

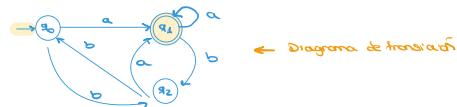
¿Qué es un autómata?

Un autómata finito es una tupla $M = (Q, A, \delta, q_0, F)$ donde:

- Q es un conjunto finito llamado conjunto de estados.
- A es un alfabeto llamado alfabeto de entrada.
- δ es una aplicación llamada función de transición: $\delta: Q \times A \rightarrow Q$
- q_0 es un elemento de Q llamado estado inicial.
- F es un subconjunto de Q , llamado conjunto de estados finales.

Sea el autómata $M = (Q, A, \delta, q_0, F)$, donde:

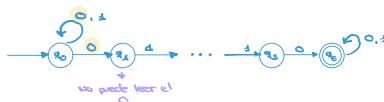
- $Q = \{q_0, q_1, q_2\}$, $A = \{a, b\}$, $F = \{q_2\}$
- $\delta(q_0, a) = q_1$, $\delta(q_0, b) = q_2$
- $\delta(q_1, a) = q_2$, $\delta(q_1, b) = q_2$
- $\delta(q_2, a) = q_1$, $\delta(q_2, b) = q_0$



Típos:

- Autómata finito no determinista :

- Se pueden tener deterministas añadiendo un nodo entre las transiciones no definidas.
- Puede haber estados con 2 transiciones para una misma entrada o geen ninguna transición.



- El lenguaje aceptado por un autómata finito no determinista es $L(M) = \{u \in A^*: \delta^*(q_0, u) \cap F \neq \emptyset\}$

- Autómata finito determinista:

- cada estado tiene una sola transición para una determinada entrada y no puede tener transiciones nulas.

- Autómata finito no determinista con transiciones nulas:

- Es un autómata finito no determinista donde algunas transiciones son la palabra vacía (ϵ).



Equivalencia

- Un lenguaje L puede ser aceptado por un autómata finito determinista $\Leftrightarrow L$ puede ser aceptado por un autómata finito no determinista.

\Leftarrow Los autómatas deterministas son también no deterministas.

\Leftarrow Dado un autómata no determinista se le hace corresponder no determinista que recorre todos los caminos al mismo tiempo.

- Un autómata no-determinista y su determinista asociado aceptan el mismo lenguaje.

- Un lenguaje L es aceptado por un autómata finito determinista $\Leftrightarrow L$ es aceptado por un autómata finito no determinista con transiciones nulas.

- El autómata finito determinista generado por un autómata finito no determinista con transiciones nulas, genera el mismo lenguaje.

- ④ Considera el siguiente Autómata Finito Determinista (AFD), $M = (Q, A, \delta, q_0, F)$, donde:

$$Q = \{q_0, q_1, q_2\}$$

$$A = \{0, 1\}$$

- La función de transición:

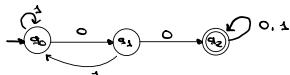
$$\delta(q_0, 0) = q_1 \quad \delta(q_0, 1) = q_0$$

$$\delta(q_1, 0) = q_2 \quad \delta(q_1, 1) = q_1$$

$$\delta(q_2, 0) = q_2 \quad \delta(q_2, 1) = q_2$$

$$F = \{q_2\}$$

Para describir el lenguaje dibujaremos el diagrama de transición:



, luego, tendremos $G = \langle V, \{S, A, B\}, T = \{0, 1\}, \varnothing, \Sigma \rangle$

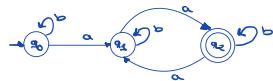
$$S \rightarrow 1S10A$$

$$A \rightarrow 1S10B$$

\Rightarrow Palabras que contienen al 00 como subcadena

$$B \rightarrow 0S11B1\Sigma$$

② Dado el AFN



describir el lenguaje aceptado por dicho autómata.

La gramática generativa será: $G = (V = \{S, A, B\}, T = \{a, b\}, P, S)$:

$$S \rightarrow bS1aA$$

$$A \rightarrow bA1aB$$

$$B \rightarrow bB1aA1E$$

$$\Rightarrow L(G) = \{b^k a^l b^m a^n : k \leq m \leq n\}$$

③ Dibujar AFNs que acepten los siguientes lenguajes con alfabeto {0,1}:

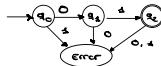
a) El lenguaje vacío



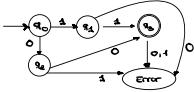
b) El lenguaje formado por la palabra vacía



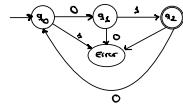
c) El lenguaje formado por la palabra 01, o sea, 101.



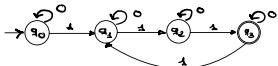
d) El lenguaje 111,000



e) El lenguaje $\{(01)^n \mid n \geq 0\}$



f) El lenguaje formado por las cadenas con 0's y 1's donde el número de unos es divisible por 3.

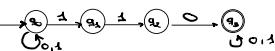
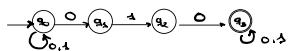


④ Construir un autómata finito no determinista capaz de aceptar una cadena ue $0^0.1^4$, que contenga la subcadena 010. Construir un autómata finito no determinista capaz de aceptar una cadena ue $0^0.1^4$, que contenga la cadena 110. Dibujar un AFD capaz de aceptar una cadena ue $0^0.1^4.1^4$, que contenga simultáneamente las subcadenas 010 y 110.

AFND con 010 como subcadena

AFND con 110 como subcadena

AFD que contenga simultáneamente las dos cadenas

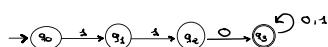


Como es no determinista, pide reducir entradas que no generen transición

⑤ obtener a partir de la gramática regular $G = (V = \{S, B\}, T = \{0, 1\}, P, S)$, con

$$P: S \rightarrow 110B, B \rightarrow 1B, B \rightarrow 0B, B \rightarrow \epsilon$$

un AFND que reconozca el lenguaje generado por esa gramática.

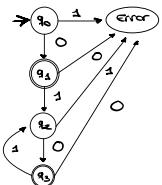


(como no es determinista, no todas las entradas implican un cambio de transición de estados)

- ⑥ Dada la gramática regular $G_1 = (L, S, \Sigma, P, S)$, con

$$P = \{ S \rightarrow S10, S \rightarrow 0 \}$$

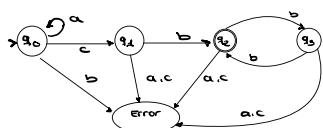
obtener un AFD que reconozca el lenguaje generado por esa gramática.



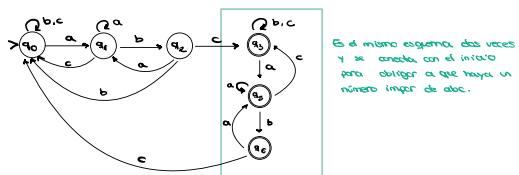
- ⑦ Construir un AFD que acepte el lenguaje generado por la siguiente gramática:

$$S \rightarrow AB, \quad A \rightarrow aA, \quad A \rightarrow c$$

$$B \rightarrow bBb, \quad B \rightarrow b$$



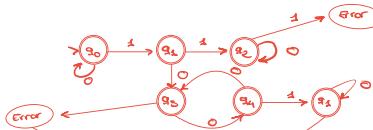
- ⑧ Construir un AFD que acepte el lenguaje $L(S)_{\text{dúctil}}$ de todas las palabras con un número impar de ocurrencias de la subcadena abc.



- ⑨ Sea L el lenguaje de todas las palabras sobre el alfabeto $\{0, 1\}$ que no contienen dos unos que estén separados por un número impar de símbolos. Describir un AFD que acepte este lenguaje.

Solución de Marim:

1001001 no lo acepta. Solo puede haber dos 1



Expresiones regulares

Si A es un alfabeto, una expresión regular sobre este alfabeto se define de la siguiente forma:

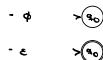
- \emptyset es una expresión regular que denota el lenguaje vacío.
- e es una expresión regular que denota el lenguaje $\{e\}$.
- Si $a \in A$, a es una expresión regular que denota el lenguaje $\{a\}$.
- Si R y S son expresiones regulares denotando los lenguajes R y S , entonces definimos las siguientes operaciones:
 - Unión: $(R \cup S)$ es la expresión regular de $R \cup S$.
 - Concatenación: (RS) denota RS .
 - Cierre: R^* denota R^* . Es el lenguaje sobre A , definido como la clausura de Kleene de R como $L^* = \bigcup_{n=0}^{\infty} R^n$. Representa el conjunto de las cadenas que se pueden formar teniendo cualquier número de ocurrencias del conjunto inicial, posiblemente con repeticiones, y concatenándolas entre si. $A \cap R^*$ se le denota R^+ .
- $00 \Rightarrow$ El conjunto $\{00\}$
- $01^* + 0 \Rightarrow$ Conjunto de palabras que empiezan por 0 y después tienen una sucesión de 1.
- $(1+10)^* \Rightarrow$ Conjunto de palabras en las que 0 está siempre precedido por 1.
- $(0+1)^* 011 \Rightarrow$ Conjunto de palabras que terminan en 011.
- $0^* 1^* \Rightarrow$ Conjunto de palabras formadas por una sucesión de ceros, seguidas por una sucesión de unos (las sucesiones pueden ser vacías).
- $00^* 11^* = 0^* 1^* \Rightarrow$ Lo mismo, pero ahora ninguna sucesión puede ser vacía.

Las propiedades de las expresiones regulares son:

1) $r_1 + r_2 = r_2 + r_1$	5) $r_1(r_2 + r_3) = r_1r_2 + r_1r_3$
2) $r_1 + (r_2 + r_3) = (r_1 + r_2) + r_3$	9) $(r_1 + r_2)r_3 = r_1r_3 + r_2r_3$
3) $r_1(r_2r_3) = (r_1r_2)r_3$	10) $r^* + \epsilon = r^*$
4) $r\epsilon = r$	11) $r^* + \epsilon = r^*$
5) $r\phi = \phi$	12) $(r + \epsilon)^* = r^*$
6) $r + \phi = r$	13) $(r + \epsilon)^* = r^*$
7) $\epsilon^* = \epsilon$	14) $(r_1^* + r_2^*) = (r_1 + r_2)^*$

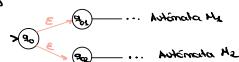
Equivalecia autómatas - Expresiones regulares

- Un lenguaje es aceptado por un autómata finito determinista \Leftrightarrow Puede representarse mediante una expresión regular.

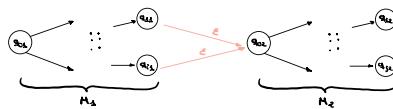


- Si M_1 es el autómata que acepta el lenguaje representado por r_1 y M_2 el que acepta r_2 , entonces:

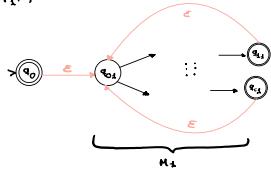
1) Unión (r_1r_2)



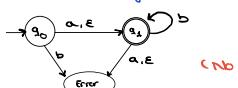
2) Concatenación (r_1r_2)



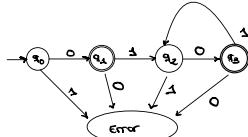
3) Cláusula (r_1^*)



- ⑩ Dada la expresión regular $(a + \epsilon)b^*$ encontrar el AFD asociado.



- ⑪ Obtener un AFD que acepte el lenguaje generado por la expresión regular $0(01)^*$



- ⑫ obtener una expresión regular para el lenguaje complementario al aceptado por la gramática.

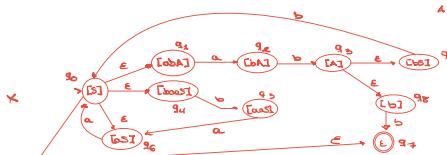
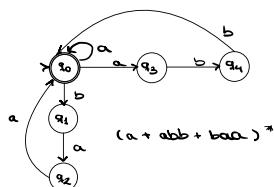
$$S \rightarrow abA \mid bB \mid baB \mid \epsilon$$

$$A \rightarrow b\bar{a} \mid b$$

$$B \rightarrow a\bar{b}$$

Pista. Construir un AFD asociado.

comentemos transformada en una gramática regular: $S \rightarrow abA \mid a\bar{b}1 \mid baB \mid \epsilon$. $A \rightarrow b\bar{a} \mid b$. Tenemos una gramática regular, tendré que ir a la derecha:



(13) Dar expresiones regulares para los lenguajes sobre el alfabeto $\{a, b\}$ dados por las siguientes tres condiciones:

a) Palabras que no contienen la subcadena a .

b^*

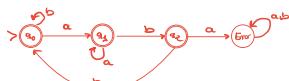
b) Palabras que no contienen la subcadena ab .

$b^* a^*$

c) Palabras que no contienen la subcadena aba .

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

En clase lo hicimos construyendo el autómata



$$\begin{aligned} q_0 &= \epsilon + bq_0 + aq_1 \\ q_1 &= cbq_2 + aq_2 \\ q_2 &= \epsilon + bq_3 \\ \Rightarrow q_0 &= \epsilon + bq_0 + aq_1 \\ q_1 &= a^* (\epsilon + bq_2) \\ q_2 &= \epsilon + bq_3 \\ \Rightarrow q_0 &= \epsilon + bq_0 + a(a^* (\epsilon + b(\epsilon + bq_2))) \\ q_1 &= a^* (\epsilon + b + bbq_3) \\ \Rightarrow q_0 &= \epsilon + bq_0 + a(a^* (a^* + a^* b + a^* bbq_3)) \\ \Rightarrow q_0 &= \epsilon + bq_0 + a(a^* + a^* b + a^* bbq_3) \\ \Rightarrow q_0 &= \epsilon + b^* + a^* + a^* b + a^* bbq_3 \\ \Rightarrow q_0 &= \epsilon + a^* + a^* b + (b + a^* bb)q_3 \\ \Rightarrow q_0 &= (b(a^* bb))^* (\epsilon + a^* + a^* b) \end{aligned}$$

(como se hacen este tipo de operación estaré explicando más adelante)

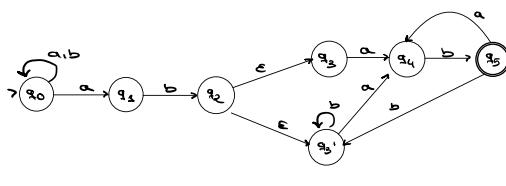
(14) Determinar si el lenguaje generado por la siguiente gramática es regular:

$$S \rightarrow AabbB$$

$$A \rightarrow aa1BA1E$$

$$B \rightarrow Bab1bb1ab1b$$

En caso de que lo sea, encontrar la expresión regular asociada.



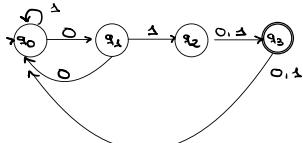
Además, este automata no tiene bucles ni transiciones nulas.

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

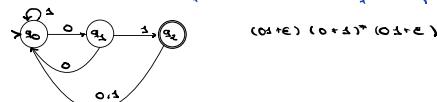
También podemos sacarlo como la concatenación de 3 lenguajes trivialmente regulares.

(15) Sobre el alfabeto $A = \{0, 1\}$ realizar las siguientes tareas:

a) Describir un autómata finito determinista que acepte todas las palabras que contienen al 011 o al 050 (o a las dos) como subcadenas.



b) Describir un autómata finito determinista que acepte todas las palabras que emplean 0 terminan (o ambas cosas)



c) Dar una expresión regular para el conjunto de las palabras en las que hay dos ceros separados por un número de símbolos que es múltiplo de 4 (los símbolos que separan los ceros pueden ser ceros y puede haber otros símbolos delante o detrás de estos dos ceros).

Solución valentín: $(0+1)^* 0 (0+1)(0+1)(0+1)(0+1)^* 0 (0+1)^*$

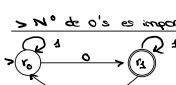
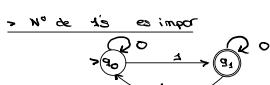
d) Dar una expresión regular para las palabras en las que el número de ceros es divisible por 4.

Solución valentín: $1^* (00^* 00^* 00^* 00^*)^*$

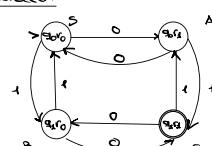
(16) Construye una gramática regular que genere el siguiente lenguaje:

$$L_1 = \{w \in \{0, 1\}^* : \text{el número de } 1's \text{ y de } 0's \text{ es igual}\}$$

Podemos dar un autómata finito determinista:



> Intersección:

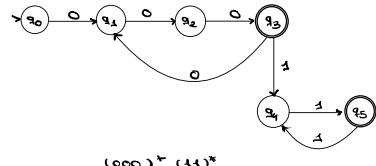


$$\begin{aligned} S &\rightarrow 0A11B \\ A &\rightarrow 0S11C \\ B &\rightarrow 1S11C \\ C &\rightarrow 1A10B1C \end{aligned}$$

⑫ Encuentra una expresión regular que represente el siguiente lenguaje:

$$L_2 = \{ 0^m 1^n : m \geq 0, n \text{ múltiplo de } 3 \text{ y } m \text{ es par} \}$$

Comencemos dando un autómata finito, y de ahí pasaremos a la expresión regular:



$$\begin{aligned} q_0 &= 0q_1 \\ q_1 &= 0q_2 \\ q_2 &= 0q_3 \\ q_3 &= 0q_4 + 1q_4 \\ q_4 &= 1q_5 \\ q_5 &= 1q_4 + \epsilon \end{aligned}$$

$$\begin{aligned} q_0 &= 000q_3 \\ q_1 &= 000q_3 + 11q_5 \\ q_2 &= 11q_5 + \epsilon \Rightarrow q_0 = (11)^* \\ q_3 &= 000q_3 \\ q_4 &= 000q_3 + 01^+ \\ q_5 &= 000q_3 + \epsilon \end{aligned}$$

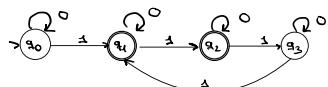
$$q_0 = 000q_3$$

$$q_0 = (000)^* (11)^*$$

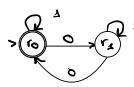
⑬ Dibuja un autómata finito determinista que reconozca el siguiente lenguaje:

$$L_3 = \{ u \in \{0,1\}^* : \text{el número de } 1's \text{ no es múltiplo de } 3 \text{ y el número de } 0's \text{ es par} \}$$

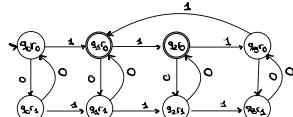
> N° de 1's no es múltiplo de 3:



> N° 0's es par:



> Intersección:



¿Cuál es el proceso correcto para hallar la expresión regular aceptada por un autómata?

La expresión regular del lenguaje aceptado por un autómata viene dada por la expresión $\bigcup_{i,j \in I} R_{ij}^n$

- Supongamos que tenemos $Q = \{q_0, \dots, q_n\}$, entonces R_{ij}^k , con $i, j = 0, 1, \dots, n$ y $k = 0, 1, \dots, n$ se define como el conjunto de palabras del alfabeto tal que:

$$\begin{array}{c} q_i \xrightarrow{\alpha} \dots \xrightarrow{\alpha} q_l \xrightarrow{\alpha} \dots \xrightarrow{\alpha} q_j \\ k \leq l \end{array} \quad \text{síntesis un prefijo no propio (no sea } \epsilon \text{ o } u)$$

Palabras que permiten pasar el autómata del estado $i \rightarrow j$ y que los estados intermedios tienen numeración $\leq k$.

A cada R_{ij}^k le podemos asignar una expresión regular r_{ij}^k de la siguiente forma:

$$r_{ij}^k = \begin{cases} \alpha \dots + \alpha p(ij) & \text{si } i \neq j \\ \alpha + \dots + \alpha q_j & \text{si } i = j \end{cases} \quad \text{donde } p(ij) = \delta(q_i, \alpha) = q_j$$

Esto se debe a que R_{ij}^0 o pasar de $i \rightarrow j$ y los estados intermedios tienen numeración menor o igual que 0, pero eso no es posible ya que los estados tienen numeración mayor que 0, luego son las símbolos que nos llevan de $i \rightarrow j$ sin estados intermedios.

Si este conjunto es vacío la expresión regular será:

$$r_{ij}^0 = \begin{cases} \emptyset & \text{si } i \neq j \\ \epsilon & \text{si } i = j \end{cases}$$

Recurriendo, tendremos:

$$R_{ij}^k = R_{ij}^{k-1} \cup R_{ik}^{k-1} (R_{kk}^{k-1})^* R_{jk}^{k-1} \longrightarrow r_{ij}^k = r_{ij}^{k-1} + r_{ik}^{k-1} (r_{kk}^{k-1})^* r_{jk}^{k-1}$$



Por tanto, si $F = \{q_1, \dots, q_n\}$ y q_f el estado final, $L(M) = \bigcup_{i \in F} q_f^n$ (unión de los lenguajes que llevan el autómata desde el estado inicial al final).

y la expresión regular será sumar dichas expresiones regulares $\Rightarrow r_{1f}^n + \dots + r_{nf}^n$

Método alternativo: Equaciones expresiones regulares

Si r_1, \dots, r_n son variables representando expresiones regulares, una ecuación para la expresión regular r_i es una expresión de la forma:

$$r_i = \alpha_i + \beta_1 r_1 + \dots + \beta_n r_n$$

donde $\alpha_i, \beta_1, \dots, \beta_n$ son expresiones regulares concretas.

Luego, si r_1, \dots, r_n son variables representando expresiones regulares concretas y tenemos una ecuación para cada expresión regular, una ecuación es una asignación de una expresión regular concreta a cada variable r_i que satisface todas las ecuaciones.

Existen varios métodos de resolución:

- Si tenemos la ecuación

$$r_i = \alpha_i + \beta_1 r_1 + \dots + \beta_n r_n$$

y r_i no aparece en la derecha, entonces se sustituye cada aparición de r_i en las otras ecuaciones por $r_i = \beta r_i + \dots + \beta r_m$ y se aplican las reglas de cálculo de ϵ -r. para definir como una nueva ecuación donde ha desaparecido la variable r_i .

- Si en la parte derecha aparece r_i :

$$r_i = \alpha + \beta r_i$$

donde $\alpha \in \Sigma$ pueden aparecer otras variables, pero no en β , se aplica el teorema de Arden para despejar r_i . Si β no incluye la palabra vacía, entonces la única solución es $r_i = \beta^* \alpha$ y se sustituye r_i en el resto de ecuaciones.

De autómata finito a sistema de ecuaciones

- Se puede aplicar a autómatas no deterministas, pero no con transiciones nulas, ya que podría dar lugar a ecuaciones donde no existe una única solución.
- Hay una variable ϵ -r. " q_i " para cada estado q_i , que indica el conjunto de palabras tales que leyéndose desde el estado q_i permiten llegar a un estado final.
- Por cada estado q_i se asocia una ecuación con " q_i " en la parte izquierda y en la que la derecha es una suma que se calcula siguiendo los siguientes términos:
 - Si q_i es final se suma ϵ .
 - Si $q_j \in \delta(q_i, a)$, entonces se suma $a q_j$.
- Una vez resuelto el sistema, la expresión regular del lenguaje asociado al autómata es " q_0 ", donde q_0 es el estado inicial.

Vemos un ejemplo. Dibujemos el siguiente autómata:



Usando las reglas anteriores, las ecuaciones serían:

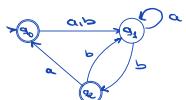
$$\left\{ \begin{array}{l} q_0 = 0q_2 + 1q_3 \\ q_2 = \epsilon + 0q_1 + 1q_3 \\ q_3 = \epsilon + 0q_2 + 1q_2 = \epsilon + (0+1)q_2 \end{array} \right.$$

Resolvemos el sistema:

$$\left. \begin{array}{l} q_0 = \epsilon + (0+1)q_2 \stackrel{1)}{\Rightarrow} q_1 = 0q_2 + 1(\epsilon + (0+1)q_2) \\ \stackrel{2)}{\Rightarrow} q_2 = \epsilon + 0q_1 + 1(\epsilon + (0+1)q_2) \end{array} \right\} \Rightarrow \left. \begin{array}{l} q_1 = 1 + [0 + 1(0+1)]q_2 \\ q_2 = \epsilon + 1 + 0q_1 + 1(0+1)q_2 \end{array} \right. . \quad \text{Ahora, aplicamos la}$$

$$\begin{aligned} & \text{ley de Arden en la segunda ecuación} \Rightarrow q_1 = \underbrace{\epsilon + 0q_1}_{\alpha} + \underbrace{1(0+1)q_2}_{\beta} \Rightarrow q_1 = \alpha + (\alpha + (0+1)\beta)q_2 \\ & \Rightarrow \left. \begin{array}{l} q_1 = \alpha + (0+1)(\epsilon + 0q_1) \\ q_2 = (\epsilon + 0q_1)^*(\epsilon + 1(0+1))q_2 \end{array} \right\} \stackrel{q_2 \rightarrow q_1}{\Rightarrow} q_1 = 1 + (\epsilon + 0q_1)[(\epsilon + 0q_1)^*(\epsilon + 1) + (\epsilon + 0q_1)^* 0q_2] \Rightarrow \\ & \Rightarrow q_1 = \underbrace{\alpha + (0+1)(\epsilon + 0q_1)}_{\alpha} + \underbrace{(\epsilon + 0q_1)^*(\epsilon + 1(0+1))^* 0q_2}_{\beta} \Rightarrow \text{siguiendo la ley de Arden} \Rightarrow \\ & \Rightarrow q_1 = [\epsilon + (0+1)(\epsilon + 0q_1)^*(\epsilon + 1(0+1))]^* (\alpha + (0+1)(\epsilon + 0q_1)^*(\epsilon + 1)) \end{aligned}$$

- ② Dar una expresión regular para el lenguaje aceptado por el siguiente autómata:



Comencemos hallando las ecuaciones iniciales:

$$\left. \begin{array}{l} q_0 = \epsilon + (a+b)q_1 \\ q_1 = \underbrace{aq_1 + bq_2}_{\alpha} \\ q_2 = \epsilon + aq_0 + bq_1 \end{array} \right\} \Rightarrow \text{Usando la ley de Arden en la 2 ecuación} \Rightarrow \left. \begin{array}{l} q_1 = a^* (bq_2) \\ q_2 = \epsilon + aq_0 + bq_1 \end{array} \right\} \stackrel{\text{siguiendo}}{\Rightarrow} \left. \begin{array}{l} q_0 = \epsilon + (a+b)(a^* (bq_2)) \\ q_2 = \epsilon + aq_0 + b(a^* (bq_2)) \end{array} \right\} \stackrel{\text{siguiendo}}{\Rightarrow}$$

$$\left. \begin{array}{l} \Rightarrow q_0 = \epsilon + (a+b)a^* (bq_2) \\ \Rightarrow q_2 = \epsilon + aq_0 + ba^* b^* q_2 \end{array} \right\} \Rightarrow \text{Usando la ley de Arden en } q_2 \Rightarrow q_2 = (ba^* b)^*(\epsilon + aq_0) \Rightarrow$$

$$\Rightarrow q_0 = \epsilon + (a+b)a^* b((ba^* b)^*(\epsilon + aq_0)) \Rightarrow$$

$$\Rightarrow q_0 = \epsilon + (a+b)a^* b((ba^* b)^* + (ba^* b)^* aq_0) \Rightarrow$$

$$\Rightarrow q_0 = \underbrace{\epsilon + (a+b)a^* b((ba^* b)^*)}_{\alpha} + \underbrace{(a+b)a^* b((ba^* b)^* aq_0)}_{\beta}$$

Aplicando la ley de Arden, tenemos que el resultado final es: $((a+b)a^* b((ba^* b)^*)^* (\epsilon + (a+b)a^* b((ba^* b)^*))^* = q_0$

Gramáticas regulares o tipo 3

Existen dos tipos de gramáticas regulares:

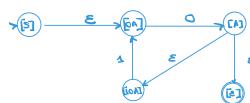
- Líneales por la derecha: cuando todas las producciones tienen la forma $A \rightarrow uB$ o $A \rightarrow u$
- Líneales por la izquierda: cuando todas las producciones tienen la forma $A \rightarrow Bu$ o $A \rightarrow u$

2) Es una gramática regular \Leftrightarrow Existe un autómata finito determinista que lo reconoce.

Sea la gramática líneal por la derecha:

$$S \rightarrow aA \\ A \rightarrow dA \cup \epsilon$$

podemos obtener el siguiente autómata



En el caso de ser líneal por la izquierda invertimos las producciones, hallamos el autómata γ , por último, invertimos el autómata:

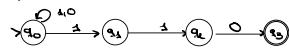


② Dado el lenguaje

$$L = \{a, b, c\}^* : ac \in \{3, 0\}^*$$

encontrar la expresión regular, la gramática líneal por la derecha, la gramática líneal por la izquierda y el AFD asociado.

Comentemos por el autómata finito asociado. Usaremos uno no determinista por simplicidad:



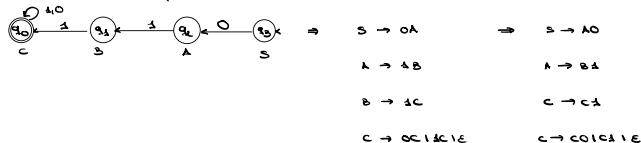
Para hallar la expresión regular, extraemos un sistema de ecuaciones del autómata:

$$\begin{aligned} q_0 &= (0+1)q_0 + \epsilon q_1 \\ q_1 &= \epsilon q_2 \\ q_2 &= 0q_3 \\ q_3 &= \epsilon \end{aligned} \quad \left| \begin{array}{l} \Rightarrow q_0 = (0+1)q_0 + \epsilon q_1 \\ \Rightarrow q_1 = \epsilon q_2 \\ q_2 = 0 \end{array} \right. \quad \left| \begin{array}{l} \Rightarrow q_0 = (0+1)q_0 + 1q_1 \\ q_1 = 0 \end{array} \right. \quad \left| \begin{array}{l} \Rightarrow q_0 = (0+1)q_0 + 1q_1 \\ \Rightarrow q_0 = (0+1)^*q_0 + 110 \stackrel{L.A.}{=} q_0 = (0+1)^*110 \end{array} \right.$$

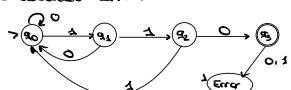
La gramática líneal por la derecha vendría determinada por:

$$S \rightarrow 0S1AS1\bar{A}S$$

Mientras que la líneal por la izquierda viene dada por:



Por último, un AFD asociado sería:



③ Dado un AFD, determinar el proceso que habría que seguir para construir una gramática líneal por la izquierda capaz de generar el lenguaje aceptado por dicho autómata.

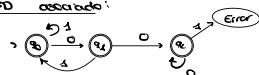
1) Invertir el autómata

2) Construir una gramática líneal por la derecha.

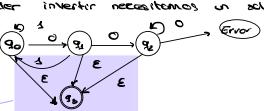
3) Invertir las reglas de producción.

④ Construir un autómata finito determinista que acepte el lenguaje de todos los palabras sobre el alfabeto $\{0, 1\}$, que no contengan la subcadena 001 .

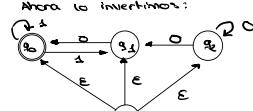
AFD asociado:



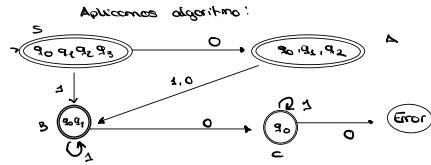
Para poder invertir necesitamos un solo estado final:



Ahora lo invertiremos:



Para poder aplicar el algoritmo necesitamos un único estado final, luego creamos un estado auxiliar, y lo traspasaremos con producciones nulas.



El algoritmo funciona de forma que:

- 1) Tomamos como nodo inicial la unión de todos los que el nodo inicial original puede ir directamente con transiciones nulas, y el mismo.
- 2) Consideramos todas las transiciones. Los nodos de llegada son la unión de los de llegada original, más posibles concatenaciones del tipo $(q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 = q_0 \xrightarrow{ab} q_2)$
(si contiene al nodo final se marca como estado final)

Luego, la gramática generadora (en la forma) sería:

$$S \rightarrow 0A1B1C$$

$$A \rightarrow 0A1B1C$$

$$B \rightarrow 1B1C1E$$

$$C \rightarrow 1C1E$$

Gramática por la izquierda \Rightarrow

$$S \rightarrow A01B1C$$

$$A \rightarrow A01B1C$$

$$B \rightarrow B1C1E$$

$$C \rightarrow C1E$$

16) Sea $B_n = f\alpha^n : K$ múltiplo de n . Demostrar que B_n es regular para todo n .

Como K es múltiplo de n , podemos reescribirlo como $k \in \mathbb{N}$. Luego, dado $n \in \mathbb{N}$, $B_n = f\alpha^{nk} = f(\alpha^n)^k : k \in \mathbb{N}$

- * $n=0 \Rightarrow B_0 = f \in f \Rightarrow q_0 \xrightarrow{\epsilon} q_1$
- * $n=1 \Rightarrow B_1 = f\alpha, \alpha, \alpha^2, \alpha^3, \dots, f \Rightarrow q_0 \xrightarrow{\epsilon} q_1 \xrightarrow{\alpha} q_2$
- * $n=2 \Rightarrow B_2 = f\epsilon, \alpha^2, \alpha^4, \dots, f$

Basicamente, podemos representar B_n con la expresión regular α^* .