Autómatas Finitos Deterministas

Un AFD es $(Q, T, \delta: Q \times T \to Q,\ q_0 \in Q, F \subseteq Q)$
que "acepta" la palabra $u \in T^*$ si
$$\delta'(q_0, u) = \delta(\delta(\ldots\delta(q_0, u_1), u_2), \ldots), u_n) \in F.$$

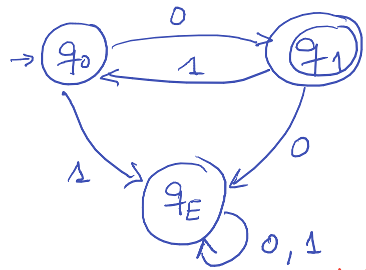
3. $G = (\{S\}, \{0,1\}, \{S \to S10, S \to 0\}, S)$

$L = \{0, 010, 01010, \ldots\}$

Autómata F.D.: (Q, A, δ, q_0, F)

$Q = \{q_0, q_1, q_E\}$ $A = \{0, 1\}$, $F = \{q_1\}$

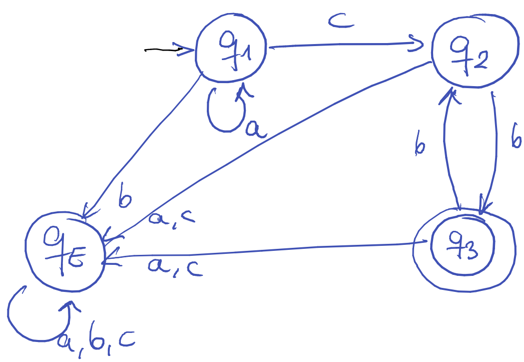
δ:



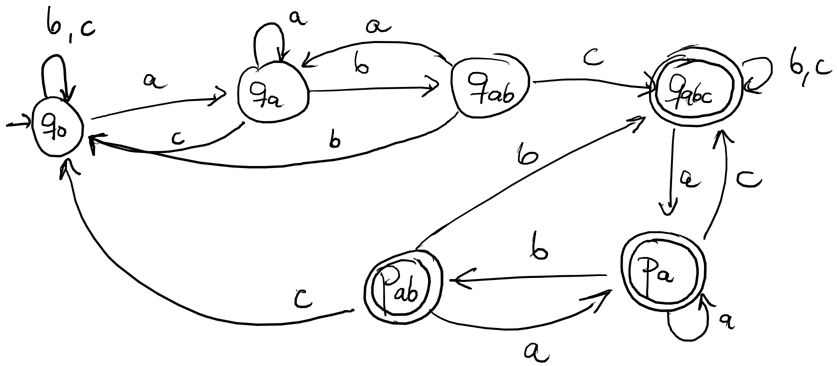
Todas las transiciones deben estar definidas, de ahí la necesidad de q_E.

11. $S \to AB \quad A \to aA \mid c \quad B \to bBb \mid b$

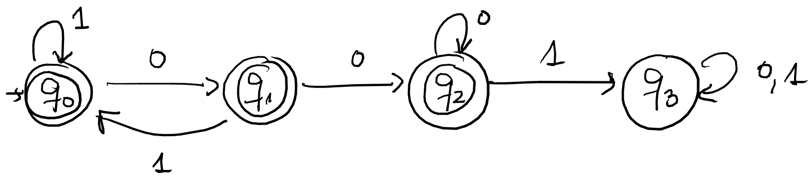
$L = \{a^n c b^m \ /\ n \geq 0, m \text{ es impar}\}$



14. $\{ u \in \{a,b,c\}^* \ / \ $ "abc" aparece un nº impar de veces$\}$



15. Palabras en las que no aparece "001".



- Números racionales

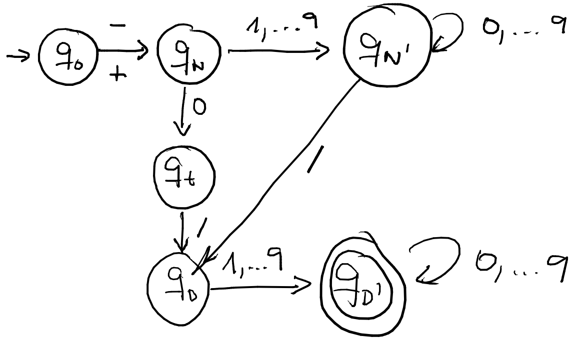
$S \to -N \mid +N$

$N \to 0/D \mid 1N' \mid \dots \mid 9N'$

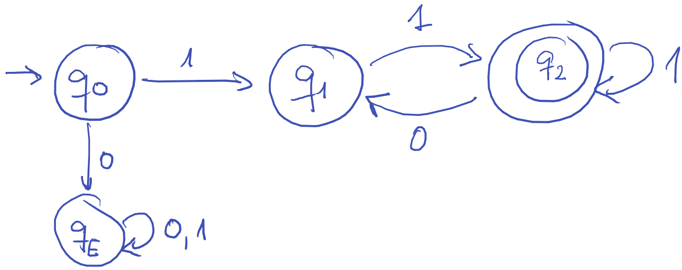
$N' \to 0N' \mid N \mid /D$

$D \to 1D' \mid \dots \mid 9D'$

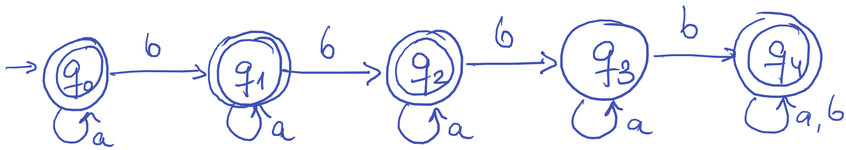
$D' \to 0D' \mid D \mid \varepsilon$



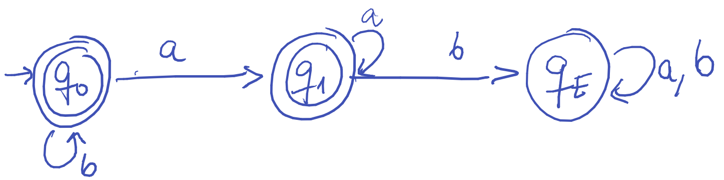
R1.7. $L = \{1u1 \,/\, u \in \{0,1\}^*\}$



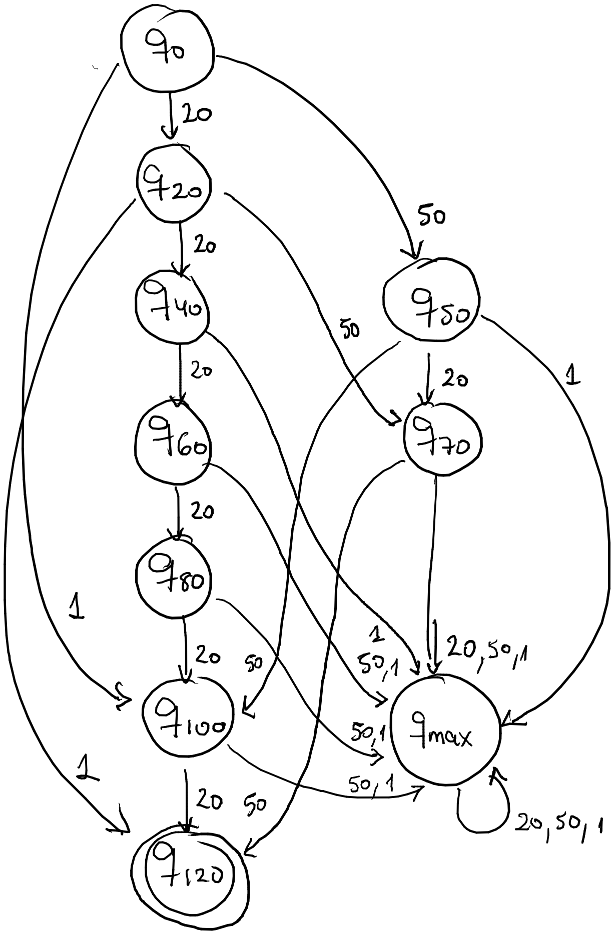
R1.6.i. $N_b(u) \neq 3$

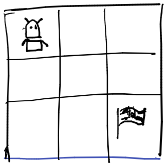


iii. "ab" no está contenido en u



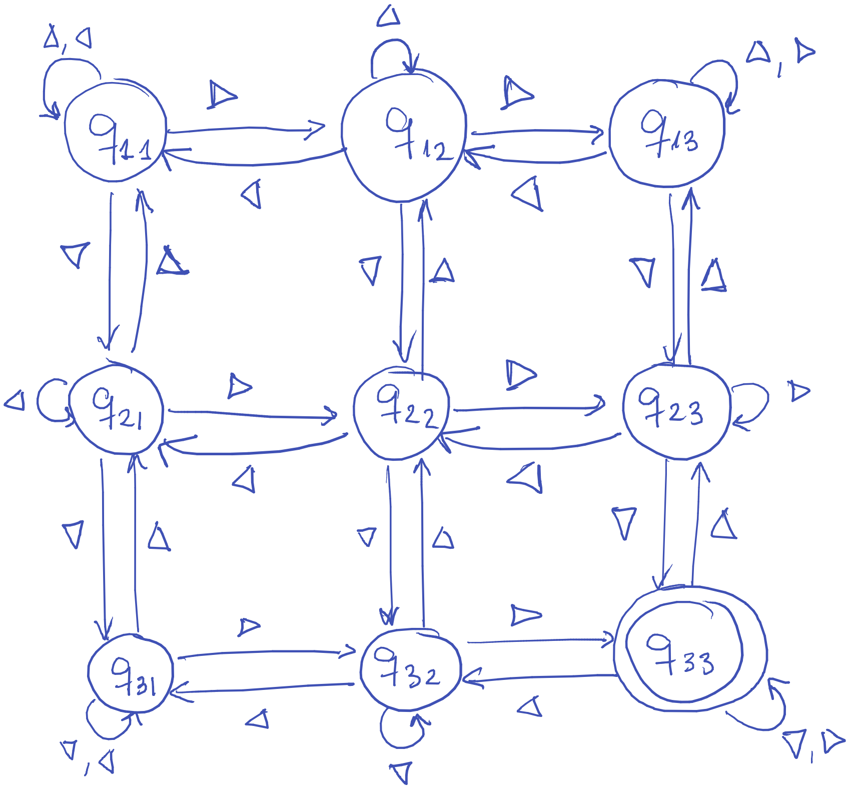
Una máquina de vending acepta 1,20€ y no da cambio. Lleva un AFD que detecta si se ha introducido la cantidad correcta.
La máquina acepta monedas de 20, 50 cents y 1€.



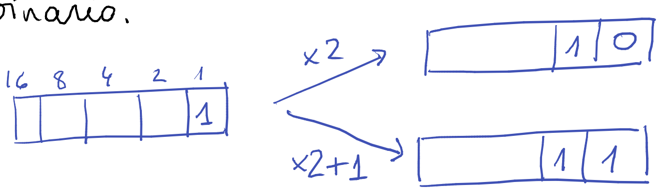


$A = \{ \triangle, \triangleleft, \triangledown, \triangleright \}$
$= \{ W, A, S, D \}$

$L =$ caminos del bot a la meta



Autómata que acepte los múltiplos de 3 en binario.



Restos posibles: 0, 1, 2.

δ	0	1
$\rightarrow q_0$	q_0	q_1
q_1	q_2	q_0
q_2	q_1	q_2

$0 \cdot 2 = 0$

$0 \cdot 2 + 1 = 1$

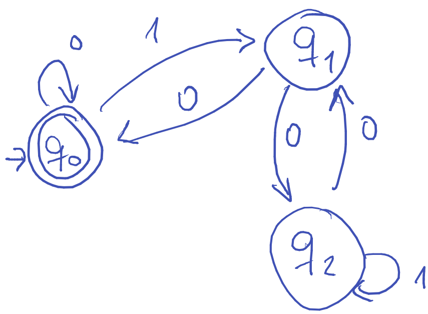
$1 \cdot 2 = 2$

$1 \cdot 2 + 1 \equiv 0$

$2 \cdot 2 \equiv 1$

$2 \cdot 2 + 1 \equiv 2$

1011 : $q_0 \xrightarrow[(1)]{1} q_1 \xrightarrow[(2)]{0} q_2 \xrightarrow[(5)]{1} q_2 \xrightarrow[(11)]{1} q_2$



Autómatas finitos no deterministas

Recordatorio:

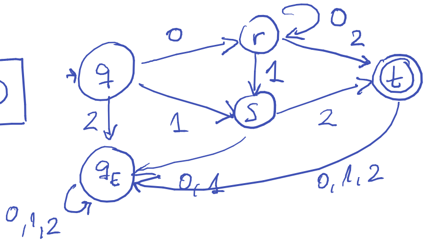
Un AFND es una quíntupla (Q, A, δ, q_0, F) donde:

- Q conjunto de estados
- A alfabeto
- $\delta : Q \times A \to \mathcal{P}(Q) = 2^Q$
- $q_0 \in Q$ inicial
- $F \subset Q$ finales

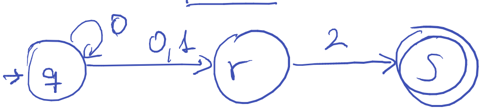
Diferencia con autómatas deterministas:

EJEMPLO $\{0^n u 2 : n \geq 0,\ u \in \{0,1\}\}$

AFD



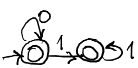
AFND

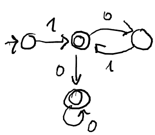


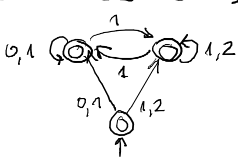
δ	0	1	2
$\to q$	r	s	q_E
r	r	s	t
s	q_E	q_E	t
$\boxed{t}$	q_E	q_E	q_E
q_E	q_E	q_E	q_E

δ	0	1	2
q	$\{q, r\}$	$\{r\}$	$\emptyset$
r	$\emptyset$	$\emptyset$	$\{s\}$
s	$\emptyset$	$\emptyset$	$\emptyset$

EJEMPLOS: $0^i 1^j$, $1(01)^i 0^j$, palabras que no contengan "02" o "20";

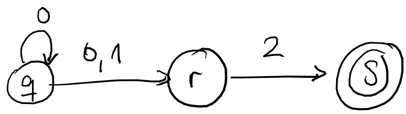




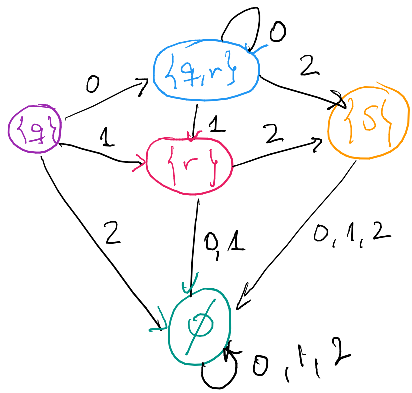


Paso de AFND a AFD

Si δ es la f. transición del AFND: $\delta: Q \times A \to \mathcal{P}(Q)$, hacemos $Q' \subset \mathcal{P}(Q)$ de forma que $\delta': Q' \times A \to Q'$ sea la f. transición del AFD.



$$Q' = \{ \{q\}, \{q,r\}, \{r\}, \emptyset, \{s\} \}$$

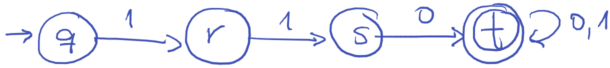


Ejercicios R2.

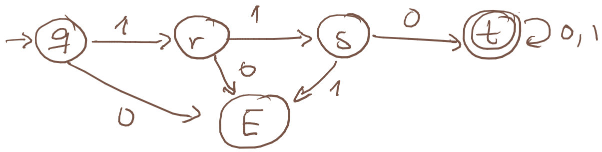
2. $P = \{ S \to 110B \ , \ B \to 1B \mid 0B \mid \varepsilon \}$

Lenguaje: palabras que empiezan por 110.

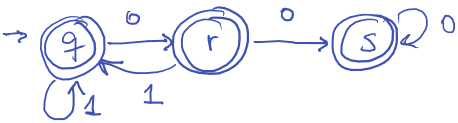
δ	0	1
q	$\emptyset$	r
r	$\emptyset$	s
s	t	$\emptyset$
t	t	t



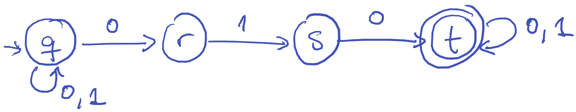
AFD:



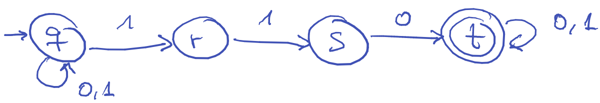
15. "001" no está en u.



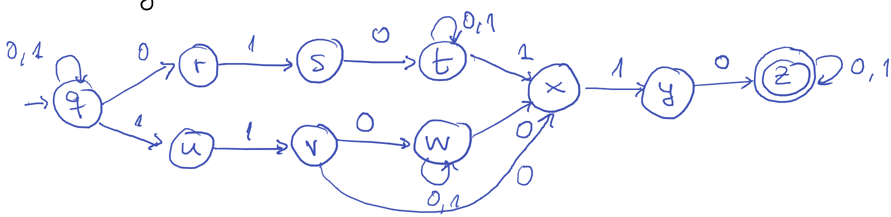
1. "010" está en u:



"110" está en u:

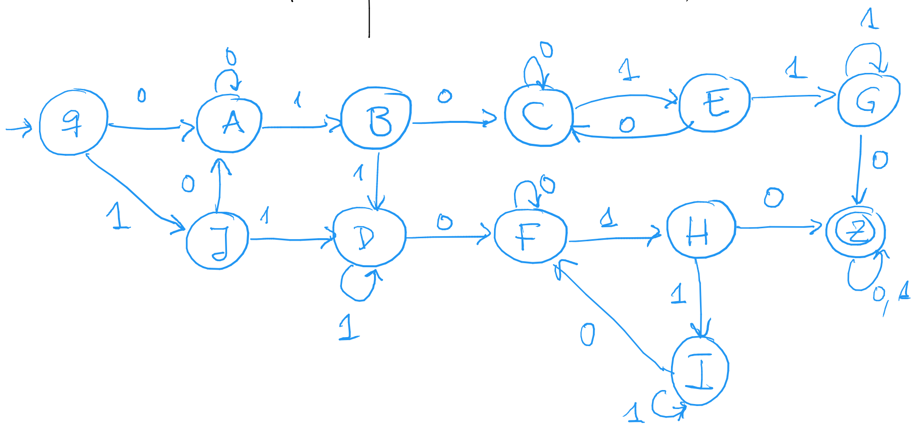


"010" y "110" están ambos en u.

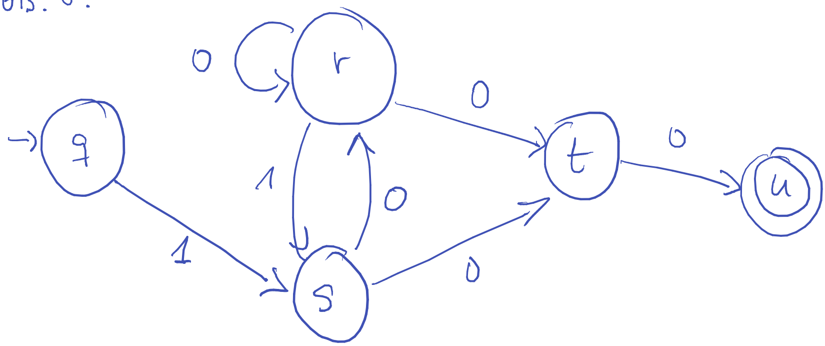


Paso a AFD:

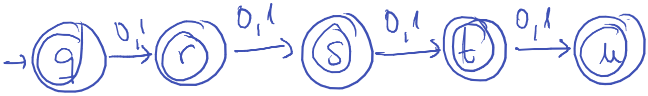
δ		0	1
	$\to q$	$\{q,r\}$	$\{q,u\}$
	r	$\emptyset$	$\{s\}$
	s	$\{t\}$	$\emptyset$
	t	$\{t\}$	$\{t,x\}$
	u	$\emptyset$	$\{v\}$
	v	$\{w,x\}$	$\emptyset$
	w	$\{w,x\}$	$\{w\}$
	x	$\emptyset$	$\{y\}$
	y	$\{z\}$	$\emptyset$
	z	$\{z\}$	$\{z\}$
A	$\{q,r\}$	$\{q,r\}$	$\{q,u,s\}$
B	$\{q,u,s\}$	$\{q,r,t\}$	$\{q,u,v\}$
C	$\{q,r,t\}$	$\{q,r,t\}$	$\{q,u,s,t,x\}$
D	$\{q,u,v\}$	$\{q,r,w,x\}$	$\{q,u,v\}$
E	$\{q,u,s,t,x\}$	$\{q,r,t\}$	$\{q,u,v,t,x,y\}$
F	$\{q,r,w,x\}$	$\{q,r,w,x\}$	$\{q,u,s,w,y\}$
G	$\{q,u,v,t,x,y\}$	z	$\{q,u,v,t,x,y\}$
H	$\{q,u,s,w,y\}$	z	$\{q,u,v,w\}$
I	$\{q,u,v,w\}$	$\{q,r,w,x\}$	$\{q,u,v,w\}$
J	$\{q,u\}$	$\{q,r\}$	$\{q,u,v\}$



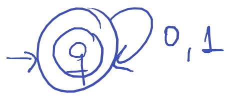
R1bis.b.



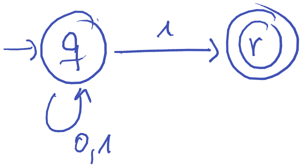
a.



$L = \{0,1\}^*$



acaban en 1

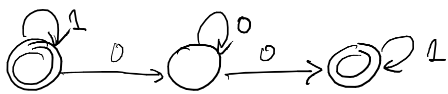


ε - AFND

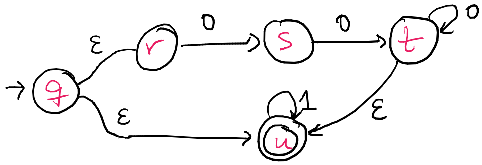
Igual que los AFND pero ahora:

$$\delta : Q \times (A \cup \{\varepsilon\}) \longrightarrow \mathcal{P}(Q)$$

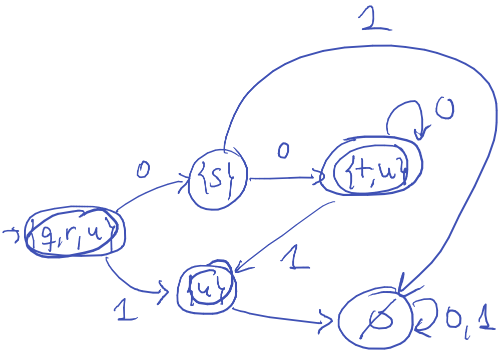
Diferencias entre AFND y ε-AFND:



$\{0^n 1^m : n \neq 1, m \geq 0\}$



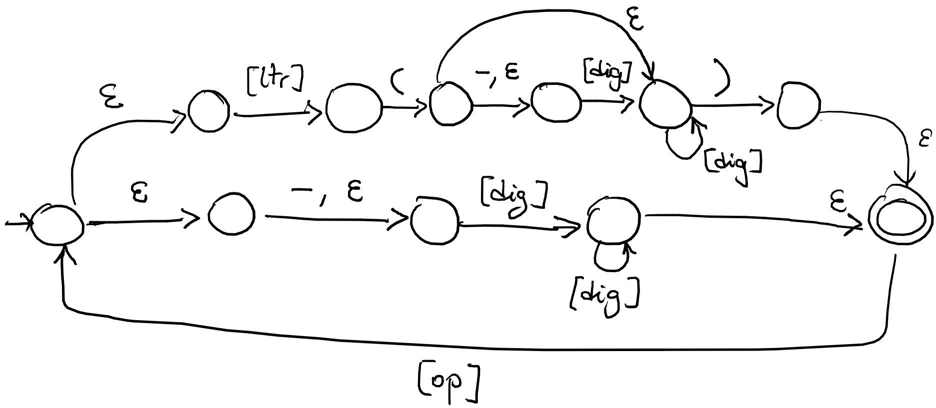
δ	ε	0	1
q	{r,u}	∅	∅
r	∅	{s}	∅
s	∅	{t}	∅
t	{u}	{t}	∅
u	∅	∅	{u}



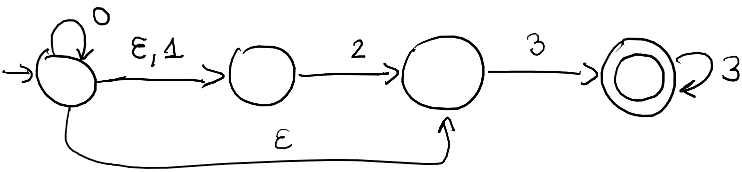
	0	1
{q,r,u}	{s}	{u}
{s}	{t,u}	∅
$\boxed{\{t,u\}}$	{t,u}	{u}
$\boxed{\{u\}}$	∅	{u}
∅	∅	∅

Construir un ε-AFND que reconozca expresiones matemáticas incluyendo: $T = \{-, (,), [dig], [ltr], [op]\}$

- números $-?[dig]^+$
- funciones $[ltr](-?[dig]^+)$ o $[ltr]()$ (e.g. $f(3)$).
- operaciones <expr> [op] <expr>

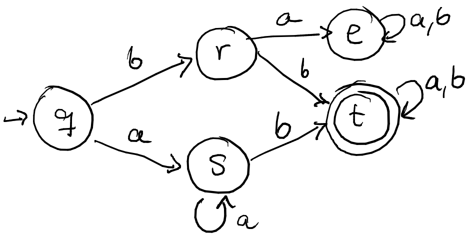


$$\{0^n 1 2 3^m,\ 0^n 2 3^m,\ 0^n 3^m,\ n \geq 0,\ m \geq 1\}$$



AFD y gramáticas regulares

- Gramática lineal por la derecha: paso directo



Palabras que empiezan por 'bb' o una sucesión de a's seguida por una b.

Paso a gramática: creo las variables Q, R, S, T, E

y las producciones
$$\begin{cases} U \to aV & \text{cuando } u \xrightarrow{a} v \\ V \to \varepsilon & \text{cuando } v \text{ es estado final} \end{cases}$$

inicial

$Q \to bR \mid aS$

$R \to aE \mid bT$

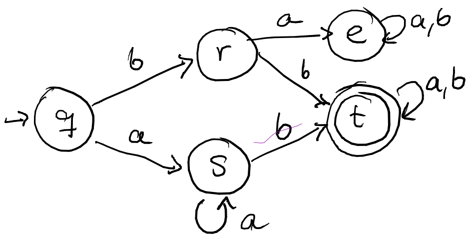
$S \to aS \mid bT$

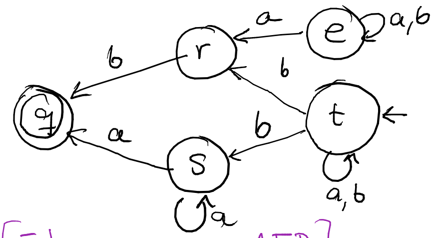
$T \to aT \mid bT \mid \varepsilon$

$E \to aE \mid bE$

AFD y gramáticas regulares

- Gramática lineal por la izquierda: intercambiar flechas del AFD





[Este no es un AFD]

Paso a gramática: creo las variables Q, R, S, T, E

y las producciones $\begin{cases} U \to Va & \text{si } (u) \xrightarrow{a} (v) \\ V \to \varepsilon & \text{si } v \text{ es final} \end{cases}$

inicial

T → Ta | Tb | Rb | Sb

R → Qb

S → Sa | Qa

Q → ε

E → Ea | Eb | Ra ← variable inalcanzable

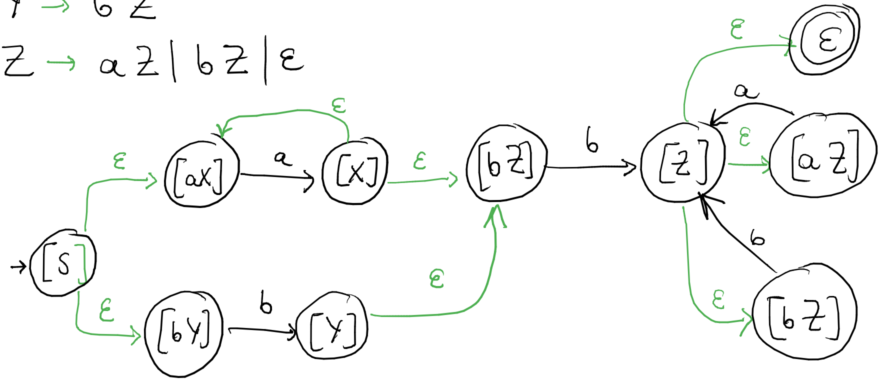
De gramática a autómata

$S \rightarrow aX \mid bY$

$X \rightarrow aX \mid bZ$

$Y \rightarrow bZ$

$Z \rightarrow aZ \mid bZ \mid \varepsilon$

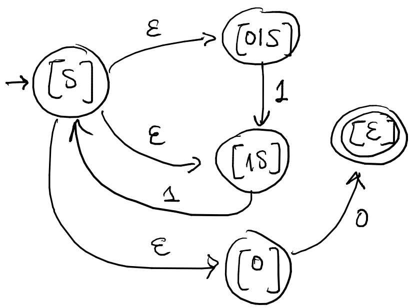


Lineales por la izquierda:

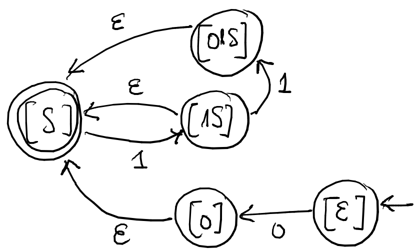
$S \rightarrow S10 \mid S1 \mid 0$

① Invertir producciones: $S \rightarrow 01S \mid 1S \mid 0$

② Pasar a autómata con 1 estado final:



③ Invertir autómata e intercambiar estados inicial y final:



Expresiones regulares

Son expresiones regulares:

- $\emptyset$ denotando el lenguaje vacío: $\{\ \}$
- ε denotando la palabra vacía: $\{\varepsilon\}$
- a denotando la palabra "a": $\{a\}$ $\forall a \in A$
- concatenación de regex: $r_1 r_2 \rightsquigarrow \{uv / u \in L_1, v \in L_2\}$

  EJEMPLO: ab, abc
- unión de regex: $r_1 + r_2 \rightsquigarrow L_1 \cup L_2 = \{u / u \in L_1 \text{ o } u \in L_2\}$

  EJEMPLO: $a + b + ab \rightsquigarrow \{a, b, ab\}$
- clausura de regex: $r^* \rightsquigarrow L^* = \{u^i / i \geq 0, u \in L\}$

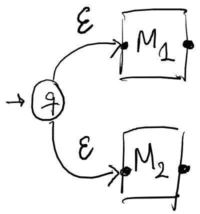
  EJEMPLO: $a^* \rightsquigarrow \{\varepsilon, a, aa, aaa \dots\}$

  $(a^* + b^*)c \rightsquigarrow \{c, ac, bc, aac, bbc, \dots\}$

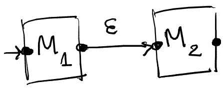
  $(a + b)^* c \rightsquigarrow \{c, ac, abc, bc, bac, \dots\}$

ε - AFND y expresiones regulares

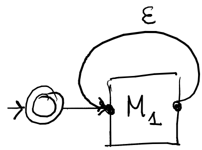
Supongamos que tenemos AFND pero queremos realizar con ellos las operaciones de expresiones regulares: unión, concatenación y clausura. De forma natural surgen las transiciones nulas:



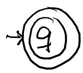
operación +

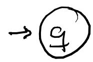


operación ·



operación *

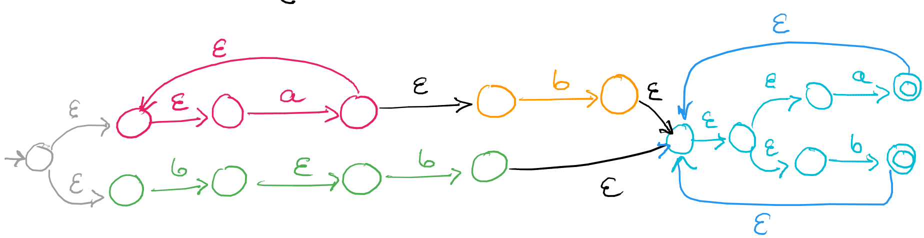
Sabiendo que ε viene dado por ,

∅ viene dado por  y

a viene dado por 

podemos pasar cualquier expresión regular a ε-AFND

EJEMPLO $(a^*b + bb)(a+b)^*$



ε - AFND y expresiones regulares

Paso de AFD/AFND a Regex

Definimos $x_i = \left\{ \text{palabras } u \;/\; \exists\, (q_i, u) \xrightarrow{*} (q_f, \dots) \text{ con } q_f \text{ final} \right\}$

- si $q_i \in F \Rightarrow \varepsilon \in x_i$
- si $\delta(q_i, a) = q_j \in F \Rightarrow a \in x_i$
- si $\delta(q_i, a) = q_j \Rightarrow a x_j \subseteq x_i$

Se define el sistema de ecuaciones para cada x_i:

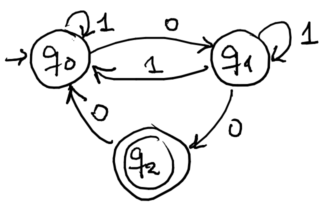
$$x_i = \varepsilon[q_i \in F] + \underbrace{a x_j}_{\delta(q_i,a)=q_j} + \dots$$

Se resuelve hasta llegar a $x_i = \underset{\not\ni\,\varepsilon}{A}\, x_i + \underset{\not\ni\,x_i}{B}$, Ecuación característica

que se soluciona como $x_i = A^* B$.

La regex final es la solución de x_0.

EJEMPLO



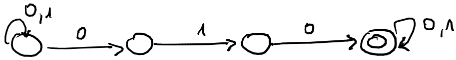
$$\begin{cases} x_0 = 1x_0 + 0x_1 \\ x_1 = 1x_1 + 1x_0 + 0x_2 \\ x_2 = 0x_0 + \varepsilon \end{cases} \Rightarrow$$

$$\Rightarrow \begin{cases} x_0 = 1^*0x_1 \\ x_1 = 1x_1 + 1x_0 + 0x_2 \\ x_2 = 0x_0 + \varepsilon \end{cases} \Rightarrow \begin{cases} x_0 = 1^*0x_1 \\ x_1 = 1x_1 + 1x_0 + 0x_2 \\ x_2 = 01^*0x_1 + \varepsilon \end{cases}$$

$$\Rightarrow x_1 = 1x_1 + 11^*0x_1 + 0(01^*0x_1 + \varepsilon) =$$
$$= (1 + 11^*0 + 001^*0)x_1 + 0\varepsilon =$$
$$= (1 + 11^*0 + 001^*0)^*0$$

$$\boxed{x_0 = 1^*0(1 + 11^*0 + 001^*0)^*0}$$

EJEMPLO 2. Pasar a expresión regular:



$$\begin{cases} x_0 = 0x_0 + 1x_0 + 0x_1 \\ x_1 = 1x_2 \\ x_2 = 0x_3 \\ x_3 = 0x_3 + 1x_3 + \varepsilon \end{cases} = (0+1)x_3 + \varepsilon = (0+1)^*\varepsilon = (0+1)^*$$

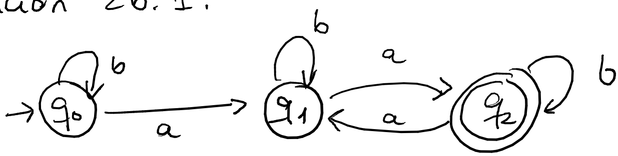
$$x_2 = 0x_3 = 0(0+1)^*$$

$$x_1 = 1x_2 = 10(0+1)^*$$

$$x_0 = 0x_0 + 1x_0 + 0x_1 = (0+1)x_0 + 010(0+1)^* =$$

$$= (0+1)^* 010 (0+1)^*$$

Relación 26.1.



$$\begin{cases} x_0 = bx_0 + ax_1 \\ x_1 = bx_1 + ax_2 \\ x_2 = bx_2 + ax_1 + \varepsilon \end{cases} \rightsquigarrow \begin{cases} x_0 = b^*ax_1 \\ x_1 = b^*ax_2 \\ x_2 = bx_2 + a(b^*ax_2) + \varepsilon \end{cases}$$

$$= (b + ab^*a)x_2 + \varepsilon =$$

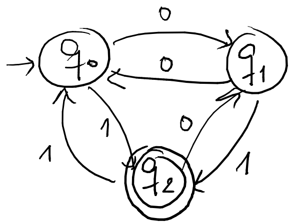
$$= (b+ab^*a)^*\varepsilon =$$

$$= (b+ab^*a)^*$$

$$x_1 = b^*a(b+ab^*a)^*$$

$$\underline{x_0 = b^*ab^*a(b+ab^*a)^*}$$

3.



$$\begin{cases} x_0 = 0x_1 + 1x_2 \\ x_1 = 0x_0 + 1x_2 \\ x_2 = 0x_1 + 1x_0 + \varepsilon \end{cases}$$

$$x_0 = 0x_1 + 1(0x_1 + 1x_0 + \varepsilon)$$

$$x_1 = 0x_0 + 1(0x_1 + 1x_0 + \varepsilon) = 10x_1 + (0+11)x_0 + 1$$

$$= (10)^*(0+11)x_0 + 1$$

$$x_0 = 0\,10^*(0+11)x_0 + 01 +$$

$$+ 1(0(10)^*(0+11)x_0 + 1 + 1x_0 + \varepsilon) =$$

$$= (0(10)^*(0+11) + 1010^*(0+11) + 11)x_0 +$$

$$+ 01 + 11 + 1 =$$

$$= (0(10)^*(0+11) + 10(10)^*($$

Máquinas de Mealy y Moore

Tienen una función salida:

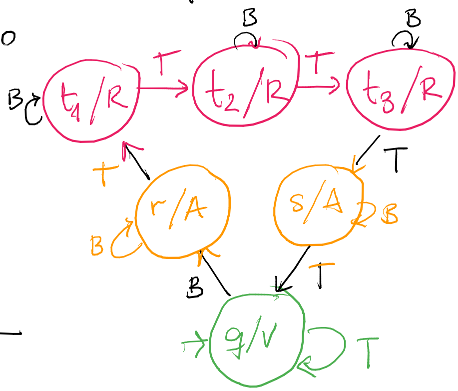
MOORE $\lambda : Q \to B$

estados   alfabeto de salida

MEALY $\lambda : Q \times A \to B$

En la máquina de Moore la salida depende solo del estado, pero en la de Mealy depende del estado y la entrada actual.

Ejemplo: máquina de Moore que controla un semáforo verde / ámbar / rojo

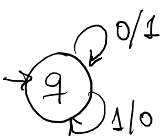


B = botón pulsado

T = temporizador

δ/λ	B	T
q/V	r	q
r/A	r	t_1
	$\vdots$	

Ejemplo: máquina de Mealy que devuelve el complemento a uno en binario:



δ/λ	0	1
q	$0/1$	$1/0$

Ejercicios máquinas de Mealy / Moore

- Máquina de Moore que devuelve el mód 4 del producto parcial

Si leemos 372947 queremos los mód 4 de los productos parciales:

- $3 \equiv 3$ mód 4
- $3 \cdot 7 = 21 \equiv 1$ mód 4
- $3 \cdot 7 \cdot 2 = 42 \equiv 2$ mód 4
- $3 \cdot 7 \cdot 2 \cdot 9 = 378 \equiv 2$ mód 4
- $3 \cdot 7 \cdot 2 \cdot 9 \cdot 4 = 1512 \equiv 0$ mód 4
- $3 \cdot 7 \cdot 2 \cdot 9 \cdot 4 \cdot 7 = 10584 \equiv 0$ mód 4

Salida: 312200

Sabemos que si $n \equiv k$ (mód 4) entonces para $m \in \mathbb{Z}$
$nm \equiv km$ (mód 4).

Por tanto, nos basta con memorizar el mód 4 del resultado anterior. $d_1 \equiv m_1$ (mód 4) $\Rightarrow d_1 d_2 \equiv m_1 d_2$ (mód 4)...

- $3 \equiv 3$ mód 4
- $3 \cdot 7 \equiv 1$ mód 4 $\nearrow$ $1 \cdot 2 \equiv 2$ mód 4
- $2 \cdot 9 \equiv 2$ mód 4 $\nearrow$ $2 \cdot 4 \equiv 0$ mód 4
- $0 \cdot 7 \equiv 0$ mód 4

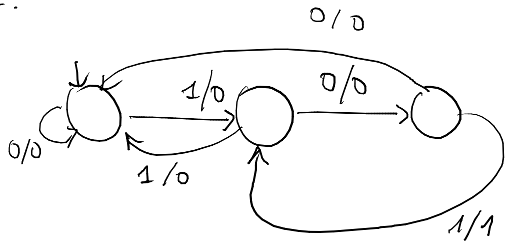
Tendremos 4 estados: q_0, q_1, q_2, q_3, indicando q_i: $d \equiv i$ (mód 4). El alfabeto tiene 10 caracteres. El estado inicial ha de ser 1 para verificar la regla anterior ($1 \equiv 1$ mód 4 $\rightsquigarrow$ $d_1 \cdot 1 \equiv d_1$ mód 4)

δ / λ	0	1	2	3	$4 \equiv 0$	$5 \equiv 1$	$6 \equiv 2$	$7 \equiv 3$	$8 \equiv 0$	$9 \equiv 1$
$q_0/0$	q_0	q_0	q_0	q_0	q_0	q_0	q_0	q_0	q_0	q_0
→ $q_1/1$	q_0	q_1	q_2	q_3	q_0	q_1	q_2	q_3	q_0	q_1
$q_2/2$	q_0	q_2	q_0	q_2	q_0	q_2	q_0	q_2	q_0	q_2
$q_3/3$	q_0	q_3	q_2	q_1	q_0	q_3	q_2	q_1	q_0	q_3

Leyendo 372947: $q_1 \xrightarrow{3} q_3/3 \xrightarrow{7} q_1/1 \xrightarrow{2} q_2/2 \rightarrow$
$\xrightarrow{9} q_2/2 \xrightarrow{4} q_0/0 \xrightarrow{7} q_0/0$

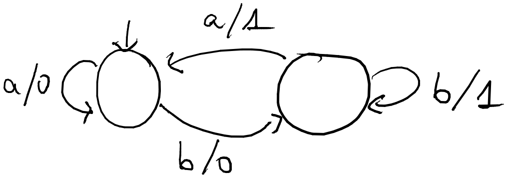
R-Mealy-Moore.

1.

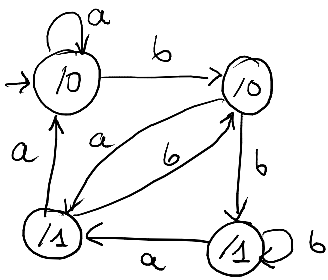


2.

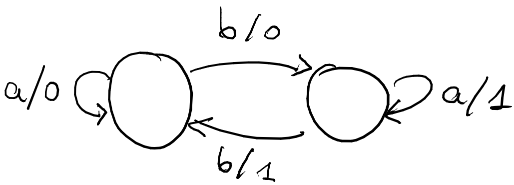
MEALY

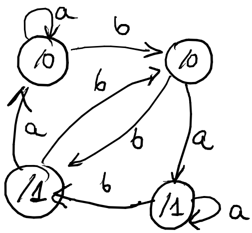


MOORE



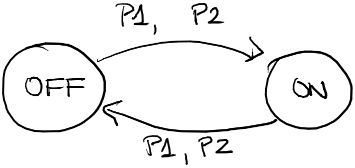
3.



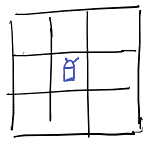


aba → 001 abba → 0011

11.



- Máquina de Mealy que transforme un movimiento relativo (avanzar, girar izq., girar dcha) en movimiento absoluto (norte, sur, este, oeste, nada).



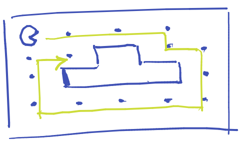
1. Alfabeto de entrada $\{A, I, D\}$
2. Alfabeto de salida $\{\uparrow, \downarrow, \leftarrow, \rightarrow, \cdot\}$
3. Estados? Indican orientación: $\{\underset{\triangle}{q_N}, \underset{\triangledown}{q_S}, \underset{\triangleleft}{q_O}, \underset{\triangleright}{q_E}\}$

δ/λ	A	I	D
$\rightarrow q_N$	$q_N/\uparrow$	$q_O/\cdot$	$q_E/\cdot$
q_S	$q_S/\downarrow$	$q_E/\cdot$	$q_O/\cdot$
q_O	$q_O/\leftarrow$	$q_S/\cdot$	$q_N/\cdot$
q_E	$q_E/\rightarrow$	$q_N/\cdot$	$q_S/\cdot$

- Máquina de Mealy que codifique un 'Pacman' que siempre va siguiendo un rastro de comida (C), se puede encontrar con bordes (B) y con casillas vacías (O). Pacman puede avanzar (A), girar a la izquierda (I) o a la derecha (D). Pacman termina (T) cuando no hay comida cerca.

1. Estados? Indican cuánta vuelta ha dado Pacman:

   q_0, q_1, q_2, q_3



δ/λ	C	B	O
q_0	q_0/A	q_1/D	q_1/D
q_1	q_0/A	q_2/D	q_2/D
q_2	q_0/A	q_3/D	q_3/D
q_3	q_0/A	q_3/T	q_3/T

- Máquina de Moore que sume números en binario.

La máquina lee de derecha a izquierda los números por parejas de dígitos:

$01011 + 10001 = 11100 \rightsquigarrow$

$$\begin{array}{ccccc} & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ \hline 0 & 0 & 1 & 1 & 1 \end{array}$$

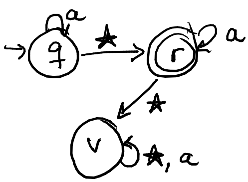
δ/λ	$\begin{smallmatrix}0\\0\end{smallmatrix}$	$\begin{smallmatrix}0\\1\end{smallmatrix}$	$\begin{smallmatrix}1\\0\end{smallmatrix}$	$\begin{smallmatrix}1\\1\end{smallmatrix}$
$q_0/0$	q_0	q_1	q_1	q_{0c}
$q_1/1$	q_0	q_1	q_1	q_{0c}
$q_{0c}/0$	q_1	q_{0c}	q_{0c}	q_{1c}
$q_{1c}/1$	q_1	q_{0c}	q_{0c}	q_{1c}

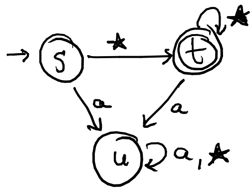
Máquina de Mealy:

δ/λ	$\begin{smallmatrix}0\\0\end{smallmatrix}$	$\begin{smallmatrix}0\\1\end{smallmatrix}$	$\begin{smallmatrix}1\\0\end{smallmatrix}$	$\begin{smallmatrix}1\\1\end{smallmatrix}$
q_0	$q_0/0$	$q_0/1$	$q_0/1$	$q_1/0$
q_1	$q_0/1$	$q_1/0$	$q_1/0$	$q_1/1$

Autómata producto

$L = \{ u \in \{a, \star\}^* / N_\star(u) = 1 \} \cup \{\star\}^+$





Producto: