GRAMATICAS REGULARES - EXPRESIONES REGULARES

Gramáticas

Las gramáticas formales definen un lenguaje describiendo cómo se pueden generar las cadenas del lenguaje.

Una gramática formal es una cuadrupla G = (N, T, P, S) donde

- N es un conjunto finito de símbolos no terminales
- T es un conjunto finito de símbolos terminales $N \cap T = \varnothing$
- P es un conjunto finito de producciones

  Cada producción de P tiene la forma

  $\alpha \rightarrow \beta,\ \alpha = \varphi A \rho$ y $\beta = \varphi \omega \rho$

  $\varphi, \omega, \rho \in (N \cup T)^*$ y A es S ó $A \in N$
- S es el símbolo distinguido o axioma $S \notin (N \cup T)$

Restringiendo los formatos de producciones permitidas en una gramática, se pueden especificar cuatro tipos de gramáticas (tipo 0, 1, 2 y 3) y sus correspondientes clases de lenguajes.

Gramáticas regulares (Tipo 3)

Generan los lenguajes regulares (aquellos reconocidos por un autómata finito). Son las gramáticas más restrictivas. El lado derecho de una producción debe contener un símbolo terminal y, como máximo, un símbolo no terminal. Estas gramáticas pueden ser:

- Lineales a derecha, si todas las producciones son de la forma

$$A \rightarrow aB \quad \text{ó} \quad A \rightarrow a \qquad \begin{cases} A \in N \cup \{S\} \\ B \in N \\ a \in T \end{cases}$$

(en el lado derecho de las producciones el símbolo no terminal aparece a la derecha del símbolo terminal)

- Lineales a izquierda, si todas las producciones son de la forma

$$A \rightarrow Ba \quad \text{ó} \quad A \rightarrow a \qquad \begin{cases} A \in N \cup \{S\} \\ B \in N \\ a \in T \end{cases}$$

(en el lado derecho de las producciones el símbolo no terminal aparece a la izquierda del símbolo terminal)

En ambos casos, se puede incluir la producción $S \rightarrow \varepsilon$, si el lenguaje que se quiere generar contiene la cadena vacía.

Por ejemplo las siguientes gramáticas G_1 y G_2, son gramáticas regulares lineales a derecha y lineales a izquierda respectivamente, que generan el lenguaje $L = \{a^{2n} / n \geq 0\}$

$G_1 = (\{A, B\}, \{a\}, P_1, S_1)$ donde P_1 es el cjto.

$S_1 \rightarrow \varepsilon$
$S_1 \rightarrow aA$
$A \rightarrow aB$
$A \rightarrow a$
$B \rightarrow aA$

$G_2 = (\{C, D\}, \{a\}, P_2, S_2)$ donde P_2 es el cjto.

$S_2 \rightarrow \varepsilon$
$S_2 \rightarrow Ca$
$C \rightarrow Da$
$C \rightarrow a$
$D \rightarrow Ca$

Algoritmo para obtener la gramática regular desde el autómata finito

Existe un algoritmo que permite obtener una gramática regular que genera un lenguaje regular dado a partir del autómata finito que reconoce ese lenguaje. Los pasos a seguir son los siguientes:

1) Asociar al estado inicial el símbolo distinguido S.
2) Asociar a cada estado del autómata (menos el estado inicial) un símbolo no terminal. Si al estado inicial llega algún arco asociar también un símbolo no terminal (además del símbolo distinguido). No asociar símbolo no terminal a aquellos estados finales de los que no salen arcos.
3) Para cada transición definida $\delta (e_i, a) = e_j$, agregar al conjunto de producciones, la producción $A \rightarrow aB$, siendo A y B los símbolos no terminales asociados a e_i y e_j respectivamente. Si e_j es un estado final, agregar también la producción $A \rightarrow a$. Si e_j es el estado inicial (tiene dos símbolos asociados, el distinguido y un no terminal), utilizar el símbolo no terminal (de esta manera se evita que el símbolo distinguido aparezca a la derecha de una producción).
4) Si el estado inicial es también final agregar la producción $S \rightarrow \varepsilon$.

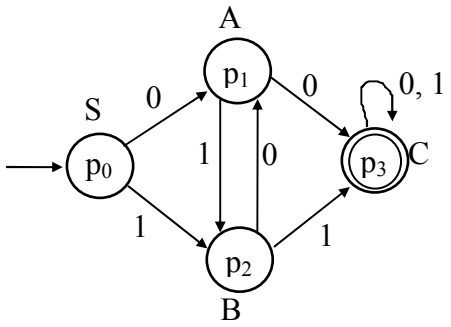
Ejemplo 1:

Derivación de la gramática correspondiente al lenguaje del ej. 4 del apunte de autómatas finitos

$L_4 = \{ x / x \in \{0, 1\}^* \text{ y x contiene la subcadena 00 ó x contiene la subcadena 11}\}$

$L_4 = L(M_{4Dmin})$, $M_{4Dmin} = < \{p_0, p_1, p_2, p_3\}, \{0, 1\}, \delta, p_0, \{p_3\}>$

δ está definida por el siguiente diagrama de transición de estados



Como al estado inicial no entran arcos, se asocia únicamente el símbolo distinguido S.

La gramática correspondiente a este lenguaje es

G = ({A, B, C}, {0, 1}, P, S), siendo P el siguiente conjunto:

$S \rightarrow 0A$ ya que $\delta (p_o, 0) = p_1$ y S y A están asociado a p_0 y p_1 respectivamente.

$S \rightarrow 1B$ ya que $\delta (p_o, 1) = p_2$ y S y B están asociado a p_0 y p_2 respectivamente.

$A \rightarrow 0C$

$A \rightarrow 0$

$A \rightarrow 1B$

$B \rightarrow 0A$

$B \rightarrow 1C$

$B \rightarrow 1$

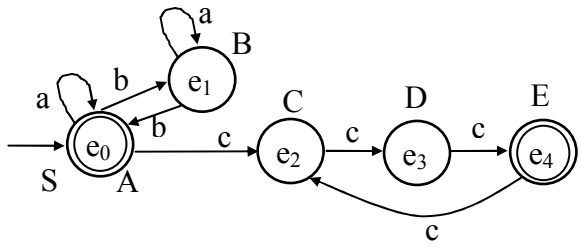
$C \rightarrow 0C$

$C \rightarrow 0$

$C \rightarrow 1C$

$C \rightarrow 1$

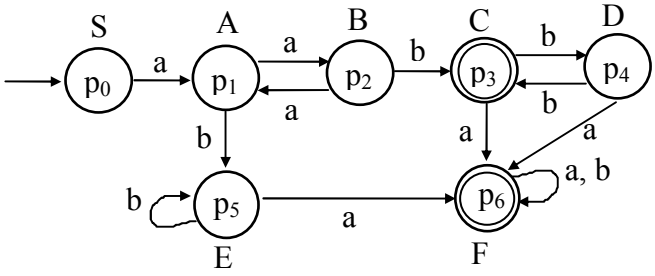
Ejemplo 2:
Derivación de la gramática correspondiente al lenguaje del ej. 3 del apunte de autómatas finitos.
$L_3 = \{xc^{3m} / x \in \{a, b\}^*$ y la cantidad de b's es par y $m \geq 0\}$, siendo $L_3 = L(M_{3D})$
$M_{3D} = < \{e_0, e_1, e_2, e_3, e_4\}, \{a, b, c\}, \delta_{3D}, e_0, \{e_0, e_4\}>$
δ_{3D} está definida por el siguiente diagrama de transición de estados



Como al estado inicial entran arcos, se asocia el símbolo distinguido S y además un símbolo no terminal A.
La gramática correspondiente a este lenguaje es
G = ({A, B, C, D, E}, {a, b, c}, P, S), siendo P el siguiente conjunto:

$S \rightarrow \varepsilon$ (el estado inicial es también final)	$A \rightarrow cC$
$S \rightarrow aA$	$B \rightarrow aB$
$S \rightarrow a$	$B \rightarrow bA$ (se usa el símbolo no terminal asociado al estado inicial)
$S \rightarrow bB$	$B \rightarrow b$
$S \rightarrow cC$	$C \rightarrow cD$
$A \rightarrow aA$	$D \rightarrow cE$
$A \rightarrow a$	$D \rightarrow c$
$A \rightarrow bB$	$E \rightarrow cC$

Ejemplo 3:
Derivación de la gramática correspondiente al lenguaje del ej. 7 del apunte de autómatas finitos.
$L_7 = \{ a^{2n}b^{2k+1} / n \geq 1$ y $k \geq 0\} \cup \{ax / x \in \{a, b\}^*$ y x contiene la subcadena ba$\}$
siendo $L_7 = L(M_{7Dmin})$, $M_{7Dmin} = < \{p_0, p_1, p_2, p_3, p_4, p_5, p_6\}, \{a, b\}, \delta, p_0, \{p_3, p_6\}>$
δ está definida por el siguiente diagrama de transición de estados



La gramática correspondiente a este lenguaje es
G = ({A, B, C, D, E, F}, {a, b}, P, S), siendo P el siguiente conjunto:

$S \rightarrow aA$	$B \rightarrow bC$	$C \rightarrow a$	$D \rightarrow a$	$F \rightarrow aF$
$A \rightarrow aB$	$B \rightarrow b$	$D \rightarrow bC$	$E \rightarrow bE$	$F \rightarrow a$
$A \rightarrow bE$	$C \rightarrow bD$	$D \rightarrow b$	$E \rightarrow aF$	$F \rightarrow bF$
$B \rightarrow aA$	$C \rightarrow aF$	$D \rightarrow aF$	$E \rightarrow a$	$F \rightarrow b$

Expresiones regulares

Se denominan expresiones regulares sobre un alfabeto A, a las expresiones que se pueden construir a partir de las siguientes reglas:

- $\varnothing$ es una expresión regular que describe el lenguaje vacío;
- ε es una expresión regular que describe el lenguaje $\{\varepsilon\}$, esto es el lenguaje que contiene únicamente la cadena vacía;
- Para cada símbolo $a \in A$, a es una expresión regular que describe el lenguaje $\{a\}$, esto es el lenguaje que contiene únicamente la cadena a;
- Si r y s son expresiones regulares que describen los lenguajes L(r) y L(s) respectivamente:

i) $r + s$ es una expresión regular que describe el lenguaje $L(r) \cup L(s)$

ii) $r \,.\, s$ es una expresión regular que describe el lenguaje $L(r) \,.\, L(s)$

iii) r^* es una expresión regular que describe el lenguaje $L(r)^*$.

El operador de clausura es el que tiene mayor precedencia, seguido por el operador de concatenación y por último el operador de unión.

Las expresiones regulares describen los lenguajes regulares (aquellos reconocidos por autómatas finitos).

Por ejemplo las siguientes son expresiones regulares válidas:

- $a^* \,.\, b$ que describe el lenguaje $L = \{a^n b / n \geq 0\}$
- $(a + b)^*$ que describe el lenguaje $L = \{ x / x \in \{a, b\}^* \}$

Leyes algebraicas para expresiones regulares

Dos expresiones regulares r y s son equivalentes ($r \equiv s$) si $L(r) = L(s)$

Sean r, s y t expresiones regulares:

1) $r + \varnothing \equiv \varnothing + r \equiv r$
2) $r \,.\, \varepsilon \equiv \varepsilon \,.\, r \equiv r$
3) $r \,.\, \varnothing \equiv \varnothing \,.\, r \equiv \varnothing$
4) $r + s \equiv s + r$
5) $(r + s) + t \equiv r + (s + t)$
6) $(r \,.\, s) \,.\, t \equiv r \,.\, (s \,.\, t)$
7) $r \,.\, (s + t) \equiv r \,.\, s + r \,.\, t$
8) $(s + t) \,.\, r \equiv s \,.\, r + t \,.\, r$
9) $r + r \equiv r$
10) $\varnothing^* \equiv \varepsilon$
11) $r \,.\, r^* \equiv r^* \,.\, r$
12) $r \,.\, r^* + \varepsilon \equiv r^*$
13) $(r^* \,.\, s^*)^* \equiv (r + s)^*$
14) $(r^*)^* \equiv r^*$

Construcción de autómatas finitos a partir de expresiones regulares

Los lenguajes descriptos por expresiones regulares son los lenguajes reconocidos por los autómatas finitos. Existe un algoritmo para convertir una expresión regular en el autómata finito no determinístico correspondiente. El algoritmo construye a partir de la expresión regular un autómata con transiciones vacías, es decir un autómata que contiene arcos rotulados con ε. Luego este

autómata con transiciones vacías se puede convertir en un autómata finito sin transiciones vacías que reconoce el mismo lenguaje.

Construcción del autómata finito con transiciones ε

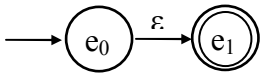
Si r es una expresión regular con n operadores y sin variables como operandos atómicos, existe un autómata finito no determinístico M con transiciones ε (AFND-ε) que acepta solamente aquellas cadenas que están en L(r). M tiene un estado final, no entran arcos al estado inicial y no salen arcos del estado final.

r puede ser una expresión sin operadores (∅, ε o un símbolo) o con operadores (+, ., *).

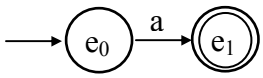
Si r no tiene operadores, entonces:

Para $r = \varnothing$ el AFND-ε es

Para $r = \varepsilon$ el AFND-ε es



Para $r = a$ $(a \in A)$ el AFND-ε es

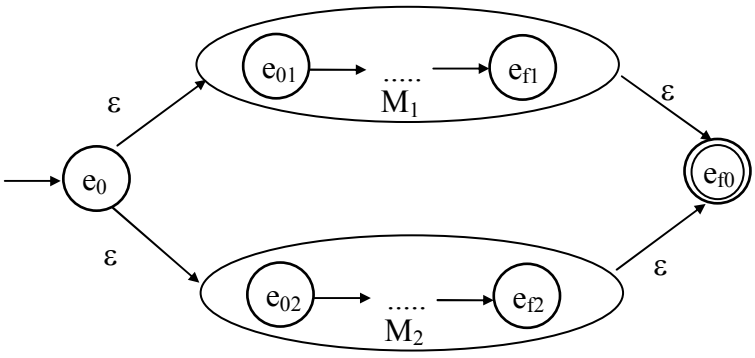


Si r tiene operadores, se dan tres casos dependiendo de la forma de r:

1) $r = r_1 + r_2$

Sean $M_1 = <E_1, A, \delta_1, e_{01}, \{e_{f1}\}>$ y $M_2 = <E_2, A, \delta_2, e_{02}, \{e_{f2}\}>$, los autómatas correspondientes a r_1 y r_2.

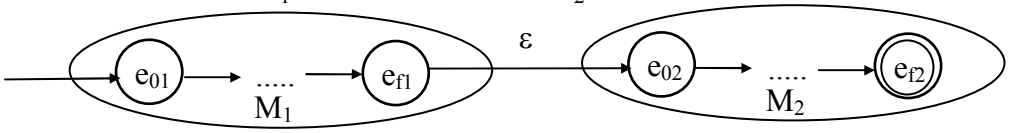
Se construye un nuevo autómata M que une a estos dos autómatas M_1 y M_2 agregando un estado inicial e_0 y un estado final e_{f0}; $M = < E_1 \cup E_2 \cup \{e_0, e_{f0}\}, A, \delta, e_0, \{e_{f0}\}>$. El estado inicial de M tiene transiciones ε a los estados iniciales de M_1 y M_2; los estados finales de estos autómatas tienen transiciones ε al estado final del autómata M.



2) $r = r_1 \cdot r_2$

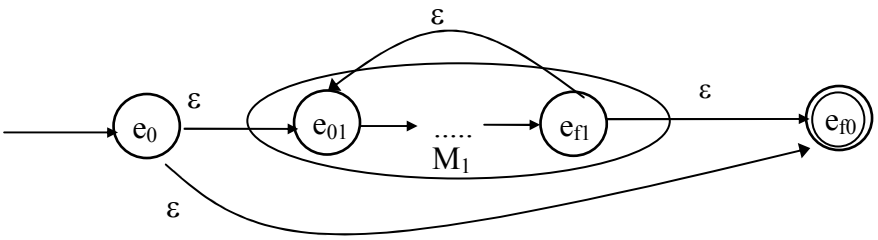
Sean $M_1 = <E_1, A, \delta_1, e_{01}, \{e_{f1}\}>$ y $M_2 = <E_2, A, \delta_2, e_{02}, \{e_{f2}\}>$, los autómatas correspondientes a r_1 y r_2.

Se construye un nuevo autómata M, $M = < E_1 \cup E_2, A, \delta, e_{01}, \{e_{f2}\}>$ que tiene como estado inicial al estado inicial de M_1 y como estado final al estado final de M_2; tiene además un arco rotulado ε desde el estado final de M_1 al estado inicial de M_2.



3) $r = r_1^*$

Sea $M_1 = <E_1, A, \delta_1, e_{01}, \{e_{f1}\}>$ el autómata correspondiente a r_1. Se construye un nuevo autómata M, $M = < E_1 \cup \{e_0, e_{f0}\}, A, \delta, e_0, \{e_{f0}\}>$; y se agregan arcos rotulados ε desde e_0 al estado inicial de M_1 y al estado final de M, y desde el estado final de M_1 al estado inicial de M_1 y a e_{f0}.

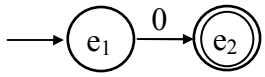


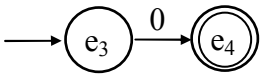
Ejemplo 4:

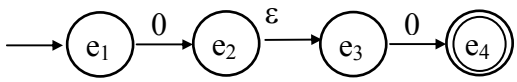
Construir el AFND-ε correspondiente a la siguiente expresión regular $r = 0 \,.\, 0 + 0^* \,.\, 1$

r es de la forma $r_1 + r_2$, donde $r_1 = 0 \,.\, 0$ y $r_2 = 0^* \,.\, 1$

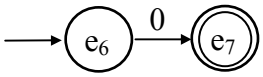
r_1 se puede expresar como $r_3 \,.\, r_4$, donde

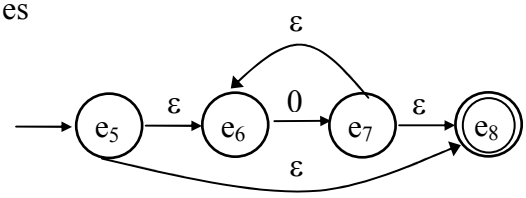
$r_3 = 0$ 

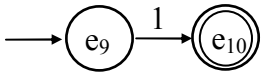
$r_4 = 0$ 

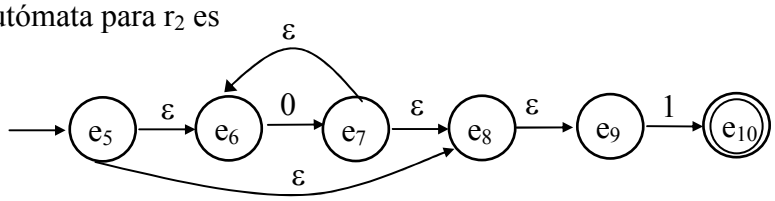
El autómata para r_1 es 

r_2 se puede expresar como $r_5 \,.\, r_6$, donde

$r_5 = r_7^*$ siendo $r_7 = 0$ 

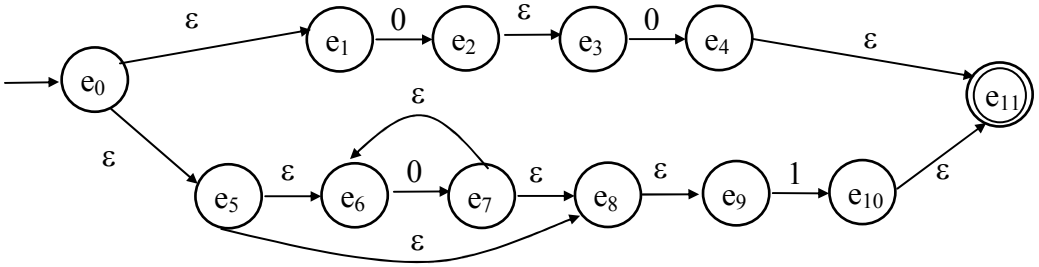
El autómata para r_5 es 

$r_6 = 1$ 

El autómata para r_2 es 

El autómata correspondiente a r es

AFND-ε =<{e_0, e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9, e_{10}, e_{11}}, {0, 1}, δ, e_0, {e_{11}}>, con δ definida por el siguiente diagrama de transición de estados



Construcción del AFND a partir del AFND-ε

Dado un AFND-ε es posible construir un AFND equivalente sin transiciones vacías que reconoce el mismo lenguaje.

Los estados del nuevo autómata son los *estados importantes* del AFND-ε y el estado inicial del AFND-ε. Se denominan estados importantes aquellos estados a los que llega un arco con un símbolo real como rótulo.

El estado inicial es el estado inicial del AFND-ε.

La función de transición se define teniendo en cuenta que existe una transición del estado importante e_i al estado importante e_j con el símbolo x, si existe algún estado e_k tal que:

- se puede llegar desde el estado e_i al estado e_k con un camino de 0 ó mas transiciones ε; se permite $e_i = e_k$;
- en el AFND-ε existe una transición del estado e_k al estado e_j rotulada con el símbolo x.

Los estados finales del nuevo autómata son los estado finales del AFND-ε y todos los estados e_i del AFND-ε para los cuales existe un camino con transiciones ε, en el AFND-ε, a algún estado final del AFND-ε.

Ejemplo 5:

Construcción del AFND correspondiente al AFND-ε del ejemplo 4.

El conjunto de estados está formado por e_0 (estado inicial) y por los estados e_2, e_4, e_7 y e_{10} (estados importantes).

Los estados finales son e_4 y e_{10} (existe un camino en el AFND-ε con transiciones ε a un estado final del AFND-ε).

El estado inicial es e_0.

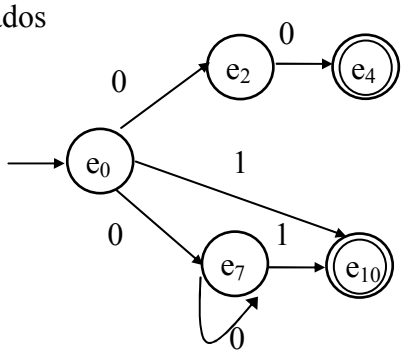
Transiciones para el estado e_0: desde el estado e_0 se pueden alcanzar con transiciones ε los estados e_0, e_1, e_5, e_6, e_8 y e_9. En el AFND-ε existe una transición de e_1 a e_2 con 0, una transición de e_6 a e_7 con 0 y una transición de e_9 a e_{10} con 1. Entonces en el nuevo autómata hay una transición de e_0 a e_2 con 0, una transición de e_0 a e_7 con 0 y una transición de e_0 a e_{10} con 1.

Transiciones para el estado e_2: desde el estado e_2 se pueden alcanzar con transiciones ε los estados e_2 y e_3. En el AFND-ε existe una transición de e_3 a e_4 con 0. Entonces en el nuevo autómata, hay una transición del estado e_2 al estado e_4 con 0.

Transiciones para el estado e_7: desde el estado e_7 se pueden alcanzar con transiciones ε los estados e_6, e_7, e_8 y e_9. En el AFND-ε existe una transición de e_6 a e_7 con 0 y una transición de e_9 a e_{10} con 1. Entonces en el nuevo autómata, hay una transición del estado e_7 al estado e_7 con 0 y una transición del estado e_7 al estado e_{10} con 1.

Transiciones para el estado e_4: desde el estado e_4 se pueden alcanzar con transiciones ε los estados e_4 y e_{11}. Como en el AFND-ε no existen transiciones sobre símbolos reales desde el estado e_{11}, entonces no se agregan transiciones desde el estado e_4 en el nuevo autómata.

Transiciones para el estado e_{10}: desde el estado e_{10} se pueden alcanzar con transiciones ε los estados e_{10} y e_{11}. Como en el AFND-ε no existen transiciones sobre símbolos reales desde el estado e_{11}, entonces no se agregan transiciones desde el estado e_{10} en el nuevo autómata.
El autómata correspondiente sin transiciones ε se define
AFND = $<\{e_0, e_2, e_4, e_7, e_{10}\}, \{0, 1\}, \delta, e_0, \{e_4, e_{10}\}>$, con δ definida por el siguiente diagrama de transición de estados



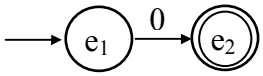
Se puede observar que este autómata es no determinístico. Como ya se ha visto, es posible construir a partir del mismo el autómata finito determinístico equivalente que reconoce el mismo lenguaje.

Ejemplo 6
Construir el AFND-ε correspondiente a la siguiente expresión regular $r = (0 + 0^*) \cdot 1$

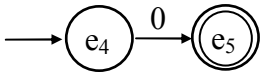
r es de la forma $r_1 \cdot r_2$, donde $r_1 = 0 + 0^*$ y $r_2 = 1$

r_1 se puede expresar como $r_3 + r_4$, donde

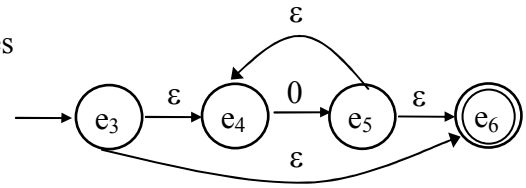
$r_3 = 0$



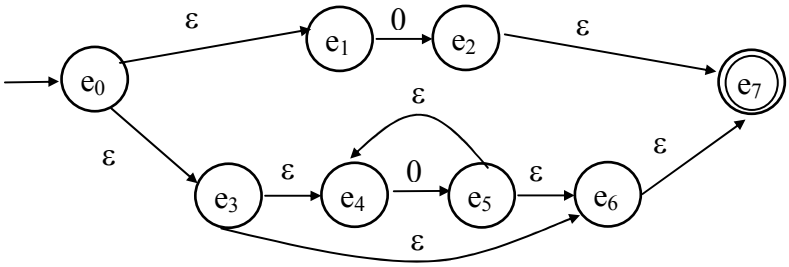
$r_4 = r_5^*$ siendo $r_5 = 0$



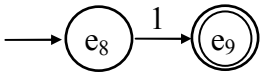
El autómata para r_4 es



El autómata para r_1 es

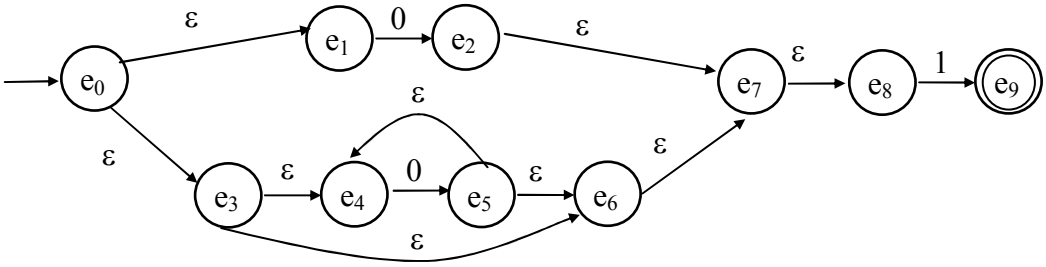


El autómata para r_2 es



El autómata correspondiente a r es

AFND-ε =<{e_0, e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9}, {0, 1}, δ, e_0, {e_9}>, con δ definida por el siguiente diagrama de transición de estados



Construcción del AFND correspondiente al AFND-ε anterior

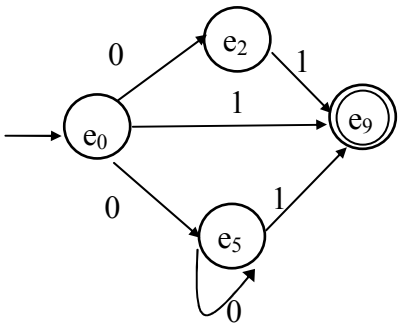
El conjunto de estados está formado por e_0 (estado inicial) y por los estados e_2, e_5 y e_9 (estados importantes).

El único estado final es e_9

El estado inicial es e_0.

El autómata correspondiente sin transiciones ε se define

AFND = <{e_0, e_2, e_5, e_9}, {0, 1}, δ, e_0, {e_9}>, con δ definida por el siguiente diagrama de transición de estados



Se puede observar que este autómata es no determinístico. Como ya se ha visto, es posible construir a partir del mismo el autómata finito determinístico equivalente que reconoce el mismo lenguaje, que podría ser luego minimizado.

Otra posibilidad es simplificar la expresión regular primero para obtener directamente el AFD mínimo.

$r = (0 + 0^*) \,.\, 1$

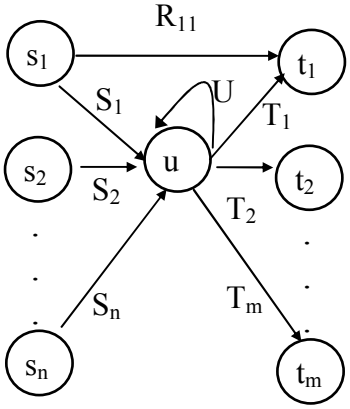
$$
\begin{aligned}
(0 + 0^*) \,.\, 1 &\equiv (\, 0 \,.\, \varepsilon + 0 \,.\, 0^* + \varepsilon) \;\; .\, 1 \\
&\equiv (0 \,.\, (\, \varepsilon + 0^* \,) + \varepsilon \,) \,.\, 1 \\
&\equiv (0 \,.\, (\, \varepsilon + 0 \,.\, 0^* + \varepsilon \,) + \varepsilon \,) \,.\, 1 \\
&\equiv (0 \,.\, (0 \,.\, 0^* + \varepsilon \,) + \varepsilon \,) \,.\, 1 \\
&\equiv (0 \,.\, 0^* \; + \varepsilon \,) \,.\, 1 \\
&\equiv 0^* \,.\, 1
\end{aligned}
$$

Construcción de la expresión regular a partir del autómata

A partir de cualquier autómata finito es posible obtener una expresión regular que describe el lenguaje reconocido por el autómata. Esta conversión consiste en ir eliminando los estados del autómata uno por uno, reemplazando los rótulos sobre los arcos, que inicialmente son símbolos, por expresiones regulares más complicadas.

Eliminación de estados del autómata:

Se desea eliminar el estado u, pero se deben mantener los rótulos de las expresiones regulares sobre los arcos de modo tal que los rótulos de los caminos entre cualesquier par de estados de los estados restantes no cambien.



Si no existe arco de u a u se puede agregar uno rotulado ∅.

Los nodos s_i, para i = 1, 2, ..., n, son nodos predecesores del nodo u, y los nodos t_j, para j = 1, 2, ..., m, son nodos sucesores del nodo u.

Existe un arco de cada s_i a u, rotulado por una expresión regular S_i, y un arco de u a cada t_j rotulado por una expresión regular T_j. Si se elimina el nodo u, estos arcos y el arco rotulado U desaparecerán. Para preservar estas cadenas, se debe considerar cada par s_i y t_j y agregar al rótulo del arco de s_i a t_j, una expresión regular que represente lo que desaparece.

En general se puede suponer que existe un arco rotulado R_{ij} de s_i a t_j para i = 1, 2, ..., n y j = 1, 2, ..., m. Si el arco de s_i a t_j no está presente se puede agregar con rótulo ∅.

El conjunto de cadenas que rotulan los caminos de s_i a u, incluyendo el ciclo de u a u, y luego de u a t_j, se puede describir por la expresión regular $S_iU^*T_j$.

Por lo tanto, después de eliminar u y todos los arcos que llegan y salen de u, se debe reemplazar el rótulo R_{ij} del arco de si a t_j por la expresión regular

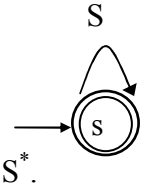
$$R_{ij} + S_iU^*T_j$$

Algoritmo para construir la expresión regular a partir del autómata:

Los pasos a seguir son los siguientes:

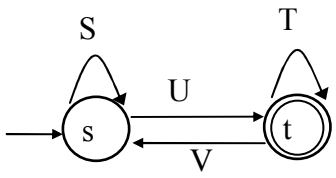
1) Repetir para cada estado final:

- Si el estado final es también inicial, eliminar todos los estados excepto el estado inicial.



La expresión regular correspondiente es S^*.

- Sino, eliminar los estados del autómata hasta que queden únicamente el estado inicial y el estado final en consideración.

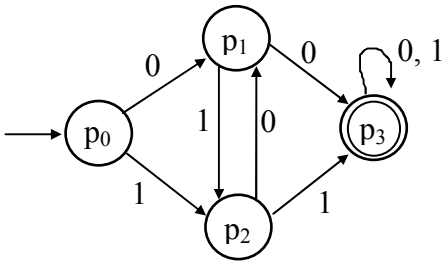


La expresión regular que nos lleva del estado s al estado t es

$$S^* U (T + V S^* U)^* \equiv S^* U (T^* (V S^* U)^*)^*$$

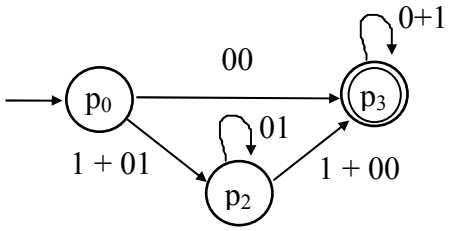
2) Realizar la unión de las expresiones regulares obtenidas para cada estado final del autómata.

Ejemplo 7:
Obtener la expresión regular correspondiente al lenguaje del ejemplo 1

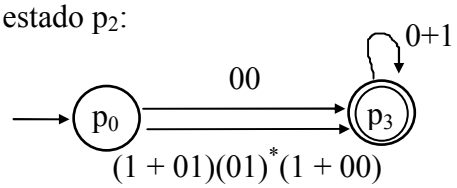


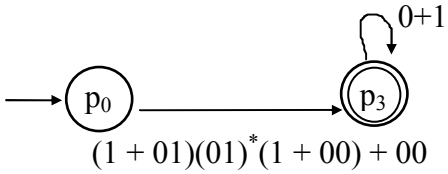
Paso 1) El único estado final es p_3. Como no es inicial se deben eliminar los estados p_1 y p_2 para que queden únicamente p_0 (el estado inicial) y p_3 (el estado final en consideración).

- Eliminación del estado p_1:



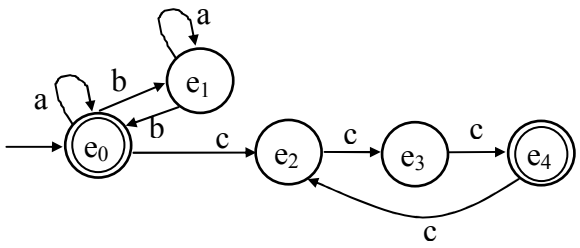
- Eliminación del estado p_2:





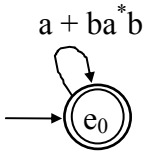
Paso 2) La expresión regular correspondiente es: $((1 + 01)(01)^*(1 + 00) + 00)(0 + 1)^*$

Ejemplo 8:
Obtener la expresión regular correspondiente al lenguaje del ejemplo 2



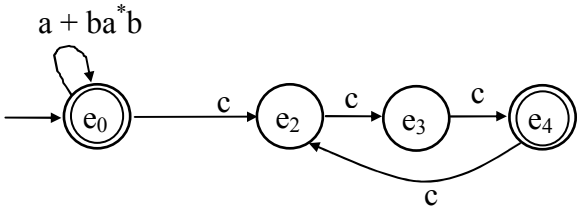
Paso 1) Como el autómata tiene dos estados finales, se calculará una expresión regular para cada uno de ellos.

El estado final e_0 es también inicial. Por lo tanto se deberán eliminar todos los estados que están en el camino de e_0 a e_0. El único estado a eliminar es e_1.
- Eliminación del estado e_1:

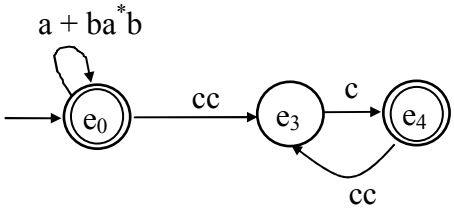


La expresión regular para el estado e_0 es $ER_1 = (a + ba^*b)^*$

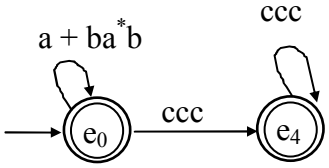
El otro estado final es e_4. Como no es inicial se deben eliminar los estados e_1, e_2 y e_3 para que queden únicamente e_0 (el estado inicial) y e_4 (el estado final en consideración).
- Eliminación de e_1



- Eliminación de e_2



- Eliminación de e_3

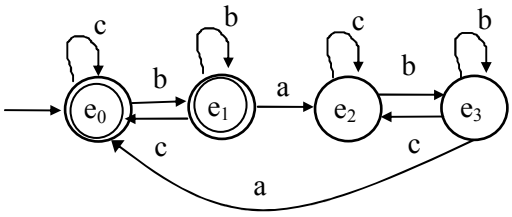


La expresión regular para el estado e_4 es $ER_2 = (a + ba^*b)^*ccc(ccc)^*$

Paso 2) La expresión regular correspondiente al autómata se obtiene uniendo las expresiones regulares resultantes para cada estado final.

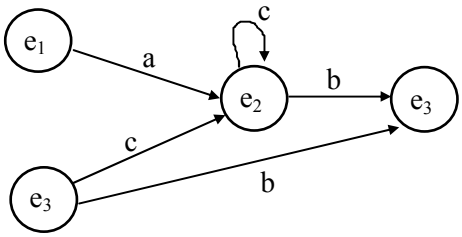
$ER = ER_1 + ER_2 = (a + ba^*b)^* + (a + ba^*b)^*ccc(ccc)^*$

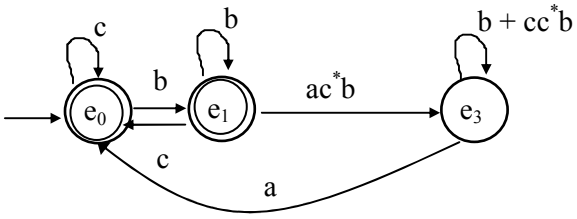
Ejemplo 9:
Obtener la expresión regular correspondiente al lenguaje
L = {x / x ∈ {a, b, c}* y x contiene cantidad par de a´s y cada a en x está precedida por al menos una b}
AFD = <{e_0, e_1, e_2, e_3}, {a, b, c}, δ, e_0, {e_0, e_1}>



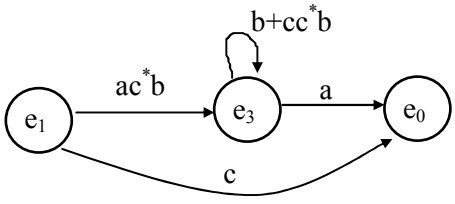
Paso 1) Como el autómata tiene dos estados finales, se calculará una expresión regular para cada uno de ellos.

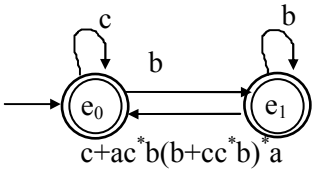
Estado final e_1
Como no es inicial se deben eliminar los estados e_2 y e_3 para que queden únicamente e_0 (el estado inicial) y e_1 (el estado final en consideración).
- Eliminación de e_2





- Eliminación de e_3



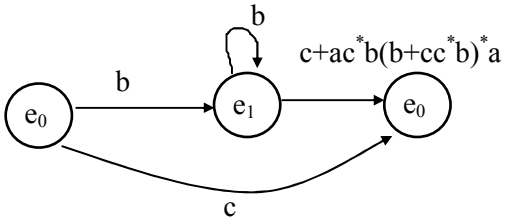


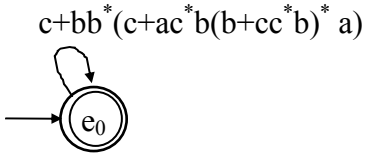
La expresión regular para el estado e_1 es $ER_1 = c^*b(b+(c+ac^*b(b+cc^*b)^*a)c^*b)^*$

Estado final e_0

El estado final e_0 es también inicial. Por lo tanto se deberán eliminar todos los estados que están en el camino de e_0 a e_0, es decir e_1, e_2 y e_3. Retomando el autómata obtenido en el paso anterior, sólo quedaría por eliminar el estado e_1

- Eliminación de e_1





$ER_0 = (c+bb^*(c+ac^*b(b+cc^*b)^*a))^*$

Paso 2) La expresión regular correspondiente al autómata se obtiene uniendo las expresiones regulares resultantes para cada estado final.

$ER = ER_1 + ER_0 = c^*b(b+(c+ac^*b(b+cc^*b)^*a)c^*b)^* + (c+bb^*(c+ac^*b(b+cc^*b)^*a))^*$