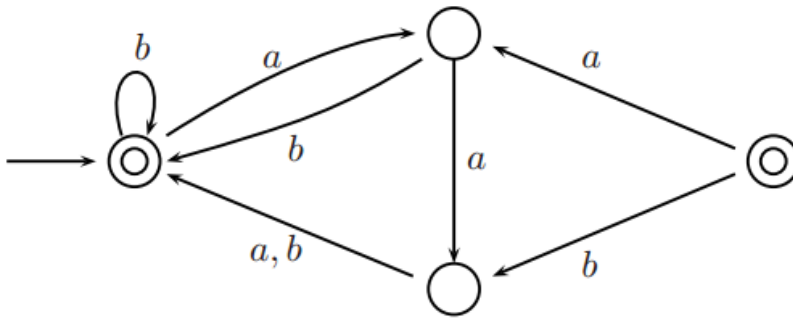
Construye patrones que generen cada uno de los siguientes lenguajes sobre el alfabeto terminal  $\Sigma = \{a, b, c\}$

- a)  $\{x \in \Sigma^* : \text{cantidad de } a\text{'s mod } 2 = 0\}$
- b)  $\{x \in \Sigma^* : x \text{ empieza por } a \text{ y contiene la subpalabra } bbb\}$
- c)  $\{x \in \Sigma^* : x \text{ no contiene tres } b\text{'s consecutivas}\}$
- d)  $\{x \in \Sigma^* : \text{cantidad de } a\text{'s mas } b\text{'s mod } 3 = 0\}$
- e)  $\{x \in \Sigma^* : x \text{ contiene las subpalabras } aa, bb \text{ y } cc\}$
- f)  $\{x \in \Sigma^* : x \text{ no contiene las subpalabras } aba \text{ ni } aca\}$
- g)  $\{x \in \Sigma^* : \text{cada aparición de } a \text{ en } x \text{ es precedida de } b \text{ o seguida de } c\}$
- h)  $\{x \in \Sigma^* : x \text{ no contiene ninguna aparición de la subcadena } aa \text{ después de la última aparición del símbolo } c\}$
- i)  $\{x \in \Sigma^* : x \text{ contiene dos } a\text{'s separadas por un número impar de símbolos}\}$
- j)  $\{x \in \Sigma^* : x \text{ contiene la subpalabra } ab \text{ pero no contiene la subpalabra } aba\}$

otros

- a)  $L = \{w | w \text{ tiene un número par de } a\text{'s que terminan en } b\}$
  - b)  $L = \{w | w \text{ tiene un número impar de } a\text{'s terminando en } a\}$
  - c)  $L = \{w | w \text{ tiene un número múltiplo de 3 de } a\text{'s}\}$
  - d)  $L = \{w | \text{ toda } a \text{ en } w \text{ está entre dos } b\text{'s}\}$
  - e)  $L = \{w | \text{ no hay dos } a\text{'s consecutivas en } w\}$
  - f)  $L = \{w | w \text{ no contiene la subcadena } aa \text{ ni } bb\}$
  - g)  $L = \{w | w \text{ contiene la subsecuencia } ab\}$
  - h)  $L = \{w | w \text{ contiene la subsecuencia } aacbb\}$
  - i)  $L = \{w | w \text{ contiene la subsecuencia } aba\}$
  - j)  $L = \{w | w \text{ contiene la subcadena } aba\}$
- 
- a)  $L = \{w | w \text{ contiene un número impar de } a\text{'s}\}$
  - b)  $L = \{w | w \text{ contiene un número par de } b\text{'s}\}$
  - c)  $L = \{w | w \text{ contiene un número par de } a\text{'s y un número par de } b\text{'s}\}$
  - c)  $L = \{w | w \text{ contiene un número impar de } a\text{'s y un número impar de } b\text{'s}\}$
  - d)  $L = \{w | w \text{ contiene un } ab \text{ o } ba \text{ como subcadenas}\}$
  - e)  $L = \{w | w \text{ contiene un } ab \text{ y } ba \text{ como subcadenas}\}$
  - f)  $L = \{w | w \text{ contiene un } ab \text{ ó } ba \text{ como subcadenas, pero no ambas}\}$
  - g)  $L = \{w | w \text{ contiene un número par de } a\text{'s y un número impar de } b\text{'s}\}$

**TALLER AUTOMATAS**  
**UNIVERSIDAD DE CALDAS**  
**CARLOS GUTIERREZ**



Determinar si las siguientes palabras o cadenas del alfabeto  $\{a,b\}$  son aceptadas por el autómata anterior usando la función de transición extendida

a) bab   b) aaba   c) aaabaaab   d) babababab   e) aabbb   f) aaaba   g) babababa   h) a  
i) babba   j) baaabbb   k) aaa   l) ababbbbaa

2. Determinar si las siguientes expresiones regulares son o no aceptadas por el autómata anterior

a)  $a^*$    b)  $aa^*$    c)  $ab^*$    d)  $(ab)^*$    e)  $a^*b^*$    f)  $a(a|b)^*$    g)  $b^*$    h)  $(ab)^*b$    i)  $(aa(a|b))^*$    j)  $b^*(aaa)^*(ab)^*$

3. Hallar un autómata determinista que reconozca las siguientes expresiones regulares sobre el alfabeto  $\Sigma = \{x, y\}$

a)  $xyxxy$    b)  $xy+x^*$    d)  $x(yx)^*y$    e)  $(x|y)(yx|xyx)^*$    f)  $(x|y)(yx|yxy)$    g)  $(x|y)+y?$    h)  $x+y^*x?$    i)  $x+y+x$    j)  $(xx)+(yy)?x+$

4. Describa con sus palabras el lenguaje representado por las siguientes expresiones regulares

a.  $((z|y)^*x)$    b.  $((xx^*)(yy^*))$    c.  $((xx^*)|(yy^*))$    d.  $((x^*y^*)z^*)$

Realizar los autómatas finitos deterministas y representarlos en las tres notaciones

- a- Reconocer números binarios múltiplos de tres
- b- Reconocer números binarios de máximo 8 bits
- c- Reconocer números binarios de máximo 8 bits sin ceros a la izquierda
- d- Reconocer números binarios de máximo 8 bits con ceros a la izquierda

- Modelar un AFD para una maquina expendedora de café, teniendo en cuenta que solo recibe monedas de 100, 200, 500 y 1000, y el café vale 800 pesos.
- Modelar un AFD para una llamada telefónica entre dos personas
- Modelar con un autómata un cajero electrónico solo con las operaciones de retirar

**TALLER AUTOMATAS**  
**UNIVERSIDAD DE CALDAS**  
**CARLOS GUTIERREZ**

- Modelar un AFD para el funcionamiento de un ascensor
17. Sea  $\Sigma = \{a,b\}$ . Generar la ER, el AFND de la forma larga y con el mínimo numero de transiciones lambda, convertir a AFD y minimizar por los dos métodos.:
- a ) cadenas con al menos tres **a**'s
  - b ) cadenas con a lo sumo tres **a**'s
  - c ) cadenas con un número de **a**'s divisible por 3
  - d ) cadenas con al menos una aparición de la subcadena **aaa**
  - e ) cadenas en las que las **a**'s van agrupadas como mínimo de tres en tres.
- 18 Generar la ER, el AFND de la forma larga y con el mínimo numero de transiciones lambda, convertir a AFD y minimizar por los dos métodos.:
- a )  $\{ w \in \{a,b\}^* \mid w \text{ acaba en } ab \}$
  - b )  $\{ w \in \{a,b\}^* : |w| \neq 1 \}$  .
  - c )  $\{ w \in \{a,b\}^* : w \text{ tiene un número par de } a\text{'s y termina por } ab \}$
  - d )  $\{ w \in \{1,0\}^* : w \text{ no contiene } 101 \text{ como subpalabra} \}$
  - e )  $\{ w \in \{1,0\}^* : w \text{ comienza por } 101 \text{ y termina por } 101 \}$
  - f )  $\{ w \in \{a,b,c\}^* : \text{entre cada dos } a\text{'s de } w \text{ hay un número de } c\text{'s múltiplo de 3} \}$

Generar el AFND de la forma larga y con el mínimo número de transiciones lambda, para las siguientes expresiones regulares, convertir a AFD y minimizar por los dos métodos. Alfabeto= $\{a,b,c\}$

- $a^*b$
- $(abc)^*$
- $(b|bc)^+$
- $(a|b)^*abb$
- $((b|b^*a)^*)a$
- $(a^*|b^+)^+$
- $(a|b)^*abb$
- $(a^+b^+)(b^*a^+)?$
- $(a^+b^+c^*)(cc)?a^+$

- $(a(bb)^+c^*)b?$

Generar la ER, el el AFND de la forma larga y con el mínimo numero de transiciones lambda, convertir a AFD y minimizar por los dos métodos.  
Alfabeto={a,b,c}

- 1- Palabras con numero par de c's sin importar la ubicación
- 2- Palabras que tengan siempre dos b's seguidas luego de una a
- 3- Palabras que no tienen dos consonantes juntas
- 4- Palabras que no tienen dos consonantes ni dos vocales juntas
- 5- Palabras que empiezan siempre por a's y terminan en una b y n
- 6- gramatica para aceptar palabras que empiecen por dos símbolos del alfabeto y que termine en los mismos símbolos de manera inversa
- 7- gramatica que acepte un numero impar de a's o numero múltiplo de tres de b's
- 8- gramatica que acepte palabras de tal forma que la vocal siempre este entre dos consonantes
- 9- gramatica con numero impar de simbolos
- 10- Palabras con numero par de a's al comienzo y seguidas por un numero impar de b's mas c's al final