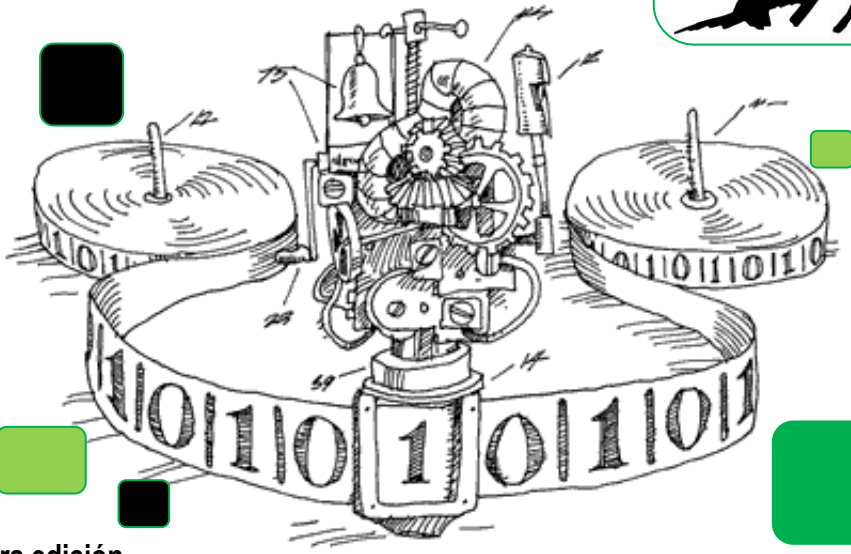
Introducción a la
TEORÍA COMPUTACIONAL



Primera edición

Alcides RAMOS ■ Fredy H. VILLASANTE ■ Teresa P. ALVAREZ ■ Fred TORRES

Alcides Ramos Calcina / Fredy H. Villasante Saravía / Teresa P. Alvarez Rozas / Fred Torres Cruz

INTRODUCCIÓN A LA TEÓRIA COMPUTACIONAL

Puno - Perú

INTRODUCCIÓN A LA TEORÍA COMPUTACIONAL

Autores:
Alcides Ramos Calcina
Fredy Heric Villasante Saravia
Teresa Paola Alvarez Rozas
Fred Torres Cruz

Editado por:
Fredy Heric Villasante Saravia
Jr. Iquitos N° 181, Puno
Puno - Perú

Primera edición digital, enero 2023

Hecho el depósito Legal en la Biblioteca Nacional del Perú

Registro N° 2023- 00290

ISBN: 978-612-00-8280-5

Publicación electrónica en:
https://repositorio.unap.edu.pe/handle/20.500.14082/19253

PRESENTACIÓN

Para la elaboración de la primera parte del texto titulado "Introducción a la Teoría Computacional", se tuvo mucho cuidado en la presentación de los conceptos y/o definiciones de cada uno de los temas tratados en los diferentes capítulos. Este texto está elaborado para los estudiantes que se inician a nivel de pregrado en las ciencias de la computación.

El texto consta de tres capítulos; en la cual se incluye información sobre los temas de Teoría de Autómatas y Lenguajes Formales, correspondiente a las titulaciones de Ingeniero Estadístico e Informática. La asignatura depende de la Facultad de Ingeniería Estadística e Informática de la Universidad Nacional del Altiplano - Puno.

El objetivo del texto es el de tratar de comprender el concepto de sistema informático abstracto, independiente de tecnologías, lo que conlleva el estudio de los autómatas y su jerarquía. Además, deben estudiarse los lenguajes formales como parte básica de la Informática. Los contenidos de esta primera parte son los siguientes: gramáticas y autómatas finitos.

Finalmente agradecemos a las personas que tengan a bien hacer llegar sugerencias o recomendaciones, con el fin de mejorar su contenido.

Puno, marzo del 2022.

INDICE DE CONTENIDO

Capítulo 1

ALFABETOS, CADENAS Y LENGUAJES

INTRODUCCIÓN

La teoría computacional es una ciencia, específicamente una rama de la matemática y de la computación que centra su interés en estudio y definición formal de los cómputos y su principal objetivo es responder ¿cuáles son las capacidades y limitaciones de los ordenadores? Para ello se vale de otras teorías como teoría de autómatas, teoría de computabilidad y teoría de complejidad computacional.

La teoría de autómatas, estudia las máquinas abstractas y los problemas que éstas son capaces de resolver. La teoría de autómatas está estrechamente relacionada con la teoría del lenguaje formal ya que los autómatas son clasificados a menudo por la clase de lenguajes formales que son capaces de reconocer. Tiene un papel central en varias aplicaciones de las ciencias de la computación, incluyendo procesamiento de texto, compiladores, diseño de hardware e inteligencia artificial.

En este capitulo, se presenta las definiciones de los términos más importantes empleados en la teoría de autómata, tales como: alfabeto (un conjunto de símbolos), cadenas de caracteres (una lista de símbolos de un alfabeto) y lenguaje (un conjunto de cadenas de caracteres de un mismo alfabeto).

1. ALFABETOS

Un alfabeto es un conjunto finito no vacío cuyos elementos se llaman símbolos. Denotamos un alfabeto arbitrario con la letra sigma **Σ**. (De Castro, 2004)

Definición. – Una palabra sobre **Σ** es una lista finita de símbolos de **Σ**. Podemos formalmente identificar las listas $x = x_1, x_2, \cdots, x_n$ de símbolos ($x_i \in \sum$) con los elementos de producto cartesiano $\mathbf{\Sigma}^n$.

Ejemplo: sean,

- $\sum_1 = \{0,1\}$, el alfabeto binario, con $1 \in \sum_1$
- $\sum_2 = \{a,b\}$, el alfabeto que consta de dos símbolos *a* y *b*, con $b \in \sum_2$
- $\sum_3 = \{a,b,c,\cdots,x,y,z\}$, el alfabeto de todas las letras minúsculas del idioma castellano. Las palabras del castellano son cadenas sobre Σ.
- $\sum_4 = \{0,1,2,3,4,5,6,7,8,9\}$, el alfabeto de los diez primeros números naturales, con $7 \in \sum_4$
- $\sum_5 = \{0,1,2,3,4,5\}$, el alfabeto de los números en sistema de base 6, con $3 \in \sum_5$
- $\sum_6 = \{p,q,r,(,),\wedge,\vee,\sim,\rightarrow,\leftrightarrow \Delta\}$, el afabeto de tres propocisoones y los conectores lógicos.

Ejemplo: El conjunto de números naturales $\sum = \{0,1,2,3,4,5,6,7,...\}$ no es un alfabeto, dado que es un conjunto infinito.

Nota: Denotamos por $|\sum|$ la longitud de un alfabeto o el cardinal del alfabeto, es decir, el número de elementos del alfabeto, tal que:

- $|\sum| > 0$
- $|\sum| < \infty$

2. CADENA DE CARACTERES

Una cadena o palabra sobre un alfabeto Σ es cualquier sucesión (o secuencia) finita de elementos de Σ. (De Castro, 2004)

Ejemplo: sea,

- $\sum_1 = 0, 00, 001, 010, 1011, 110010000$, son cadenas sobre $\sum_1$

  Observe que 001 ≠ 010. El orden de los símbolos en una cadena es significativo ya que las cadenas se definen como sucesiones, es decir, conjuntos secuencialmente ordenados.
- $\sum_2 = ab, aba, ababaaa, aaaab$, son cadenas sobre $\sum_2$
- $\sum_3 = gato,\ auto,\ universidad,\ computacional$
- $\sum_4 = 101, 105423, 234, 2105, 6, 768$
- $\sum_5 = 102, 331, 55555, 2310001$
- $\sum_6 = p \wedge (p \vee q) \rightarrow r$

2.1. Longitud de cadena

Definición. – La longitud de una cadena $w \in \sum^*$ se denota $|w|$ y se define como el número de símbolos de w (contando los símbolos repetidos). Es decir,

$$|w| = \begin{cases} 0, & \text{si } w = \lambda, \\ n, & \text{si } w = a_1 a_2 \cdots a_n \end{cases}$$

Ejemplo: se tiene,

- $\sum_1 : w = 01101$, entonces $|w| = 5$, Es habitual decir que la longitud de una cadena es igual al "número de símbolos" que contiene; esta proposición está aceptada coloquialmente, sin embargo, no es estrictamente correcta; cuando realmente a lo que se está haciendo referencia es al "número de posiciones".
- $\sum_2 : w = aaab$, entonces $|w| = 4$; $w = baabaab$, entonces $|w| = 7$;
- $\sum_3 : w = abracadabra$, entonces $|w| = 11$

- $\sum_4 : w = 99$, entonces $|w| = 2$
- $\sum_5 : x = 102001$, entonces $|x| = 6$
- $\sum_6 : y = \sim (p \wedge q) \leftrightarrow p$, entonces $|y| = 8$

Ejemplo: Se tiene,

- La secuencia MUNDO es una cadena de longitud 5.
- La secuencia 101001 es una cadena del alfabeto binario de longitud 6.
- La secuencia infinita 0101010101… no es una cadena del alfabeto binario, tampoco lo es la secuencia 00210.

2.2. Cadena vacia

Existe una única cadena que no tiene símbolos, la cual se denomina *cadena vacía* y se denota con λ. La cadena vacía es equivalente al conjunto nulo o vacío $\varnothing$ en la teoría de conjuntos.

- Se sabe que $|\lambda| = 0$, es decir, tiene longitud cero.
- El símbolo λ no pertenece a ningún alfabeto, pero si al meta-alfabeto, $\lambda \notin \sum$.

2.3. Lenguaje universal

Definición. - El conjunto de todas las cadenas sobre un alfabeto $\sum$, incluyendo la cadena vacía (λ), se llama Lenguaje Universal y se denota por $\sum^*$. El lenguaje universal de cualquier alfabeto es infinito.

Ejemplo: sea,

- $\sum_1 = \{a\}$, entonces $\sum_1^* = \{\lambda, a, aa, aaa, aaaa, \cdots\}$
- $\sum_2 = \{a, b, c\}$, entonces
  $\sum_2^* = \{\lambda, a, b, c, aa, ab, ac, ba, bb, bc, ca, cb, cc, aaa, aab, \cdots\}$
- $\sum_3 = \{0,1\}$, entonces
  $\sum_3^* = \{0,1\}^* = \{\lambda, 0, 1, 00, 01, 10, 11, 000, 001, 010, \cdots\}$

Observaciones:

- $\sum \subset \sum^*$
- λ es la cadena o palabra de todo universo, es decir: $\lambda \in \sum^*$
- Si $\sum$ es un conjunto finito, $\sum^*$ es siempre un conjunto infinito (enumerable).
- Hay que distinguir entre los siguientes cuatro objetos, que son todos diferentes entre sí: $\varnothing$, λ, $\{\varnothing\}$ y $\{\lambda\}$.
- La mayor parte de la teoría computacional se hace con referencia a un alfabeto $\sum$ fijo (pero arbitrario).

2.4. Lenguaje formal

Definición. - L definido sobre un alfabeto $\sum$, es un conjunto cualquiera de palabras definidas sobre dicho alfabeto, y se le llama Lenguaje Formal, por lo tanto, $L \subset \sum^*$.

Ejemplo: sea $\sum = \{a, b\}$, entonces se puden definir los siguientes lenguajes:

- $L_1 = \{a, b, bb, bbb\}$
- $L_2 = \{b, bb, bbb, bbbb, bbbbb, \cdots\}$
- $L_3 = \{\lambda, a, ab, aab, aaab, aaaab, \cdots\}$
- $L_4 = \{a, b\} = \sum$
- $L_5 = \varnothing$ (un lenguaje si puede ser vacio)

2.5. Operaciones con cadenas

i) Concatenación de cadenas

Definición. - Dado un alfabeto $\sum$ y dos cadenas u, $v \in \sum^*$, la concatenación de u y v se denota como $u.v$ o simplemente uv.

Dicho de manera más precisa, si u es la cadena compuesta por i símbolos $u = a_1 a_2 a_3 \cdots a_i$ y v es la cadena compuesta por j símbolos $v = b_1 b_2 b_3 \cdots b_j$, entonces uv es la cadena de longitud $i + j$: $w = uv = a_1 a_2 a_3 \cdots a_i b_1 b_2 b_3 \cdots b_j$.

Ejemplo:

- Si $u = 011001$ y $v = 110$, entonces $uv = 011001110$ y $vu = 110011001$.
- Si x = ABRA e y = CADABRA, entonces $w = xy$ = ABRACADABRA
- Si x = some e y = temos, entonces w = sometemos
- Si x = 1234 e y = 4321, entonces w = 12344321

Propiedades:

- ✓ ***Operación cerrada.*** Si x e y están definidas sobre el mismo alfabeto, xy también lo estará.
- ✓ ***Asociativa.*** $(xy)z = x(yz)$
- ✓ ***Elemento neutro.*** $x\lambda = \lambda x = x$
- ✓ ***Conmutativa.*** No es una operación conmutativa.
- ✓ ***Cancelación por la izquierda.*** $\forall x,y,z \in \sum^*$, se cumple: $xy = xz$; entonces $y = z$.
- ✓ ***Cancelación por la derecha.*** $\forall x,y,z \in \sum^*$, se cumple: $xz = yz$; entonces $x = y$.
- ✓ $|xy| = |x| + |y|$

Ejemplo: La cadena λ es subcadena de cualquier cadena, toda cadena es subcadena de sí misma, la cadena 01 es una subcadena de 0001000, la cadena 00 no es una subeadena de 01110.

ii) Potencia de una cadena

Definición. - Dada $w \in \sum^*$ y $n \in \mathbb{N}$, se define (descriptivamente) w^n *en* la siguiente forma:

$$w^0 = \lambda$$

$$w^n = \underbrace{www\cdots w}_{n \text{ veces}}$$

Ejemplo:

- Si $w = la$ sobre el alfabeto $\sum = \{la\}$, entonces

  $w^0 = \lambda$

  $w^1 = la$

  $w^2 = lala$

  $w^3 = lalala$

- . Si $w = 10010$ sobre el alfabeto $\sum = \{10010\}$, entonces

  $w^0 = \lambda$

  $w^1 = 10010$

  $w^2 = 1001010010$

  $w^3 = 100101001010010$

  $w^4 = 10010100101001010010$

Propiedades:

- ✓ $x^{n+m} = x^n x^m$
- ✓ $|x^n| = n|x|$, es semejante a la propiedad de logaritmo.
- ✓ $x^0 = \lambda$

Ejemplo: Si $w \in \sum^*$ y $n,m \in \mathbb{N}$, demostrar que $|w^{n+m}| = |w^n| + |w^m|$.

Caso n, $m \geq 1$.

$$|w^{n+m}| = \underbrace{|www\cdots w|}_{n+m \text{ veces}} = (n+m)|w|$$

por otro lado, se tiene:

$$|w^n| + |w^m| = \underbrace{|www\cdots w|}_{n \text{ veces}} + \underbrace{|www\cdots w|}_{m \text{ veces}} = n|w| + m|w|$$

Caso $n = 0$, $m \geq 1$.

$$|w^{n+m}| = |w^{0+m}| = |w^m| = m|w|$$

por otro lado, se tiene:

$$|w^n| + |w^m| = |w^0| + |w^m| = |\lambda| + |w^m| = 0 + m|w|$$

Caso $n \geq 1$, $m = 0$. Similar al caso anterior.

Caso $n = 0$, $m = 0$.

$$\left|w^{n+m}\right| = \left|w^{0+0}\right| = \left|w^{0}\right| = \left|\lambda\right| = 0$$

por otro lado, se tiene:

$$\left|w^{n}\right| + \left|w^{m}\right| = \left|w^{0}\right| + \left|w^{0}\right| = \left|\lambda\right| + \left|\lambda\right| = 0 + 0 = 0$$

Observación:

Sabemos que $\sum \subset \sum^*$, es decir, el conjunto de todas las cadenas de un alfabeto $\sum$ se designa mediante $\sum^*$. Por consiguiente:

$$\sum^* = \sum^0 \cup \sum^1 \cup \sum^2 \cup \cdots$$

Ejemplo:

Si $\sum = \{0,1\}$, entonces:

- $\sum^0 = \{\lambda\}$, independientementede cuál sea el alfabeto $\sum$.
- $\sum^1 = \{0,1\}$
- $\sum^2 = \{00,01,10,11\}$
- $\sum^3 = \{000,001,010,011,100,101,110,111\}$, etc.

Podrá notar que existe una ligera confusión entre $\sum$ y $\sum^1$. El primero es un alfabeto; sus elementos 0 y 1 son los símbolos. El segundo es un conjunto de cadenas; sus elementos son las cadenas 0 y 1, cuya longitud es igual a 1.

En ese sentido, se puede expresar $\sum^*$ como:

$$\sum^* = \{0,1\}^* = \{\lambda,0,1,00,01,10,11,000,001,010,\cdots\}$$

Observación:

En muchos casos, desearemos excluir la cadena vacía del conjunto de cadenas. El conjunto de cadenas no vacías del alfabeto $\sum$ se designa como $\sum^+$. Por tanto, dos equivalencias apropiadas son:

- $\sum^+ = \sum^1 \cup \sum^2 \cup \sum^3 \cup \cdots$
- $\sum^* = \sum^+ \cup \{\lambda\}$

iii) Reflexión o inversa de una cadena

Definición. – La reflexión o inversa de una cadena $w \in \sum^*$ se denota w^I y se define descriptivamente así:

$$w^I = \begin{cases} \lambda, & \text{si } w = \lambda, \\ a_n \cdots a_2 a_1, & \text{si } w = a_1 a_2 \cdots a_n \end{cases}$$

Ejemplo: sea,

- Si $w = \text{casa}$, entonces $w^I = \text{asac}$
- Si $x = 1000011$, entonces $x^I = 1100001$
- Si $u = seroma$, entonces $w^I = (seroma)^I = amores$

Propiedades:

✓ $\left(w^I\right)^I = w$, para $w \in \sum^*$

✓ $\left|w^I\right| = |w|$, es seme

iv) Subcadenas, prefijos y sufijos

Definición. – Una cadena *v* es una **subcadena** o una **subpalabra** de *u* si existen cadenas *x*, *y* tales que $u = xvy$. Si y solo si $x = \lambda$ o $y = \lambda$ y, por lo tanto, la cadena vacía es una sub cadena de cualquier cadena y toda cadena es sub cadena de sí misma.

Definición. – Un **prefijo** de *u* es una cadena *v* tal que $u = vw$ para alguna cadena $w \in \sum^*$. Se dice que *v* es un **prefijo propio** si $v \neq u$.

Definición. – Un **sufijo** de *u* es una cadena *v* tal que $u = wv$ para alguna cadena $w \in \sum^*$. Se dice que *v* es un **sufijo propio** si $v \neq$ u.

Ejemplo: Sea $\sum = \{a, b, c, d\}$ y $u = bcbaadb$.

Preffijos de *u:*	Sufijos de *u*:
λ	λ
b	*b*
bc	*db*
bcb	*adb*
bcba	*aadb*
bcbaa	*baadb*
bcbaad	*cbaadb*
bcbaadb	*bcbaadb*

Todos son propios, excepto *bcbaadb.*

3. LENGUAJES

Definición. - Un **lenguaje** *L* sobre un alfabeto $\sum$ es un subconjunto de $\sum^*$, es decir $L \subset \sum^*$.

Un lenguaje de $\sum$ no necesita incluir cadenas con todos los símbolos de $\sum$, ya que una vez que hemos establecido que *L* es un lenguaje de $\sum$, también sabemos que es un lenguaje de cualquier alfabeto que sea un superconjunto de $\sum$.

Observaciones:

- $L = \varnothing$, es un lenguaje vacio.
- $\varnothing \neq \{\lambda\}$, el primero no contiene ninguna cadena y el segundo sólo tiene una cadena.
- $L = \sum^*$, es un lenguaje de todas las cadenas sobre $\sum$.
- Se cumple: $\varnothing \subseteq L \subseteq \sum^*$, para todo *L*, y puede ser finito o infinito.
- Los lenguajes se denotan con letras mayúsculas *A, B, C, …, L, M, N, …* En la siguiente figura se visualizan dos lenguajes *A* y *B* sobre $\sum$.

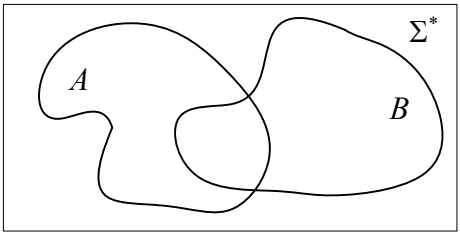


Figura 1.

Ejemplo: Se tiene los lenguajes sobre alfabetos específicos,

- $\sum = \{a,b,c\}$, $L = \{a, aba, aca\}$
- $\sum = \{0,1,2\}$, $L = \{0, 00, 000, \cdots\} = \{0^n : n \geq 1\}$
- $\sum = \{a,b,c,\cdots,x,y,z,A,B,C,\cdots,X,Y,Z\}$, L = $\{w \in \sum^* :$ w pertenece a diccionario españos DRA$\}$. L es un lenguaje finito.
- $\sum = \{a,b,c\}$, $L = \{v \in \sum^* :$ v no tiene el símbolo $c\}$. Ejemplo: $v = abbaab \in L$ y $v = abbcab \notin L$.
- $\sum = \{0,1\}$, L = conjunto de todas las secuencias binarias que continen un número impar de ceros.

3.1. Operaciones con lenguajes

Definición. – Dados dos lenguajes defnidos sobre el mismo alfabeto: $L_1, L_2 \subseteq \sum^*$. La **Unión** de los lenguales L_1 y L_2 se define:

$$L_1 \cup L_2 = \{w \in \sum^* / (w \in L_1) \vee (w \in L_2)\}$$

Propiedades:

- ✓ **Operación cerrada.** Si $L_1, L_2 \subseteq \sum^* \Rightarrow L_1 \cup L_2$, es un lenguaje.
- ✓ **Asociativa.** $(L_1 \cup L_2) \cup L_3 = L_1 \cup (L_2 \cup L_3)$
- ✓ **Conmutativa.** $L_1 \cup L_2 = L_2 \cup L_1$
- ✓ **Elemento neutro.** $L \cup \varnothing = \varnothing \cup L = L$
- ✓ **Idenpotencia.** $L \cup L = L$

Ejemplo: Si, $L_1 = \{a\}$ y $L_2 = \{b\}$, halle $L_1 \cup L_2$

$$L_1 \cup L_2 = \{a\} \cup \{b\} = \{a, b\}$$

Ejemplo: Si, $L_1 = \{a, ab\}$ y $L_2 = \{ab, aab, aaabb\}$, halle $L_1 \cup L_2$

$$L_1 \cup L_2 = \{a, ab\} \cup \{ab, aab, aaabb\} = \{a, ab, aab, aaabb\}$$

Ejemplo: Si, $L = \{0,11\}$ y $M = \{0001,111\}$, es acil verificar que,

$$LM = \{00001, 0111, 110001, 11111\}$$

3.2. Concatenación de lenguajes

Definición. – La **concatenación** de dos lenguajes L_1 y L_2 sobre Σ, denotada $L_1 . L_2$ o simplemente $L_1 L_2$ se define como:

$$L_1.L_2 = \{u.v \in \Sigma^* / (u \in L_1) \wedge (v \in L_2)\}$$

Observación:

En general $L_1.L_2 \neq L_2.L_1$

Ejemplo:

- Si $\Sigma = \{a, b, c, 0, 1\}$, $L_1 = \{a, ab, ac\}$ y $L_2 = \{0, 10\}$ entonces

$$L_1 L_2 = \{a0, a10, ab0, ab10, ac0, ac10\}$$

$$L_2 L_1 = \{0a, 0ab, 0ac, 10a, 10ab, 10ac\}$$

- Si $\Sigma = \{a, b, c\}$, $L_1 = \{a, ab, bc\}$ y $L_2 = \{b, b^2\}$ entonces

$$L_1 L_2 = \{ab, ab^2, ab^2, ab^3, bcb, bcb^2\}$$

$$L_2 L_1 = \{ba, bab, b^2c, b^2a, b^2ab, b^3c\}$$

Propiedades: Sean los lenguajes L_1, L_2, L_3 sobre $\sum$.

- ✓ $L_1.\varnothing = \varnothing.L_1 = \varnothing$
- ✓ $L_1.\{\lambda\} = \{\lambda\}.L_1 = L_1$
- ✓ **Asociativa.** $(L_1.L_2).L_3 = L_1.(L_2.L_3)$
- ✓ **Distributiva respecto a la unión.** $L_1.(L_2 \cup L_3) = L_1L_2 \cup L_1L_3$

3.3. Potencia de un lenguaje

Definición. – Se define recursivamente a partir de la concatenación. Así, dado un lenguje o una palabra $L \subseteq \sum^*$ y un número natural $n \in \mathbb{N}$, definimos la potencia $L^n = \underbrace{L.L.L.\cdots L}_{n \text{ veces}}$, mediante:

$$L^0 = \{\lambda\}$$

$$L^n = L.L^{n-1}, \text{ con } n \geq 1.$$

-

Ejemplo: Sea, $L = \{a, b\}$, determine L^3

- $L^0 = \{\lambda\}$
- $L^1 = L.L^0 = L.\{\lambda\} = L = \{a, b\}$
- $L^2 = L.L^1 = L.L = \{aa, ab, ba, bb\}$
- $L^3 = L.L^2 = L.L.L = \{aaa, aab, aba, abb, baa, bab, bba, bbb\}$

Ejemplo: Sea, $L = \{acc\}$, determine L^4

- $L^0 = \{\lambda\}$
- $L^1 = L.L^0 = L.\{\lambda\} = L = \{acc\}$
- $L^2 = L.L^1 = L.L = \{accacc\}$
- $L^3 = L.L^2 = L.L.L = \{accaccacc\}$
- $L^4 = L.L^3 = L.L.L.L = \{accaccaccacc\} = (acc)^4$

3.4. La clausura de Kleene de un lenguaje

Definición. – La **clausura de Kleene** de un lenguaje, denotado por L^* es el resultado de unir todas las potencias de dicho lenguaje, (Jurado, 2008) es decir:

$$L^* = \bigcup_{i=0}^{\infty} L^i = L^0 \cup L^1 \cup L^2 \cup L^3 \cup \cdots$$

Ejemplo: Sea, $L = \{1\}$, entonces,

- $L^0 = \{\lambda\}$
- $L^1 = \{1\}$
- $L^2 = \{11\}$
- $L^3 = \{111\}$

$\vdots$

La calusura de $L = \{1\}$ es $L^* = \{\lambda, 1, 11, 111, \cdots\}$

Ejemplo: Sea, $L = \{a, b\}$, entonces,

- $L^0 = \{\lambda\}$
- $L^1 = \{a, b\}$
- $L^2 = \{aa, ab, ba, bb\}$
- $L^3 = \{aaa, aab, aba, abb, baa, bab, bba, bbb\}$

$\vdots$

Por tanto, $L^* = \{\lambda, a, b, aa, ab, ba, bb, aaa, aab, aba, abb, baa, \cdots\}$

Definición. – La **clausura positiva** de un lenguaje, denotado por L^+ es la unión de todas las potencias de ese lenguaje, exceptuando la potencia cero.

$$L^+ = \bigcup_{i=1}^{\infty} L^i = L^1 \cup L^2 \cup L^3 \cup \cdots$$

Propiedades: Sea L un lenguaje sobre Σ, es decir, $L \subseteq \Sigma^*$

- ✓ $L^* = L^+ \cup \{\lambda\}$
- ✓ $L^+ = L^* L = L L^*$
- ✓ $\left(L^*\right)^n = L^*$, para todo $n \geq 1$
- ✓ $\left(L^*\right)^* = L^*$
- ✓ $\left(L^*\right)^+ = L^*$
- ✓ $\left(L^+\right)^* = L^*$
- ✓ $\left(L^+\right)^+ = L^+$

Ejemplo: Demuestre $L^+ = L^* L = L L^*$.

Partimos de: $L L^* = L.\left(L^0 \cup L^1 \cup L^2 \cup L^3 \cup \cdots\right)$

$$= L.\left(L^1 \cup L^2 \cup L^3 \cup \cdots\right)$$

$$L L^* = L^+$$

Del mismo modo para $L^+ = L^* L$

Ejemplo: Demuestre $\left(L^*\right)^+ = L^*$.

Se tiene que: $\left(L^*\right)^+ = \left(L^*\right)^1 \cup \left(L^*\right)^2 \cup \left(L^*\right)^3 \cup \cdots$

$$= L^* \cup L^* \cup L^* \cup \cdots$$

$$\left(L^*\right)^+ = L^*$$

Ejemplo: Demuestre $\left(L^{+}\right)^{*} = L^{*}$.

Se tiene que: $\left(L^{+}\right)^{*} = \left(L^{+}\right)^{0} \cup \left(L^{+}\right)^{1} \cup \left(L^{+}\right)^{2} \cup \cdots$

$$= \{\lambda\} \cup L^{+} \cup L^{+}L^{+} \cup \cdots$$

$$= L^{*} \cup \left(\text{conjuntos contenidos en } L^{+}\right)$$

$$\left(L^{*}\right)^{+} = L^{*}$$

3.5. Reflexión o inversa de un lenguaje

Definición. La reflexión de un lenguaje, la cual se denota por L^{I} está formada por las inversas de todas las palabras de ese lenguaje, decir,

$$L^{I} = \left\{w^{I} / w \in L\right\}$$

Propiedades: Sea L_1 y L_2 lenguajes sobre $\sum$, es decir, $L_1, L_2 \subseteq \sum^{*}$

- ✓ $\left(L_1.L_2\right)^{I} = L_2^{I}.L_1^{I}$
- ✓ $\left(L_1 \cup L_2\right)^{I} = L_1^{I} \cup L_2^{I}$
- ✓ $\left(L_1 \cap L_2\right)^{I} = L_1^{I} \cap L_2^{I}$
- ✓ $\left(L^{I}\right)^{I} = L$
- ✓ $\left(L^{*}\right)^{I} = \left(L^{I}\right)^{*}$

EJERCICIOS PROPUESTOS

1. Dar un ejemplo de un alfabeto $\sum$ y dos lenguajes diferentes A, B sobre $\sum$ tales que $AB = BA$.

2. Una de las dos contenencias siguientes es verdadera y la otra es falsa. Demostrar o refutar, según sea el caso:
   a) $A.(B \cap C) \subseteq A.B \cap A.C$
   b) $A.B \cap A.C \subseteq A.(B \cap C)$

3. Sean A = {the, my} y B = {horse, house, hose} lenguajes sobre el alfabeto inglés. Obtener $A.B$, $A.A$ y $A.B.B$.

4. Sean $A = \{\lambda, ab\}$ y $B = \{cd\}$. ¿Cuántas cadenas hay en A^nB para un n arbitrario?

5. Sean A = {a} y B = {b}. Obtener $A^n.B$, AB^n y $(AB)^n$.

6. Demuestre que para cualquier lenguaje L, $L\varnothing = \varnothing L = \varnothing$.

7. Si $L = \{0, 01, 001, 0001, \cdots\}$ calcule L^* .

8. Sean L y M dos lenguajes, demuestre que $(L \cup M)^* = (L^* M^*)^*$

Capítulo 2

GRAMÁTICAS REGULARES

INTRODUCCIÓN

Si nos referimos a los lenguajes naturales el concepto de gramática es muy antiguo. Los primeros trabajos aparecenen la India durante los comienzos del primer milenio antes de Cristo, alcanzándose el máximo apogeo con Panini (siglosVIIyVIa.C.). Al mismo tiempo en Grecia se desarrolla bauna corriente de investigación gramatical, cuyo máximo representante sería Pitágoras. Sin embargo, el concepto degram´atica desde un punto de vista formal tiene su origen en los trabajos de Chomsky amediados del siglo XX. (Jurado, 2008)

Incicaremos este capítulo dando una serie de definiciones de las reglas de formación (la gramática) de las expresiones regulares.

1. GRAMÁTICA

1.1. ¿Qué es la Gramática?

Una gramática describe la estructura de las fraces y de las palabras de un lenguaje y se aplica por aigual a:

- Las lenguas naturales humanas
- Lenguajes de programación

Una gramática es un "ente formal" para especificar de manera finita el conjunto de cadenas de símbolos que constituyen un lenguaje.

1.2. Estructura de una grámatica

Las gramáticas están integradas por varios elementos que permiten la estructuración de palabras, por tanto,

---

Definición. Una gramática formal o gramática G es una cuaterna definido como:

$$G = (\sum_N, \sum_T, S, P)$$

Donde:

$\sum_N$: *Alfabeto de símbolos no terminales*. Es un conjunto finito llamado alfabeto de símbolos no terminales.

$\sum_T$: *Alfabeto de símbolos terminales*. Es un conjunto finito no vació símbolos terminales, que verifica $\sum_N \cap \sum_T = \varnothing$.

S : *Axioma*. Símbolo especial, que se denomina símbolo inicial, donde $S \in \sum_N$.

P : *Regla de producciones*. Es un conjunto finito llamado conjunto de producciones (o, simplemente, sistema de reescritura), donde $S \in \sum_N$.

Observaciones:

- Un símbolo es cualquier carácter o figura, por ejemplo: a, b, p, 3, 5, #, &, @, etcétera.
- Se cumple entre los alabetos de G: $\sum_N \cup \sum_T = \sum$

Ejemplo: Sea la gramática $G = (\sum_N, \sum_T, S, P)$, donde: $\sum_N = \{S\}$, $\sum_T = \{a, b\}$, $P = \{S \to ab, S \to aSb\}$ y S = estado inicial. Genere la palabra *aaaabbbb*.

Se tiene las producciones:

P: $S \to ab$

$S \to aSb$

Generando se tiene: $S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaaSbbb \Rightarrow aaaabbbb$

Ejemplo: Sea la gramática $G = \left(\{S, A, B\}, \{a, b, c, d\}, S, P\right)$, donde: P son las producciones:

$S \to ASB$

$A \to b$

$aaA \to aaBB$

$S \to d$

$A \to aA$

$B \to dcd$

Genere las palabras: bddcd, abddcd, aabddcd, bbaddcddcdcd…

bddcd	*abddcd*	*aabddcd*
$S \to ASB$	$S \to ASB$	$S \to ASB$
$S \to bSB$	$S \to aASB$	$S \to aASB$
$S \to bdB$	$S \to abSB$	$S \to aaASB$
$S \to bddcd$	$S \to abdB$	$S \to aabSB$
	$S \to abddcd$	$S \to aabdB$
		$S \to aabddcd$

Queda para el lector generar: bbaddcddcdcd…

Ejemplo: Sea la gramática $G = \left(\sum_N, \sum_T, S, P\right)$, donde: $\sum_N = \{< número >, < dígito >\}$, $\sum_T = \{0,1,2,3,4,5,6,7,8,9\}$, $S =< número >$. Las reglas de producción P son:

$< número > \to < dígito >< número >$

$< número > \to < dígito >$

$< dígito > \to 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$

Genere la palabra: 2019

2019

$< número > \rightarrow < dígito >< número >$

$< número > \rightarrow < dígito >< número >< número >$

$< número > \rightarrow < dígito >< número >< número >< número >$

$< número > \rightarrow < dígito >< dígito >< número >< número >$

$< número > \rightarrow < dígito >< dígito >< dígito >< número >$

$< número > \rightarrow < dígito >< dígito >< dígito >< dígito >$

$< número > \rightarrow 2 < dígito >< dígito >< dígito >$

$< número > \rightarrow 20 < dígito >< dígito >$

$< número > \rightarrow 201 < dígito >$

$< número > \rightarrow 2019$

Genere usted la palabra: 122101

1.3. Notación

a) Vocabulario no terminal

- Letras mayúsculas latinas de comienzo del abecedario: *A*, *B*, *C*, … La única excepción es generalmente representada con *S*.
- Nombres en minúscula, pero encerrados entre paréntesis angulares: <expresión>, <operador>, …

b) Vocabulario terminal

- Letras minúsculas latinas de comienzo de abecedario: *a*, *b*, *c*, …
- Operadores tales como: +, -, *, /, …
- Caracteres especiales: #, @, (,), …
- Los dígitos: 0, 1, 2, …, 9
- Palabras reservadas de lenguajes de programación con minúsculas y negrita: **if**, **then**, **else**, …

c) Cadenas terminales

Las cadenas compuestas totalmente por símbolos terminales se representan como:

- Letras minúsculas latinas del final del abecedario: … w, x, y, z.

d) Cadenas

Las cadenas que contienen símbolos terminales y no terminales indiferenciado se representan por:

- Letras minúsculas griegas: α, β, γ, …, ω.

Ejemplo: Describir una grámatica a partir del siguiente conjunto de producciones:

$N \rightarrow ND \mid D$

$D \rightarrow 0 \mid 1 \mid 2$

Genere la palabra: 10021 y 222101

10021

$N \rightarrow ND$

$N \rightarrow NDD$

$N \rightarrow NDDD$

$N \rightarrow NDDDD$

$N \rightarrow DDDDD$

$N \rightarrow 1DDDD$

$N \rightarrow 10DDD$

$N \rightarrow 100DD$

$N \rightarrow 1002D$

$N \rightarrow 10021$

222101

$N \rightarrow ND$

$N \rightarrow NDD$

$N \rightarrow NDDD$

$N \rightarrow NDDDD$

$N \rightarrow NDDDDD$

$N \rightarrow DDDDDD$

$N \rightarrow 2DDDDD$

$N \rightarrow 22DDDD$

$N \rightarrow 222DDD$

$N \rightarrow 2221DD$

$N \rightarrow 22210D$

$N \rightarrow 222101$

2. RELACIONES ENTRE CADENAS

En esta sección se muestra las relaciones de derivación directa y derivación entre las cadenas de un determinado lenguaje descrito por una gramática.

2.1. Relación de derivación directa

Definición. Dada una gramática $G=(\sum_N,\sum_T,S,P)$ y dos palabras $\alpha',\beta'\in\sum^*$ decimos que α' deriva directamente a β' en un paso y se denota por $\alpha'\Rightarrow\beta'$, si y solo si, existen palabras $\delta_1,\delta_2\in\sum^*$ y una producción $\alpha\rightarrow\beta$, tales que $\alpha'=\delta_1\alpha\delta_2$, $\beta'=\delta_1\beta\delta_2$.

Ejemplo: Se tiene las siguientes producciones:

$S\rightarrow ab$

$S\rightarrow aSb$

Genere la palabra: *aabb*

Realizando generación:

$S\rightarrow ab$

$S\rightarrow aSb$

$S\rightarrow aabb$

Aplicando la derivación directa, $\alpha'\Rightarrow\beta'$ entonces $\delta_1\alpha\delta_2\Rightarrow\delta_1\beta\delta_2$, es decir, sustituyendo la primera regla en la segunda se tiene:

$S\rightarrow aabb$

2.2. Relación de derivación

Definición. Sean $\alpha_1,\alpha_m\in\sum_N^*$, se dice que estaán en relación de derivación en la gramática *G* si existen $\alpha_1,\alpha_2,\alpha_3,\cdots,\alpha_m$ tales que:

$$\alpha_1\rightarrow\alpha_2$$

$$\alpha_2\rightarrow\alpha_3$$

$$\alpha_3\rightarrow\alpha_4$$

$$\vdots$$

$$\alpha_{m-1} \rightarrow \alpha_m$$

entonces

$$\alpha_1 \rightarrow \alpha_m$$

al cual diremos que α_m deriva de α_1, o que α_1 produce α_m.

Ejemplo: Se tiene las siguientes reglas de producción, las numeramos para su posterior identificación:

(1) $S \rightarrow ASB$

(2) $A \rightarrow b$

(3) $aaA \rightarrow aaBB$

(4) $S \rightarrow d$

(5) $A \rightarrow aA$

(6) $B \rightarrow dcd$

Se tiene la siguiente derivación: $S \rightarrow abddcd$

Las derivaciones inmediatas necesarias para llegar a $abddcd$, se muestran a continuación, señalando en cada paso el número de regla aplicada.

$$S \overset{(1)}{\rightarrow} ASB \overset{(5)}{\rightarrow} aASB \overset{(2)}{\rightarrow} abSB \overset{(4)}{\rightarrow} abdB \overset{(6)}{\rightarrow} abddcd$$

Definición. Se denomina **sentencia** o frase de un lenguaje cualquiera a una cadena que sea el resultado último de una derivación a partir del símbolo inicial *S* y cuyos símbolos son terminales, es decir

$$S \rightarrow \alpha_m \qquad \text{y} \qquad \alpha_m \in \sum_T$$

Ejemplo: Utilizando la relación de derivación del ejemplo anterior, $abddcd$ es una sentencia

2.3. Definición formal de lenguaje

Definición. El lenguaje $L(G)$ generado por una gramática G es el conjunto de todas las sentencias que puede generar G. Es decir,

$$L(G) = \{\alpha_m \in \Sigma_T^* / S \to \alpha_m\}$$

Una sentencia pertenece a $L(G)$ si:

- Está compuesta por símbolos terminales
- La sentencia puede derivarse del símbolo inicial S aplicando las reglas de producción de la gramática.

Ejemplo: Considere la gramática: $G = (\Sigma_N, \Sigma_T, S, P)$, donde

$$\Sigma_N = \{S\},\ \Sigma_T = \{a, b\},\ P = \{(S, aS), (S, \lambda)\}$$

Utilizando la gramática podemos generar el siguiente lenguaje:

$$L = \{a, aa, aaa, \cdots\} = \{a^n / n \geq 0\}$$

Observe que la caracterización recursiva de las palabras de este lenguaje se deduce de su estructura gramática.

$S \to aS$

$S \to \lambda$

Si $n = 5$, la derivación de la palabra es:

$S \to aS \to aaS \to aaaS \to aaaaS \to aaaaaS \to aaaaa\lambda \to aaaaa$

Observación.

Se suelen usar la notación $(A, B) \in P$ en lugar de $A \to B$, para indicar una producción. Y, en el caso de tener más de una producción que comience en el mismo objeto, se suele usar $A \to B \mid C$, en lugar de escribir $A \to B$, $A \to C$.

Definición. Dos gramáticas G_1 y G_2 son **equivalentes** si ambas generan el mismo lenguaje:

$$L(G_1) = L(G_2)$$

Ejemplo: Sea la gramática: $G_1 = \left(\{S\}, \{0,1\}, S, P\right)$, donde $P = \left\{\left(S \to 00S11\right), \left(0S1 \to 01\right)\right\}$. Determinar el lenguaje que genera.

La forma de generar sentencias se obtiene de aplicar *n* veces la primera producción y finalizar con la aplicación de la segunda producción, asi obtenemos el lenguaje.

$$S \to 00S11 \to 0000S1111 \to \cdots \to 0^{(2n-1)}0S1^{(2n-1)}1 \to 0^{(2n)}1^{(2n)}$$

Por tanto, el lenguaje que genera la gramática es:

$$L\left(G_1\right) = \left\{0^{(2n)}1^{(2n)} \middle/ n \geq 1\right\}$$

Ejemplo: Si la segunda producción de la gramática del ejemplo anterior es $S \to 01$. Determinar el lenguaje que genera.

Tenemos:

$$S \to 00S11 \to 0000S1111 \to \cdots \to 0^{(2n)}S1^{(2n)} \to 0^{(2n+1)}1^{(2n+1)}$$

Por tanto, el lenguaje que genera la gramática es:

$$L\left(G_2\right) = \left\{0^{(2n+1)}1^{(2n+1)} \middle/ n \geq 0\right\}$$

Ejemplo: Sea: $G_3 = \left(\{S, X, Y\}, \{a, b, c\}, S, P\right)$, donde *P* tiene las reglas de producción.

$S \to abc$, $S \to aXbc$, $Xb \to bX$, $Xc \to Ybcc$,
$bY \to Yb$, $aY \to aaX$, $aY \to aa$

¿Qué lenguaje genera?

Generamos algunas instrucciones:

$S \to abc$
$S \to aXbc \to abXc \to abYbcc \to aYbbcc \to aabbcc$
$S \to aXbc \to abXc \to abYbcc \to aYbbcc \to aaXbbcc$

A partir de *aaXbbcc*, se puede llegar a $a^3b^3c^3$, del siguiente modo:

$\rightarrow aaXbbcc \rightarrow aabXbcc \rightarrow aabbXcc \rightarrow aabbYbccc \rightarrow aabYbbccc$
$\rightarrow aaYbbbccc \rightarrow aaabbbccc = a^3b^3c^3$ ó $a^3Xb^3c^3$.

Por inducción, el lenguaje que puede llegar a generar la gramática es:

$$L(G_3) = \left\{a^n b^n c^n / n \geq 1\right\}$$

Ejemplo: Dada la gramática $G = \left(\{S, A\}, \{a, b\}, S, P\right)$, $P = \left\{(S \rightarrow abAS), (abA \rightarrow baab), (S \rightarrow a), (A \rightarrow b)\right\}$. Determine el lenguaje que genera.

Se generan sentencias del lenguaje aplicando las reglas hasta que se pueda ver la forma general del lenguaje.

$S \rightarrow a$
$S \rightarrow abAS \rightarrow abbS \rightarrow abba$
$S \rightarrow abAS \rightarrow baabS \rightarrow baabc$
$S \rightarrow abAS \rightarrow abAabAS \rightarrow baabbaabS \rightarrow baabbaaba$
$S \rightarrow abAS \rightarrow abAabAS \rightarrow \cdots \rightarrow (abA)^n S \rightarrow (baab)^n a$
$S \rightarrow abAS \rightarrow abAabAS \rightarrow abbabbS \rightarrow abbabba$
$S \rightarrow abAS \rightarrow abAabAS \rightarrow \cdots \rightarrow (abA)^n S \rightarrow (abb)^n a$
$S \rightarrow abAS \rightarrow abAabAS \rightarrow abAabAabAS \rightarrow baababbbaaba$

$L(G)$ = {cadenas que contienen *abb* y *baab* intercambiándose y reproduciéndose cualquier número de veces, y terminando siempre con el símbolo *a*}

Ejemplo: Crear una gramática que genere los siguientes lenguajes:

a) $\{a, aa, aaa\}$

b) $\{a, aa, aaa, aaaa, aaaaa, \cdots\}$

c) $\{\lambda, a, aa, aaa\}$

d) $\{\lambda, a, aa, aaa, aaaa, aaaaa, \cdots\}$

Solución:

a) Se observa que la gramática sólo genera tres palabras y todas ellas están formadas por el símbolo "*a*". Por tanto, una posible solución es:

$$G = \left(\{S\}, \{a\}, S, P\right)$$

Con producciones:

$$S \to a \mid aa \mid aaa$$

b) En este caso, las palabras del lenguaje están formadas por una o varias ases. Por tanto, una posible solución es:

$$G = \left(\{S\}, \{a\}, S, P\right)$$

Con regla de producción:

$$S \to a \mid aS$$

c) Este caso es similar al primer caso, pero, a diferencia de éste, la palabra vacía debe pertenecer al lenguaje. Por tanto, una posible solución es:

$$G = \left(\{S\}, \{a\}, S, P\right)$$

Con regla de producción:

$$S \to \lambda \mid a \mid aa \mid aaa$$

d) Finalmente, este caso es similar al segundo caso, pero, a diferencia de éste, la palabra vacía debe pertenecer al lenguaje. Por tanto, una posible solución es:

$$G = \left(\{S\}, \{a\}, S, P\right)$$

Con regla de producción:

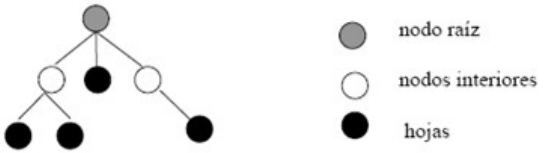
$$S \to \lambda \mid aS$$

3. ARBOL DE DERIVACIÓN

Un árbol de derivación permite mostrar gráficamente cómo se puede derivar cualquier cadena de un lenguaje a partir del símbolo de inicio de launa gramática que genera ese lenguaje.

El árbol de derivación tiene las siguientes características:

- el nodo raíz está rotulado con el símbolo distinguido de la gramática;
- cada hoja corresponde a un símbolo terminal o un símbolo no terminal;
- cada nodo interior corresponde a un símbolo no terminal.



Para cada cadena del lenguaje generado por una gramática es posible construir (al menos) un árbol de derivación, en el cual cada hoja tiene como rótulo uno de los símbolos de la cadena.

Construcción

- Cada nodo del árbol va a contener un símbolo.
- En el nodo raíz se pone el símbolo inicial *S*.
- Se efectúa una ramificación del árbol por cada producción que se aplique: Si a la variable de un nodo, *A*, se le aplica una determinada regla $A \to \alpha$, entonces para cada símbolo que aparezca en α se añade un hijo con el símbolo correspondiente, situados en el orden de izquierda a derecha.
- Este proceso se repite para todo paso de la derivación.
- Si la parte derecha es una cadena vacía, entonces se añade un solo hijo, etiquetado con λ.
- En cada momento, leyendo los nodos de izquierda a derecha se lee la palabra generada.

Ejemplo: Se tiene el siguiente conjunto de producciones:

$S \to aAS$

$S \to a$

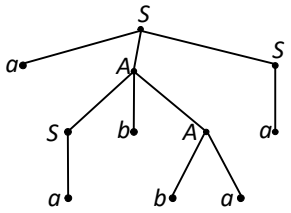
$A \to SbA \mid SS \mid ba$

Encuentre la palabra: *aabbaa* y su respectivo árbol.

Se tiene:

$$S \rightarrow aAS \rightarrow aAa \rightarrow aSbAa \rightarrow aabAa \rightarrow aabbaa$$

El árbol de derivación es el siguiente:



Un árbol de derivación puede proceder de dos cadenas de derivación distintas:

i) Se llama derivación **por la izquierda** asociada a un árbol a aquella en la que siempre se deriva primero la primera variable (más a la izquierda) que aparece en la palabra.
ii) Se llama derivación **por la derecha** asociada a un árbol a aquella en la que siempre se deriva primero la última variable (más a la derecha) que aparece en la palabra.

Ejemplo: Retomando el ejemplo anterior. Del árbol se tiene la plabra *aabbaa*

Derivación por la izquierda:

$$S \rightarrow aAS \rightarrow aSbAS \rightarrow aabAS \rightarrow aabbaS \rightarrow aabbaa$$

Derivación por la derecha:

$$S \rightarrow aAS \rightarrow aAa \rightarrow aSbAa \rightarrow aSbbaa \rightarrow aabbaa$$

Ejemplo: Sea el conjunto de producciones:

$S \rightarrow ab$

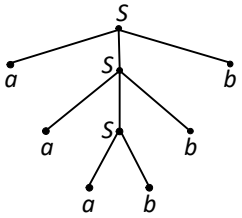
$S \rightarrow aSb$

Genere la palabra: *aaabbb* y su respectivo árbol.

Se tiene:

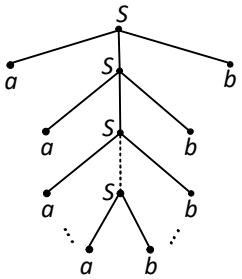
$$S \to aSb \to aaSbb \to aaabbb$$

El árbol de derivación es el siguiente:



Para n derivaciones tenemos:

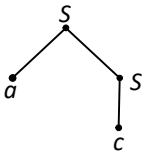
$$S \to aSb \to aaSbb \to aaaSbbb \to \cdots \to aaa...ab...bbb$$



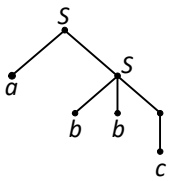
y el lenguaje generado sería: $L(G) = \{a^n b^n / n \geq 1\}$

Ejemplo: Dada la gramática $G_3 = (\{S, A\}, \{a, b, c\}, S, P)$, $P = \{(S \to aA), (A \to bbA), (A \to c)\}$. Determine el lenguaje que genera a través de los árboles de derivación:

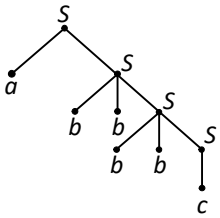
- $S \to aA \to ac$



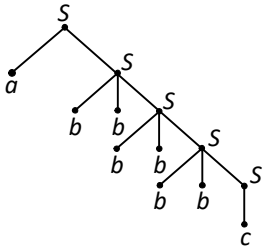
- $S \to aA \to abbA \to abbc$



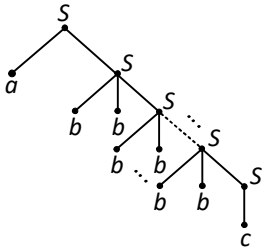
- $S \to aA \to abbA \to abbbbA \to abbbbc$



- $S \to aA \to abbA \to abbbbA \to abbbbbbA \to abbbbbbc$



- $S \to aA \to abbA \to abbbbA \to abbbbbbA \to \cdots \to ab^{(2n)}c$



El lenguaje generado es: $L(G) = \left\{ ab^{(2n)}c / n \geq 1 \right\}$ y se puede definir con la siguiente expresión regular, cuya definición se estudiará más adelante.

$$L(G) = a(bb)^* c$$

Definición. Una gramática se dice **ambigua** si existe una palabra con dos árboles de derivación distintos; de lo contrario decimos que la gramática es inambigua.

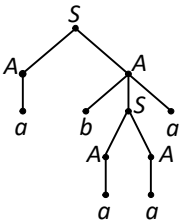
Ejemplo: la gramática generada por las siguientes:

$S \to AA$

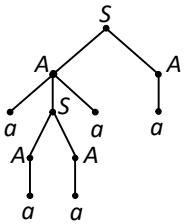
$A \to aSa$

$A \to a$

es ambigua ya que la palabra *aaaaa* = a^5 tiene dos árboles siguientes:



aaaaa



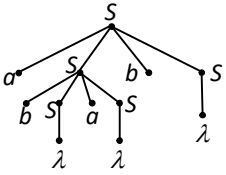
aaaaa

Ejemplo: Se tiene la siguiente regla de producciónes:

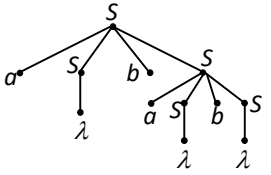
$S \to aSbS \mid bSaS \mid \lambda$

¿La gramática será ambigua? Encuentre la la palabra *abab*

Se tiene los siguientes árboles:



abab



abab

Observación.

No siempre es posible eliminar la ambigüedad. No es porque no conozcamos un algoritmo que elimine la ambigüedad, sino que estamos seguros de que no puede hacerse.

Ejemplo: Determine el lenguaje asociado a las siguientes gramáticas:

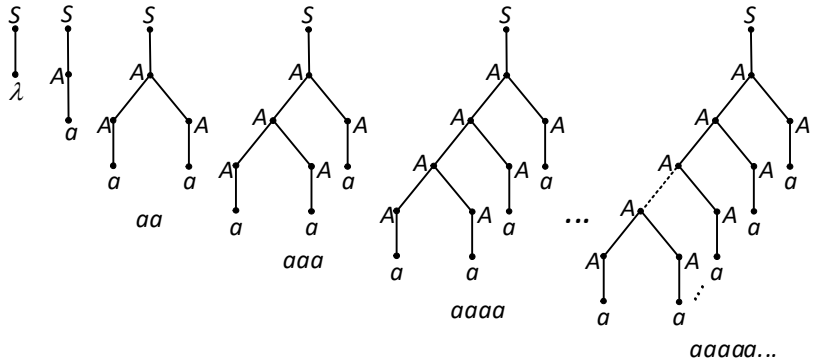
a) $G_1 = \left(\{S, A\}, \{a\}, S, P\right)$ con:

P_1:

$S \to \lambda \mid A$

$A \to AA \mid a$

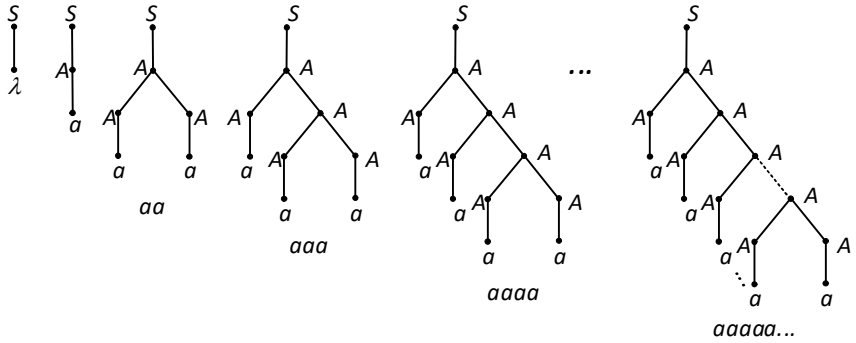
Una forma de encontrar las cadenas (o palabras) del lenguaje asociado a una gramática es haciendo uso de una estructura en forma de árbol en la que la raíz es el axioma y las hojas las palabras del lenguaje.



El lenguaje generado es:

$$L(G_1) = \{\lambda, a, aa, aaa, aaaa, aaa...\} = \{\lambda, a^n / n \geq 1\}$$

Así mismo, las cadenas se pueden encontrar por más de un camino, por tanto, la gramática es ***ambigua***.



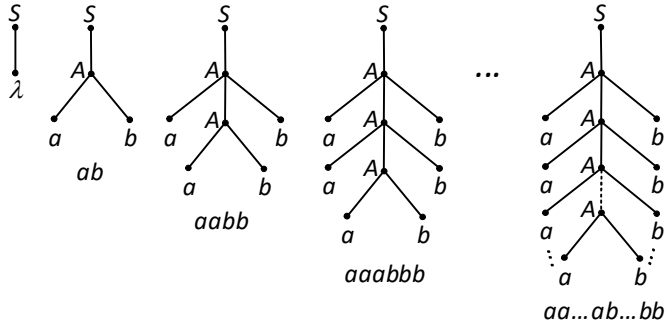
b) $G_2 = \left(\{S, A\}, \{a, b\}, S, P\right)$ con:

P_2:

$S \to \lambda \mid A$

$A \to aAb \mid ab$

Se tiene:



El lenguaje generado es:

$$L\left(G_2\right) = \left\{\lambda, ab, aabb, aaabbb, ..., aa...ab...bb\right\} = \left\{\lambda, a^n b^n \,/\, n \geq 1\right\}$$

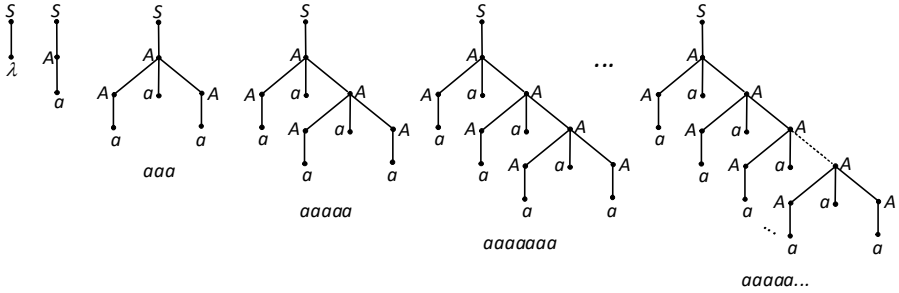
c) $G_3 = \left(\{S, A\}, \{a\}, S, P\right)$

P_3:

$S \to \lambda \mid A$

$A \to AaA \mid a$

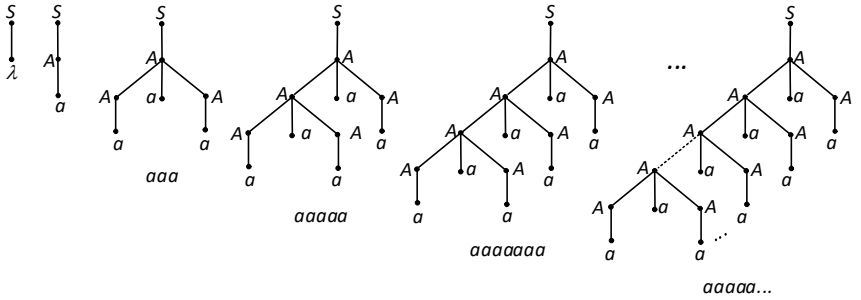
Se tiene:



El lenguaje generado es:

$$L(G_3) = \{\lambda, a, aaa, aaaaa, aaaaaaa, ...\} = \left\{\lambda, a^{(2n+1)} \middle/ n \geq 0\right\}$$

Las cadenas se pueden encontrar por más de un camino, por tanto, la gramática es ***ambigua***.



d) $G_4 = \left(\{S, A, T\}, \{a, b\}, S, P\right)$

P_4:

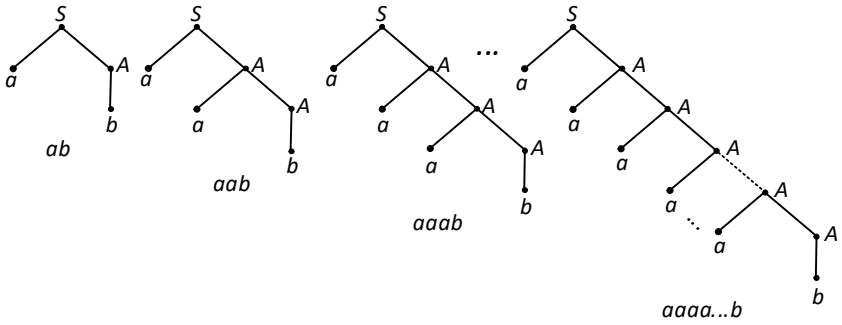
$S \to aA$

$A \to b \mid aA \mid Tb$

$T \to Tb \mid b$

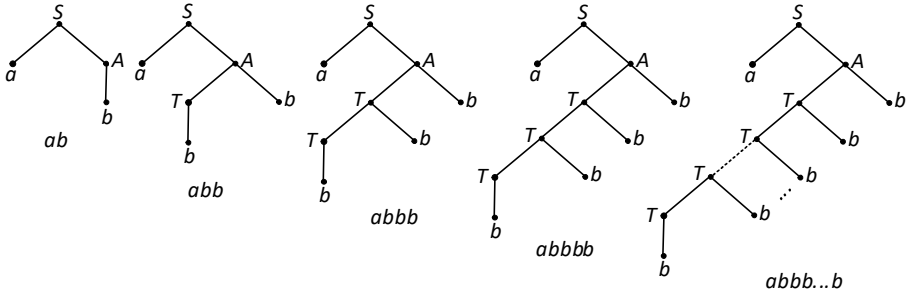
Se tiene:

- Un primer lenguaje:



entonces: $L\left(G_{4_1}\right)=\left\{a^n b / n \geq 1\right\}$

- Un segundo lenguaje:



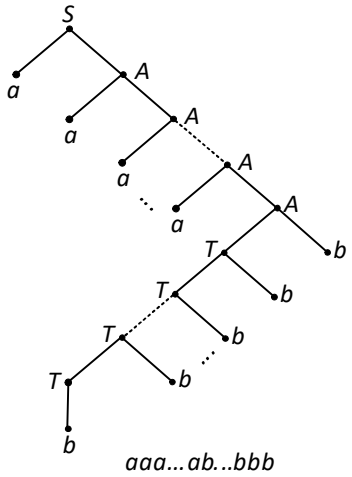
entonces: $L\left(G_{4_2}\right)=\left\{ab^m / m \geq 1\right\}$

- finalmente, el lenguaje generado sería: $L\left(G_4\right)=L\left(G_{4_1}\right) \cup L\left(G_{4_2}\right)$

entonces: $L\left(G_4\right)=\left\{a^n b / n \geq 1\right\} \cup\left\{ab^m / m \geq 1\right\}=\left\{a^n b^m / n, m \geq 1\right\}$

$L\left(G_4\right)=\{ab, aab, aaab, ..., aa...ab, aabb, aabbb, aa...ab...bb, abb, abbb, ab...b\}$

y cuyo árbol es el siguiente:



e) $G_5 = \left(\{S, A\}, \{a, b\}, S, P\right)$

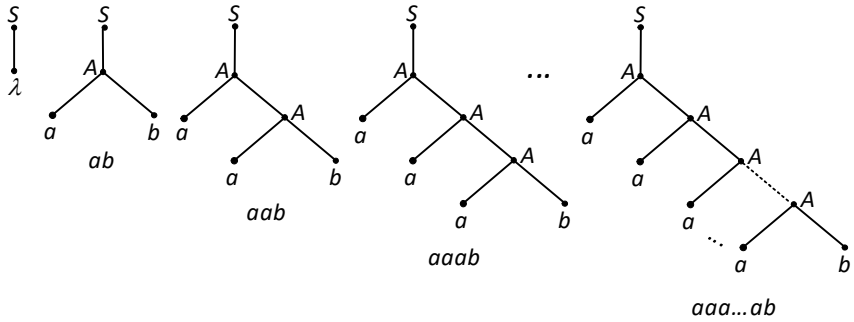
P_5:

$S \to \lambda \mid A$

$A \to Ab \mid aA \mid ab$

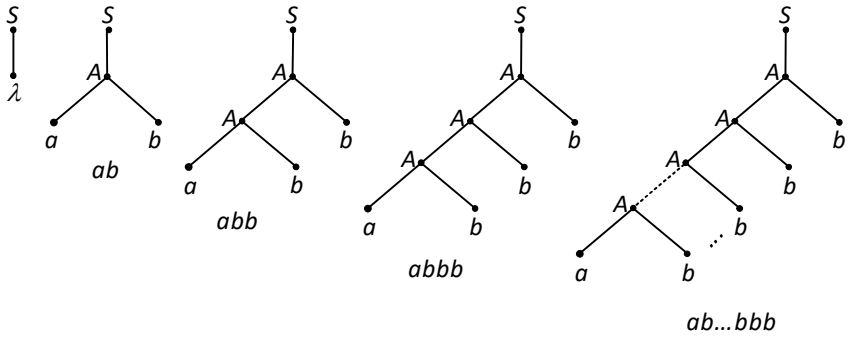
Se tiene:

- Un primer lenguaje:



entonces: $L\left(G_{5_1}\right) = \left\{a^n b / n \geq 1\right\}$

- Un segundo lenguaje:



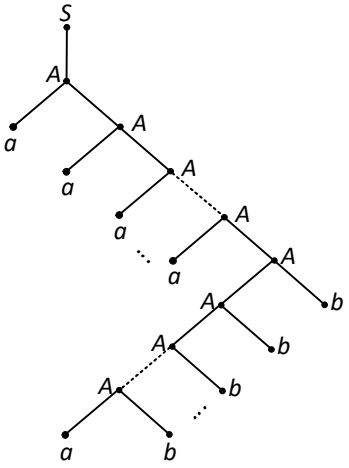
entonces: $L\left(G_{5_2}\right)=\left\{ab^m / m\geq 1\right\}$

- En general, el lenguaje sería: $L\left(G_5\right)=L\left(G_{5_1}\right)\cup L\left(G_{5_2}\right)$

entonces: $L\left(G_4\right)=\left\{a^n b / n\geq 1\right\}\cup\left\{ab^m / m\geq 1\right\}=\left\{a^n b^m / n,m\geq 1\right\}$

$L\left(G_4\right)=\left\{\lambda, ab, aab, aaab, ..., aabb, aabbb, aa...ab...bb, abb, abbb, ab...b\right\}$

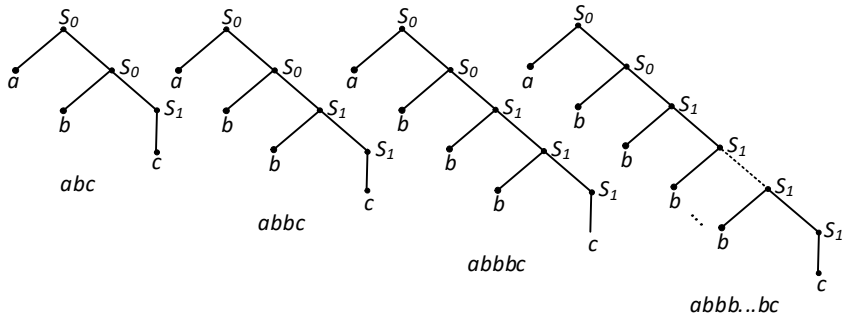
y cuyo árbol es:



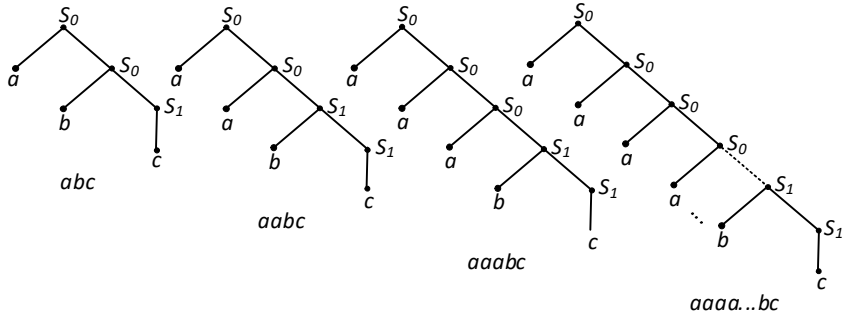
La gramática es ***ambigua***, queda para el estudiante verificarlo.

Ejemplo: Dada la gramática $G=\left(\{S_0,S_1,a,b,c\},\{a,b,c\},\{S_0\},P\right)$, $P=\left\{\left(S_0\to aS_0\right),\left(S_0\to bS_1\right),\left(S_1\to bS_1\right),\left(S_1\to c\right)\right\}$. Determine el lenguaje que genera a través del árbol de derivación:

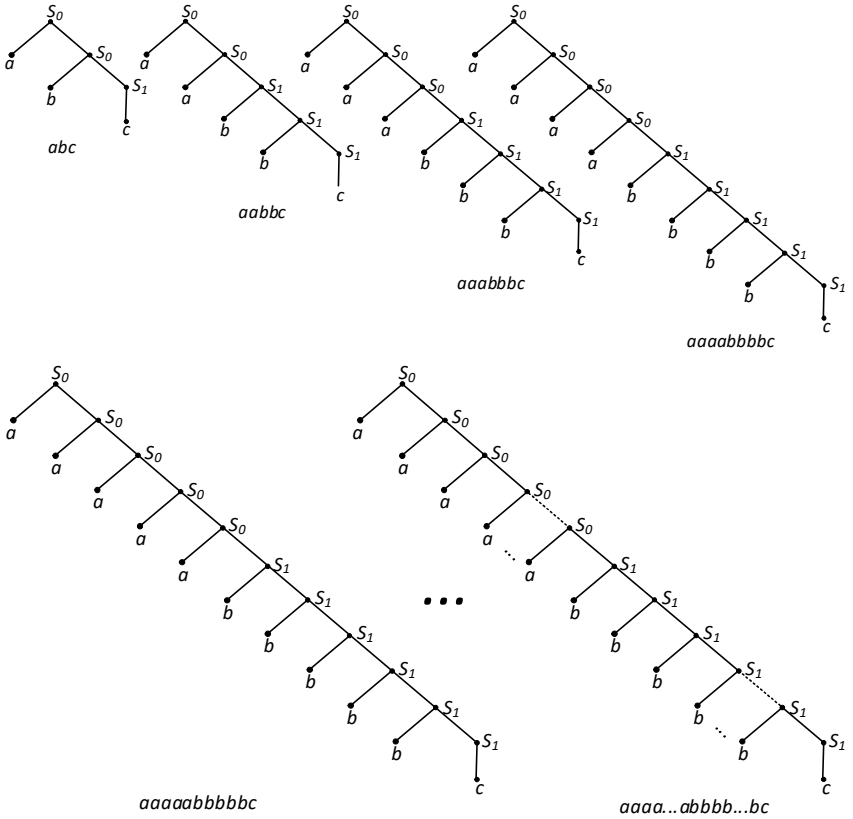
Se tiene:



Un primer lenguaje generado es: $L\left(G_1\right)=\left\{ab^nc/n\geq 1\right\}$



el segundo lenguaje generado es: $L\left(G_2\right)=\left\{a^nbc/n\geq 1\right\}$



y un tercer lenguaje generado es: $L(G_3) = \{a^n b^n c / n \geq 1\}$

Por lo tanto,

$$L(G) = L(G_1) \cup L(G_2) \cup L(G_3)$$
$$L(G) = \{ab^n c / n \geq 1\} \cup \{a^n bc / n \geq 1\} \cup \{a^n b^n c / n \geq 1\}$$

Además, se puede notar que la terminal *a* pude incrementar independientemente del crecimiento de la terminal *b* y viceversa, entonces, el lenguaje generado sería:

$$L(G) = \{a^n b^m c / n \geq 1, m \geq 1\}$$

4. JERAQUÍAS DE LAS GRAMÁTICAS

En función de la forma de sus producciones, se puede caracterizar qué tan compleja es una gramática formal. Noam Chomsky mostró que esta caracterización clasifica jerárquicamente a las gramáticas formales: Gramáticas en un nivel están incluidas en los siguientes niveles y la inclusión entre niveles es propia.

Tabla 2.1: Jerarquía gramatical de Chomsky.

Gramática	***Lenguaje***	***Regla de Producción***	***Solución***
Tipo 0	Recursivas	$\alpha \rightarrow \beta$ Sin restricciones	Máquinas de Turing
Tipo 1	Dependiente de contexto	$\alpha A \beta \rightarrow \alpha \gamma \beta$	Autómatas lineales acotados
Tipo 2	Independiente de contexto	$A \rightarrow \gamma$	Autómatas de pila
Tipo 3	Regular	$A \rightarrow aB$ $A \rightarrow a$	Autómatas finitos, regulares

Los cuatro tipos de gramáticas que se estudiarán a continuación (tipo 0, tipo 1, tipo 2, y tipo 3), cada una de ellas tiene restricciones más fuertes que las anteriores. Las gramáticas de tipo 0, contienen a todas las demás. Las de tipo 1 contienen a las de tipo 2 y tipo 3. Y por último las de tipo 2 contienen a las de tipo 3. Es decir, una gramática de tipo 3 es de tipo 2, tipo 1 y tipo 0. Por lo tanto, se define una jerarquía de gramáticas respecto de la relación de inclusión, que se puede representar gráficamente mediante el diagrama de la figura 2.1.

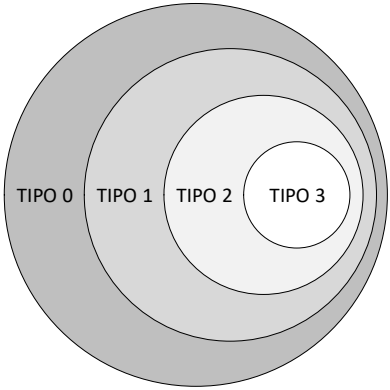


Figura 2.1: Relación de inclusión entre los distintos tipos de gramáticas.

4.1. Gramática TIPO 0

También llamadas ***gramáticas no restringidas*** o ***gramáticas con estructura de frase***. Las reglas de derivación son de la forma:

$$\alpha \to \beta$$

donde $\alpha = xAy$, con $x, y, \beta \in \Sigma^*$ y $A \in \Sigma_N$, es decir la única restricción es que no puede haber reglas de la forma $\lambda \to \beta$ donde λ es la cadena vacia.

Ejemplo: Todas las gramáticas mostradas en las secciones anteriores de este capítulo son de Tipo 0, pues en ninguna de ellas existe la producción $\lambda \to \beta$ siendo la cadena vacia.

4.2. Gramática TIPO 1

Tambien llamadas ***gramáticas sensibles al contexto*** (*context sensitive*). En ellas las reglas de producción son de la forma:

$$\alpha A \beta \to \alpha \gamma \beta$$

siendo $A \in \Sigma_N$, $\gamma \in \Sigma^+$, $\alpha, \beta \in \Sigma^*$

El nombre de gramáticas dependientes del contexto se debe a que las producciones se pueden interpretar como que "A se convierte en γ, siempre que se encuentre en un determinado contexto, es decir, precedida por α seguidapor β". Por lo tanto, es necesario conocer el contexto en el que se encuentra A para poder aplicar la producción.

Ejemplo: La gramática $G = \left(\{S, A, B\}, \{a, b\}, S, P\right)$, cuyas producciones P se muestran a continuación:

$S \to aA$ $\qquad$ $A \to bAA$

$S \to bA$ $\qquad$ $B \to b$

$A \to a$ $\qquad$ $B \to bS$

$A \to aS$ $\qquad$ $B \to aBB$

Ejemplo: La gramática $G = \left(\sum_T, \sum_N, S, P\right)$, donde: $\sum_N = \{\langle N\rangle, \langle D\rangle\}$, $\sum_T = \{0,1,2,3,4,5,6,7,8,9\}$, $S = \langle N\rangle$ y las reglas de producción son:

$\langle N\rangle \to \langle D\rangle\langle N\rangle$

$\langle N\rangle \to \langle D\rangle$

$\langle D\rangle \to 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$

A continuación, se muestran algunos ejemplos de gramáticas que no son de tipo 1, y que pueden ilustrar mejor la definición de estas gramáticas.

Ejemplo: La gramática $G = \left(\{S\}, \{a, b\}, S, P\right)$, donde las producciones P son:

$S \to aaaaSbbbb$

$aSb \to ab$

La producción $aSb \to ab$ no es del tipo 1, pues se sustituye S por vacío en el contexto *a...b*.

Sin embargo, si se esta producción fuera $S \to ab$ o $aSb \to abb$, entonces sería de tipo 1.

Propiedades de *No decrecimiento.* Las cadenas que se obtienen en cualquier derivación de una gramática de tipo 1 son de longitud no decreciente, es decir:

$$\alpha \to \beta \text{ , se tiene que } |\alpha| \leq |\beta|$$

Es importante mencionar, que α no puede ser de cadena vacía. Por esta razón, los lenguajes representados por las gramáticas del tipo 1, se llaman lenguajes dependientes del contexto o lenguajes sensibles al contexto.

4.3. Gramática TIPO 2

Las gramáticas de tipo 2 también se denominan gramáticas de ***contexto libre*** o ***libres de contexto*** (*context free*). Sus reglas de producción tan sólo admiten tener un símbolo no terminal en su parte izquierda, es decir son de la forma:

$$A \to \gamma$$

con $A \in \Sigma_N$ y $\gamma \in \Sigma^*$

En este tipo de gramáticas, la conversión de A en γ se realiza independientemente del contexto en el que se encuentre A, de ahí su nombre. Son especialmente adecuadas para representar los aspectos sintácticos de cualquier lenguaje de programación.

Ejemplo: La gramática $G = (\{S, A, B\}, \{a, b\}, S, P)$, cuyas producciones se muestran a continuación:

$S \to aB$	$A \to bAA$
$S \to bA$	$B \to b$
$A \to a$	$B \to bS$
$A \to aS$	$B \to aBB$

es de tipo 2

Ejemplo: La gramática $G = \left(\{S, A\}, \{a, b\}, S, P\right)$, cuyas producciones se muestran a continuación:

$$S \to aS$$
$$S \to aA$$
$$A \to bA$$
$$A \to b$$

es de tipo 2

4.4. Gramática TIPO 3

Las gramáticas de tipo 3 tambien denominadas ***regulares*** o gramáticas lineales a la derecha, comienzan sus reglas de producción por un símbolo terminal, que puede ser seguido o no por un símbolo no terminal, Hay dos tipos de gramáticas regulares y sus producciones pueden ser de la siguiente forma:

1) **Lineales por la derecha (GLD)**, con regla de producción de la forma:

$$A \to aB$$
$$A \to a$$

con $A, B \in \sum_N$ y $a \in \sum_T$

2) **Lineales por la izquierda (GLI),** con regla de producción de la forma:

$$A \to Ba$$
$$A \to a$$

con $A, B \in \sum_N$ y $a \in \sum_T$

Ejemplo: La gramática $G = \left(\{S, A\}, \{a, b\}, S, P\right)$, de ejemplo anterior es de tipo 3.

4.5. Lenguajes con cadena vacía

Según las definiciones anteriores la cadena vacia no puede aparecer en ningún lenguajede tipo 1, 2 o 3. Supongamos que deseamos añadir la cadena vacia a un lenguaje.

Se pretende crear un nuevo lenguaje *L'*, a partir del lenguaje *L* de tal forma que:

$$L = L' \cup \{\lambda\}$$

Bastará añadir de algún modo a la descripción del lenguaje *L*.

Una forma de hacer esto es añadir la siguiente regla de producción $S \to \lambda$ a las reglas de la gramática que describe *L*.

Propiedades:

- Si *L* es un lenguaje de tipo 1, 2 o 3 entonces $L \cup \{\lambda\}$ y $L - \{\lambda\}$ son lenguajes de tipo 1, 2, o 3 respectivamente.
- Dada una gramática *G* cualquiera de tipo 1, 2 o 3 se puede obtener otra *G'*, con $S' \to \lambda$ de forma que $L(G') = L(G) \cup \{\lambda\}$.

Ejemplo: Determinar el tipo de las siguientes gramáticas en la jerarquía de Chomsky, justifique su respuesta:

a) $G = (\{S, A, B\}, \{a, b\}, S, P)$

Con $P = \{S \to aA, A \to bB, A \to aA, A \to a, B \to \lambda\}$

Es de Tipo 0. Podría ser de tipo 3 pero tiene una regla compresora ($B \to \lambda$)

b) $G = (\{S, A, B, C\}, \{a, b, c\}, S, P)$

Con

$$P = \{S \to aAb, \mathrm{S} \to Ba, S \to \lambda, aAbC \to aAbB, aAbC \to aabC, BCc \to AaCc, BCc \to BaAbc, C \to Ca, C \to a\}$$

Es de Tipo 1. En las producciones 4, 5, 6 y 7 se mantiene el contexto; y el resto serían válidas en una gramática de tipo 2.

c) $G = (\{\mathrm{S}, \mathrm{CASERON}, \mathrm{BOSQUE}, \mathrm{TIGRE}\}, \{\mathrm{casa}, \mathrm{jardin}, \mathrm{gato}\}, S, P)$

Con

P = { S→TIGRE jardin, S→BOSQUE CASERON, BOSQUE→λ,
jardin CASERON TIGRE casa→jardin BOSQUE TIGRE casa,
gato CASERON BOSQUE→ BOSQUE casa TIGRE BOSQUE,
BOSQUE→TIGRE casa, BOSQUE→jardin}

Es de Tipo 0. Por regla compresora (BOSQUE→λ). El resto de reglas mantienen el contexto, siendo válidas en una gramática de tipo 1.

d) $G = \left(\{S, A, B, C\}, \{x, y\}, S, P\right)$

Con

$$P = \{S \to Cx, S \to Cy, S \to By, S \to Ax, S \to x, S \to y, A \to Ax, A \to Cx, A \to x$$
$$B \to By, B \to yA, C \to xA\}$$

Es de Tipo 2. Porque hay reglas de producción tipo $C \to xA$ y tipo $C \to xA$ mezcladas en la misma gramática.

e) $G = \left(\{S, B\}, \{a, b, c\}, S, P\right)$

Con $P = \{S \to abc, S \to aBSc, Ba \to aB, Bb \to bb\}$

Es de Tipo 0. Porque la regla 3 ($Ba \to aB$) no mantiene el contexto.

5. LENGUAJES Y EXPRESIONES REGULARES

En esta sección se van a estudiar las gramáticas regulares, también llamadas de tipo 3 atendiendo a la clasificación de Chomsky. También se aborda el estudio de las expresiones regulares que permiten definir de forma precisa los lenguajes generados por estas gramáticas (lenguajes regulares).

5.1. Lenguajes regular

De Castro (2004) nos indica que, los lenguajes regulares sobre un alfabeto dado $\sum$ son todos los lenguajes que se pueden formar a partir de los lenguajes básicos $\varnothing, \{\lambda\}, \{a\}, a \in \sum$, por medio de las operaciones de unión, concatenación y estrella de Kleene.

Definición. Sea $\sum$ un alabeto, se define:

i) $\varnothing, \{\lambda\}$ y $\{a\}$ para cada $a \in \sum$, son lenguajes regulares (LR) sobre $\sum$. Estos son los denominados lenguajes regulares básicos.

ii) Si A y B son lenguajes regulares sobre $\sum$, también lo son $A \cup B$, $A.B$ y A^*.

Observación.

Tanto $\sum$ como $\sum^*$ son lenguajes regulares sobre $\sum$. La unión, la concatenación y la estrella de Kleene se denominan operaciones regulares.

Ejemplo: Dado $\sum = \{a, b\}$, las siguientes afirmaciones son ciertas:

- $\varnothing$ y $\{\lambda\}$ son lenguajes regulares
- $\{a\}$ y $\{b\}$ son lenguajes regulares
- $\{a, b\}$ son regulares porque es la unión de $\{a\}$ y $\{b\}$
- $\{a, ab, b\}$ es un lenguaje regular
- $\{a^n / n \geq 0\}$ es un lenguaje regular
- $\{a^n b^m / n \geq 0, m \geq 0\}$ es un lenguaje regular
- $\{(ab)^n / n \geq 1\}$ es un lenguaje regular

Ejemplo: Sea $\sum = \{a, b\}$, los siguientes son lenguajes regulares sobre $\sum$:

- $A = \{b\}^* \{a\} \{b\}^*$ todas las cadenas que tienen exactamente una *a*.
- $B = \{b\} \{a, b\}^*$ todas las cadenas que comienzan en *b*.
- $C = \{a, b\}^* \{ba\} \{a, b\}^*$ todas las cadenas que continen *ba*.
- $(\{a\} \cup \{b\}^*) \{a\}$
- $\left[(\{a\}^* \cup \{b\}^*) \{b\} \right]^*$

Observación.

Construir y especificar lenguajes regulares, suele ser muy engorroso, por lo que, existe una manera de representarlos a través de las expresiones regulares.

5.2. Expresión regular

Con el propósito de simplificar la descripción de los lenguajes regulares se definen las llamadas expresiones regulares (ER).

Definición. Una expresión regular sobre un alfabeto $\sum$, se define recursivamente como:

i) Expresiones regulares básicas

- $\varnothing$ es una expresión regular que representa al lenguaje vacio.
- λ es una expresión regular que representa al lenguaje vacio $\{\lambda\}$
- a es una expresión regular que representa al lenguaje $\{a\}$, $\forall a \in \sum$

ii) Si α y β son expresiones regulares sobre $\sum$, también lo son:

- $\alpha + \beta$ se denota por $A \cup B$
- $\alpha.\beta$ se denota por $A.B$
- α^* se denota por A^*

Los paréntesis "(" y ")" son símbolos de agrupación y se pueden omitir si no hay peligro de ambigüedad.

Ejemplo: Dado $\sum = \{a,b\}$, las siguientes son expresiones regulares:

$\{a,b\} = \{a\} \cup \{b\} = a + b$

$\{ab\} = ab$

$\{b\}^* = b^*$

$\{a\}^+ = a^+$

Ejemplo: Dado el alfabeto $\sum = \{a,b,c\}$, la expresión regular:

$$\left(a \cup b^*\right) a^* \left(bc\right)^*$$

representa al lenguaje

$$\left(\{a\} \cup \{b\}^*\right).\{a\}^* .\{bc\}^*$$

Ejemplo: Dado el alfabeto $\sum = \{a,b\}$, son ejemplos de lenguajes regulares con su respectiva expresión regular.

Lenguaje Regular (LR)	Expresión Regular (ER)
$\{\lambda\}$	λ
$\{a\}$	a
$\{abb\} = \{a\}\{b\}\{b\}$	Abb
$\{a,b\} = \{a\} \cup \{b\}$	$a + b$
$\{ab,b\} = \{ab\} \cup \{b\}$	$ab + b$
$\{a,\lambda\}\{bbb\}$	$(a + \lambda)\ bbb$
$\{a\}^* \{ab\}$	a^*ab
$\left(\{\lambda\} \cup \{a\}\right)^* .\left(\{a\} \cup \{b\}\right)^* .\{ba\}^*$	$(\lambda + a)^*(a + b)^*\ (ba)^*$

Definición. Dos expresiones regulares α y β son iguales o equivalentes, si designan al mismo lenguaje regular.

Si $L(\alpha) = L(\beta)$, entonces $\alpha = \beta$

Propiedades:

A partir de la definición anterior se pueden enunciar las siguientes propiedades:

- Conmutativa respecto a la unión: $\alpha + \beta = \beta + \alpha$
- Asociativa: $\alpha(\beta\gamma) = (\alpha\beta)\gamma$

  $\alpha + (\beta + \gamma) = (\alpha + \beta) + \gamma$
- Distributiva: $\alpha(\beta + \gamma) = (\alpha\beta) + (\alpha\gamma)$

- $\varnothing$ es elemento neutro de la unión: $\alpha + \varnothing = \varnothing + \alpha = \alpha$
- λ es elemento neutro de la concatenación: $\alpha.\lambda = \lambda.\alpha = \alpha$
- Operación del cierre de Kleene.
  - $\lambda^* = \lambda$
  - $\varnothing^* = \lambda$
  - $a^*.a^* = a^*$
  - $a.a^* = a^*.a$
  - $\left(a^*\right)^* = a^*$
  - $a^* = \lambda + a.a^*$
  - $\left(\alpha + \beta\right)^* = \left(\alpha^* + \beta^*\right)^* = \left(\alpha^*\beta^*\right)^*$
  - $\left(\alpha + \lambda\right)^* = \left(\alpha^* + \lambda\right) = \alpha^*$

Ejemplo: Sea el alfabeto $\sum = \{a, b\}$ y la expresión regular aa^*bb^*. Indicar el lenguaje que denota, y algunas cadenas de dicho lenguaje.

Algunas cadenas son:
Ab, aab, aaaaab, abbbb, abb, aaaab

El lenguaje que se describe es *L* = {cadenas que comienzan por una *a* y continuan con varias o ninguna *a*, y siguen con una *b* y continuan con varias o ninguna *b*}

Ejemplo: Sea el alfabeto $\sum = \{1, 2, 3\}$, la expresión regular $\left(1 \cup 2\right)^* 3$ indica el conjunto de todas las cadenas formadas con los símbolos 1 y 2, sucediéndose cualquier número de veces (y en cualquier orden), y siempre terminando la cadena en el símbolo 3. Ejemplos de sentencias:

3	23
13	223
123	113
11113	121211223
221113	111212213

6. REPRESENTACIÓN DE LENGUAJES Y GRAMÁTICAS ESPECIALES

6.1. Notación Backus-Naur Form (BNF)

La notación de Backus-Naur Form, es una metasintaxis usada para expresar gramáticas libres de contexto: es decir, una manera formal de describir lenguajes formales. El BNF se utiliza extensivamente como notación para las gramáticas de los lenguajes de programación de la computadora, los sistemas de comando y los protocolos de comunicación, así como una notación para representar partes de las gramáticas de la lengua natural.

En la gramática BNF la flecha "$\rightarrow$" de una composición se indica con "::=". La tabla 2.2 muestra algunos símbolos utilizados.

Tabla 2.2: Símbolos utilizados por la notación BNF.

Símbolo	*Descripción*
$\langle \quad \rangle$	Símbolos no terminales. Declaraciones u objetos declarados
::=	Operador "se defne como"
|	Operador disyuntivo "ó"
$\alpha \ldots \omega$	Rangos de caracteres

Ejemplo: Se tiene el siguiente conjunto de producciones:

$$S \rightarrow aAS \qquad A \rightarrow SS$$
$$S \rightarrow a \qquad A \rightarrow ba$$
$$A \rightarrow SbA$$

Expresar en la notación BNF.

La representación es:

$$\langle S \rangle ::= a \langle A \rangle \langle S \rangle \mid a$$
$$\langle A \rangle ::= \langle S \rangle b \langle A \rangle \mid \langle S \rangle \langle S \rangle \mid ba$$

Ejemplo: Sea la gramática $G = \left(\{S, A, B, C\}, \{x, y\}, S, P\right)$ con reglas de producción:

$$S \to xB \qquad B \to yxC \qquad C \to BxA$$
$$A \to xBy \qquad B \to yyC \qquad C \to xy$$
$$A \to CBx \qquad B \to x \qquad A \to y$$

La representación de la gramática por medio de BNF es:

$$\langle S \rangle ::= x\langle B \rangle$$
$$\langle A \rangle ::= x\langle B \rangle y \mid \langle C \rangle \langle B \rangle x \mid y$$
$$\langle B \rangle ::= yx\langle C \rangle \mid yy\langle C \rangle \mid x$$
$$\langle C \rangle ::= \langle B \rangle x\langle A \rangle \mid xy$$

Ejemplo: La gramática libre de contexto *G* en representación BNF para leer un número real en Pascal es:

$$\sum_T = \{., -, +, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$$

$$\sum_N = \{\langle real \rangle, \langle entero \rangle, \langle fracción \rangle, \langle ent\_s\_signo \rangle, \langle ent\_c\_signo \rangle \langle dígito \rangle\}$$

Producciones:

$$\langle real \rangle ::= \langle entero \rangle \mid \langle fracción \rangle \mid \langle entero \rangle \langle fracción \rangle$$
$$\langle entero \rangle ::= \langle ent\_s\_signo \rangle \mid \langle ent\_c\_signo \rangle$$
$$\langle ent\_c\_signo \rangle ::= -\langle ent\_s\_signo \rangle \mid +\langle ent\_s\_signo \rangle$$
$$\langle ent\_s\_signo \rangle ::= \langle dígito \rangle \mid \langle dígito \rangle \langle ent\_s\_signo \rangle$$
$$\langle fracción \rangle ::= \bullet \langle ent\_s\_signo \rangle$$
$$\langle dígito \rangle ::= 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$$

Por lo tanto, la derivación para determinar si la palabra -153.87 pertenece al lenguaje es como sigue:

$$\langle real \rangle ::= \langle entero \rangle \langle fracción \rangle$$
$$\langle real \rangle ::= \langle ent\_c\_signo \rangle \langle fracción \rangle$$
$$\langle real \rangle ::= -\langle ent\_s\_signo \rangle \langle fracción \rangle$$
$$\langle real \rangle ::= -\langle dígito \rangle \langle dígito \rangle \langle dígito \rangle \langle fracción \rangle$$
$$\langle real \rangle ::= -\langle dígito \rangle \langle dígito \rangle \langle dígito \rangle \bullet \langle fracción \rangle$$
$$\langle real \rangle ::= -\langle dígito \rangle \langle dígito \rangle \langle dígito \rangle \bullet \langle dígito \rangle \langle dígito \rangle$$
$$\langle dígito \rangle ::= -153.87$$

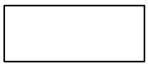
Con lo cual se concluye que la palabra -153.87 pertenece al lenguaje $L(G)$.

6.2. Diagrámas sintácticos

Ésta es una forma gráfica para representar una gramática por medio de gráficas dinámicas que permiten determinar en forma más ilustrativa si una palabra pertenece a un lenguaje; a esta gráfica se le conoce como diagrama sintáctico o diagrama de sintaxis. Toda palabra reservada de un lenguaje formal es susceptible de ser representada por medio de diagramas sintácticos (Jiménez, 2009).

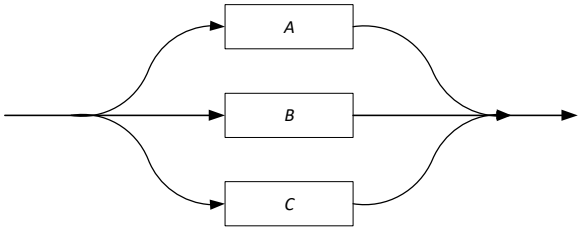
Constan de una serie de cajas o símbolos geométricos conectados por arcos dirigidos. Veamos los símbolos que rigen la construcción de cada grafo:

Tabla 2.3: Símbolos utilizados en los diagrámas de sintáxis.

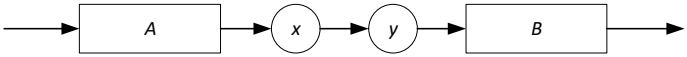
Símbolo	*Descripción*
	Símbolos no terminales.
	Símbolo terminal
	Indica la dirección del movimiento para completar la sustitución

Notación para las producciónes:

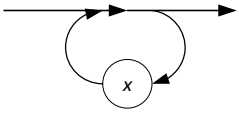
- Varias alternativas: $\langle A \rangle | \langle B \rangle | \langle C \rangle$



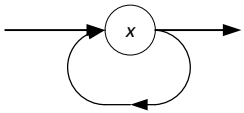
- Serie concatenada de símbolos terminales y/o no terminales: $\langle A \rangle xy \langle B \rangle$



- Cero o más repeticiones de un símbolo: $[x] = \lambda x...x$



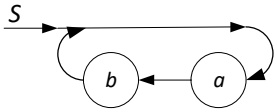
- Una o más repeticiones de un símbolo: $\{x\} = x...x$



Ejemplo: Se tiene el siguiente enunciado BNF:

$$\langle S \rangle ::= ab \langle S \rangle$$

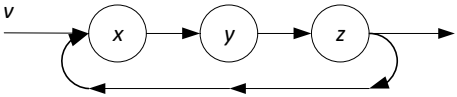
Construya el diagrama de sintaxis:



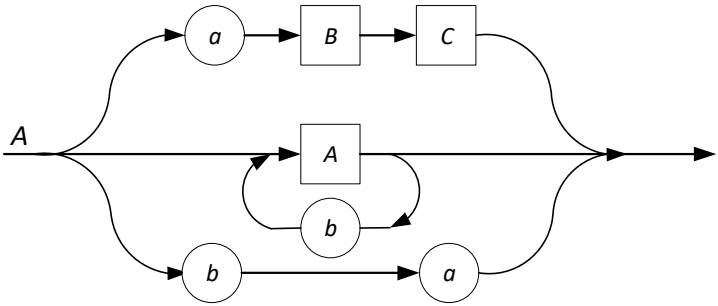
Ejemplo: Se tiene el enunciado BNF:

$$\langle v \rangle ::= xyz \mid xyz\langle v \rangle$$

Construya el diagrama de sintaxis:

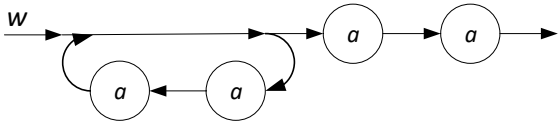


Ejemplo: El diagrama sintáctico que representa el siguiente enunciado BNF: $\langle A \rangle ::= a\langle B \rangle\langle C \rangle \mid \langle A \rangle b \mid ba$ es:



Ejemplo: Construya el diagrama de sintaxis para el siguiente enunciado: $\langle w \rangle ::= aa\langle w \rangle \mid aa$

El diagrama es el siguiente:

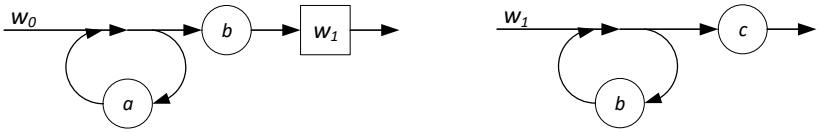


Ejemplo: Se tienen el siguiente enunciado BNF:

$$\langle w_0 \rangle ::= a\langle w_0 \rangle \mid b\langle w_1 \rangle$$

$$\langle w_1 \rangle ::= b\langle w_1 \rangle \mid c$$

Tenemos:



Ejemplo: Construya el diagrama de sintaxis para la gramática $G = (\Sigma_N, \Sigma_T, S, P)$, con las producciones dadas en BNF, donde:

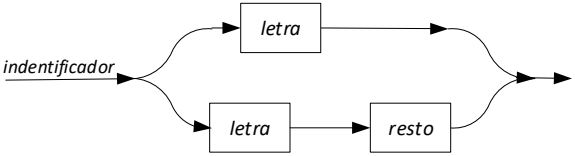
$\Sigma_T = \{a, b, c, \cdots, z, 0, 1, 2, 3, \cdots, 9\}$

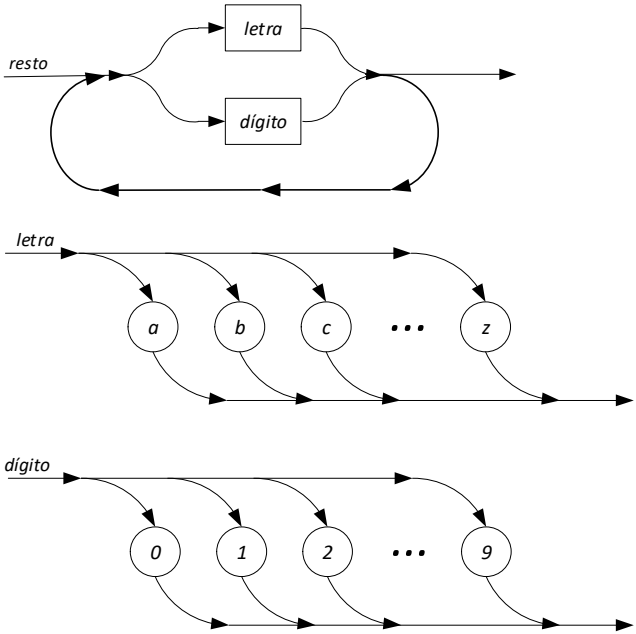
$\Sigma_N = \{identificador, resto, dígito, letra\}$

Producciones:

1) $\langle identificador \rangle ::= \langle letra \rangle \mid \langle letra \rangle \langle resto \rangle$
2) $\langle resto \rangle ::= \langle letra \rangle \mid \langle dígito \rangle \mid \langle letra \rangle \langle resto \rangle \mid \langle dígito \rangle \langle resto \rangle$
3) $\langle resto \rangle ::= a \mid b \mid c \cdots \mid z$
4) $\langle dígito \rangle ::= 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$

Se tiene:





Ejemplo: Considere la siguiente gramática $G = (\sum_N, \sum_T, S, P)$, con:

$$\sum_T = \{0,1,2,3,\cdots,9,a,b,c,\cdots,z\}$$

$$\sum_N = \{identificador, letra, dígito\}$$

Producciones:

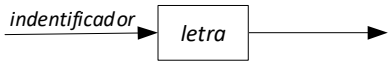
$\langle identificador \rangle ::= \langle letra \rangle$

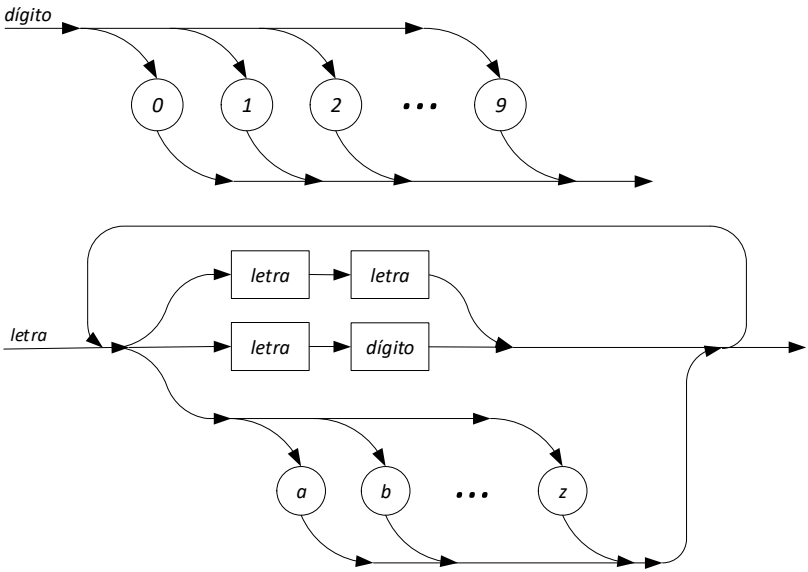
$\langle letra \rangle ::= \langle letra \rangle \langle letra \rangle \mid \langle letra \rangle \langle dígito \rangle \mid a \mid b \mid c \cdots \mid z$

$\langle dígito \rangle ::= 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$

Construya el diagrama de sintaxis.

Se tiene:





7. GRAMÁTICAS Y EXPRESIONES REGULARES

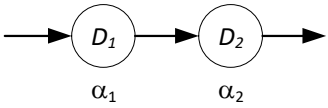
Según Kolman (1997), existe una conexión estrecha entre el lenguaje de una gramática regular y una expresión regular.

Teorema. – Sea S un conjunto finito y $L \subseteq S^*$. Entonces L es un conjunto regular si y solo si $L = L(G)$ para alguna gramática regular $G = (\sum_N, \sum_T, S, P_i)$.

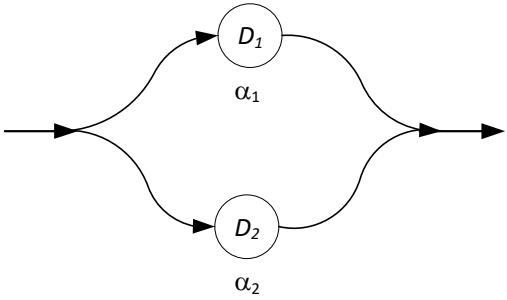
Definición. Sea $G = (\sum_N, \sum_T, S, P_i)$ una gramática regular y la relación $\rightarrow$ de G se especifica en forma BNF o en forma de diagrama sintáctico, se puede calcular la expresión regular de manera directa combinando todos los diagramas de sintaxis en un solo diagrama, la cual tenga únicamente símbolos terminale, se denomina ***diagrama maestro*** de G.

A continuación, se describen las reglas de correspondencia entre las expresiones regulares y las partes, o segmentos, del diagrama maestro *G*.

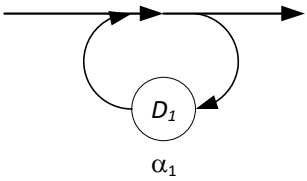
1) Los símbolos terminales del diagrama corresponden a si mismos, como expresiones regulares.
2) Si un segmento *D* del diagrama se compone de dos segmentos D_1 y D_2 en sucesión, y D_1 y D_2 corresponden a las expresiones regulares α_1 y α_2 respectivamente, entonces *D* corresponde a $\alpha_1\alpha_2$.



3) Si un segmento del diagrama se compone de segmentos alternativos D_1 y D_2, si D_1 y D_2 corresponden a las expresiones regulares α_1 y α_2 respectivamente, entonces *D* corresponde a $\alpha_1 \vee \alpha_2$.



4) Si un segmento *D* del diagrama es un ciclo a través del segmento D_1 y si D_1 corresponde a la expresión regular α, entonces D corresponde a α^*.



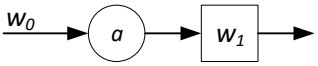
Ejemplo: La gramática $G = \left(\{w_0, w_1\}, \{a, b, c\}, S, P\right)$, con producciones:

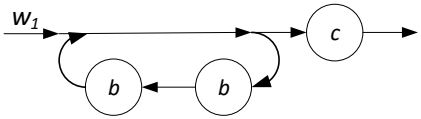
$$w_0 \to aw_1$$
$$w_1 \to bbw_1$$
$$w_1 \to c$$

Expresar en BNF y diagrama sintáctico.
Se tiene expresado en notación BNF:

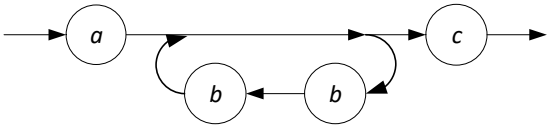
$$\langle w_0 \rangle ::= a \langle w_1 \rangle$$
$$\langle w_1 \rangle ::= bb \langle w_1 \rangle \mid c$$

y sus digramas son los siguientes:





Uniendo los dos diagramas, tenemos el diagrama maestro siguiente:



La expresión regular es: $a(bb)^* c$

Ejemplo: Sea la gramática $G = \left(\{A, B, C\}, \{+, \times, (,)\}, A, P\right)$ con reglas de producción:

$$\langle A \rangle ::= x \mid (\langle B \rangle)$$
$$\langle B \rangle \to \langle A \rangle \langle C \rangle$$
$$\langle C \rangle ::= + \langle A \rangle \langle C \rangle$$

a) Determine los diagramas sintácticos
Diagrama sintáctico de: $\langle A \rangle ::= x \mid (\langle B \rangle)$

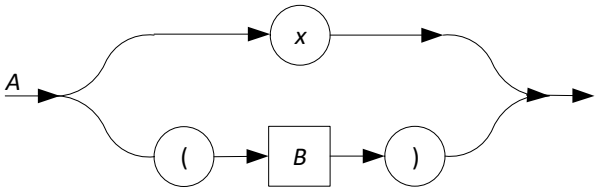


Diagrama sintáctico de: $\langle B \rangle \rightarrow \langle A \rangle \langle C \rangle$

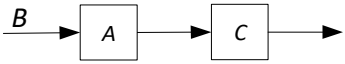
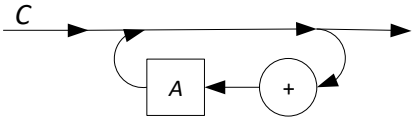
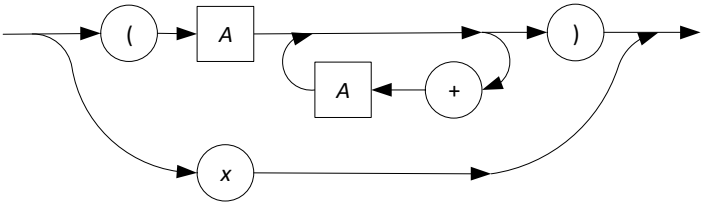


Diagrama sintáctico de: $\langle C \rangle ::= + \langle A \rangle \langle C \rangle$



En este caso, los diagramas sintácticos pueden combinarse para formar un diagrama sintáctico maestro que representa a todo el lenguaje:



b) Ejemplos de cadenas pertenecientes al lenguaje:

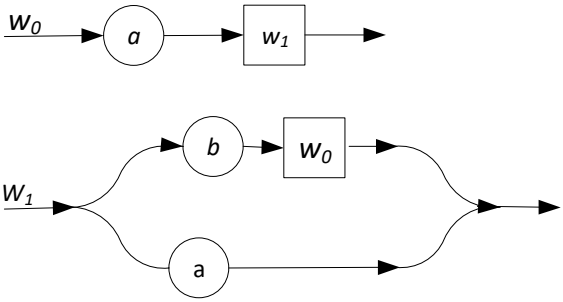
$$x, (x), (x+x), (((x))), (x+x+x), (x+x+x+x+x)$$

Ejemplo: Sea la gramática $G = \left(\{w_0, w_1\}, \{a, b\}, S, P\right)$, con producciones en BNF:

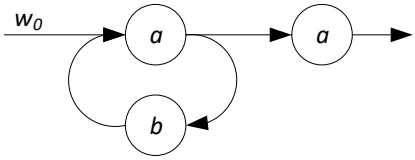
$$\langle w_0 \rangle ::= a \langle w_1 \rangle$$
$$\langle w_1 \rangle ::= b \langle w_0 \rangle \mid a$$

Determine el diagrama sintactico maestro y la expresión regular que genera.

Se tiene:



El diagrama sintactico maestro es:



La expresión regular generada: $(ab)^* aa$

EJERCICIOS PROPUESTOS

1. Crear una gramática que genere los siguientes lenguajes:
   a) $\{a, aa, aaa\}$
   b) $\{a, aa, aaa, aaaa, \cdots\}$
   c) $\{\lambda, a, aa, aaa\}$
   d) $\{\lambda, a, aa, aaa, aaaa, \cdots\}$

2. Dada la gramática $G_1 = (\{S, A\}, \{0,1,2\}, S, P)$ cuyas producciones P son $\{S \to 0A \mid 2, A \to 0S \mid 1\}$. Muestre si las cadenas 002 y 001 pertenecen al lenguaje de *G*.

3. Dada las siguientes gramáticas $G = (\Sigma_N, \Sigma_T, S, P_i)$ donde:
   a) $\Sigma_N = \{S, A\}$, $\Sigma_T = \{c\}$ y $P_1 = \{(S \to \lambda \mid A), (A \to AA \mid c)\}$
   b) $\Sigma_N = \{S, A\}$, $\Sigma_T = \{c, d\}$ y $P_2 = \{(S \to \lambda \mid A), (A \to cAd \mid cd)\}$
   c) $\Sigma_N = \{S, A\}$, $\Sigma_T = \{c\}$ y $P_3 = \{(S \to \lambda \mid A), (A \to AcA \mid c)\}$
   d) $\Sigma_N = \{S, A, T\}$, $\Sigma_T = \{c, d\}$ y
      $P_4 = \{(S \to cA), (A \to d \mid cA \mid Td), (T \to Td \mid d)\}$
   e) $\Sigma_N = \{S, A\}$, $\Sigma_T = \{c, d\}$ y $P_5 = \{(S \to \lambda \mid A), (A \to Ad \mid cA \mid \mathrm{c} \mid d)\}$

   Determine el lenguaje y su respectivo árbol asociado a dichas gramáticas.

4. Se tiene $G = (\Sigma_N, \Sigma_T, S, P)$, donde: $\Sigma_N = \{S\}$, $\Sigma_T = \{a, b\}$, $P = \{(S \to aaS), (S \to a), (S \to b)\}$. Construya el lenguaje.

5. Una expresión regular es ambigua si hay una palabra que puede ser obtenida de la expresión regular de, como mínimo, dos formas diferentes. De las expresiones regulares siguientes, cuáles son ambiguas:

a) $a\left((ab)^* cd\right)^* + a\left(ababcb^*\right)^* a^*$

b) $aab^*\left(ab\right)^* + ab^* + a^*bba^*$

c) $aaba^* + aaaba + aabba^* + a$

6. Construya el diagrama de sisntaxis para el siguiente enunciado BNF:

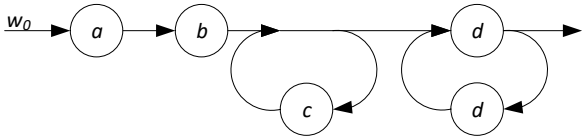
$\langle w_0 \rangle ::= aa\langle w_0 \rangle \mid b\langle w_1 \rangle$

$\langle w_1 \rangle ::= c\langle w_2 \rangle b \mid cb$

$\langle w_2 \rangle ::= bb\langle w_2 \rangle bb$

7. Proporcione una expresión regular y notación BNF para el diagrama de sintaxis que se muestra en la figura.

Nota: Usted puede proporcionar los símbolos no terminales necesarios, para utilizarlos en las producciones BNF.



Capítulo 3

AUTÓMATAS FINITOS

INTRODUCCIÓN

La Teoría de Autómatas es una rama de la Teoría Computacional que estudia las máquinas teóricas llamadas autómatas. Estas máquinas son modelos matemáticos.

Un Autómata está formado por un conjunto de estados, uno de los cuales es el estado en el que la máquina se encuentra inicialmente. Recibe como entrada una palabra (una concatenación de símbolos del alfabeto del autómata) y según esta palabra la máquina puede cambiar de estados.

Los Autómatas se clasifican según el número de estados (finito o no finito), la forma en que se realiza el cambio de estado (determinista o no determinista), si acepta o no el símbolo vacío λ, si tiene o no una pila, etc.

Los Autómatas están muy relacionados con la máquina de Turing (1936), de gran importancia en la Teoría Computacional. Esto se debe a que una máquina de Turing puede simular el almacenamiento y la unidad de control de una computadora. Tenemos certeza de que lo que no puede ser resuelto por una máquina de Turing no puede ser resuelto por una computadora real.

1. AUTÓMATA FINITO DETERMINISTA (AFD)

Los Autómatas Finitos son máquinas teóricas que van cambiando de estado dependiendo de la entrada que reciban. La salida de estos Autómatas está limitada a dos valores: aceptado y no aceptado, que pueden indicar si la cadena que se ha recibido como entrada es o no válida. Generalmente utilizaremos los Autómatas Finitos para reconocer lenguajes regulares, es decir, una palabra o cadena se considerará válida sólo si pertenece a un determinado lenguaje (Jurado, 2008).

Definición. Un Autómata Finito Determinista (AFD) M se define como una quintupla.

$$M = (Q, \Sigma, \delta, q_0, F)$$

Donde:

$Q = \{q_0, q_1, \cdots q_n\}$: Conjunto finito de estados del autómata.

Σ : Es un alfabeto finito, llamado también alfabeto del autómata.

$q_0 \in Q$: Estado inicial o arranque del autómata.

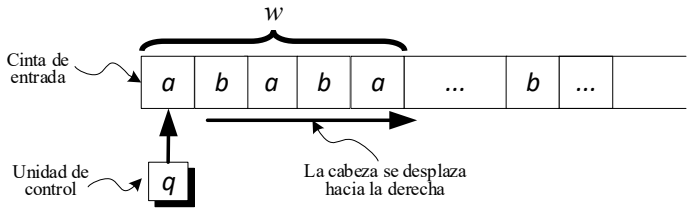
$F \subseteq Q$: Conjunto de estados finales o de aceptación. $F \neq \varnothing$

δ : Es la función de transición del autómata, la cual se define:

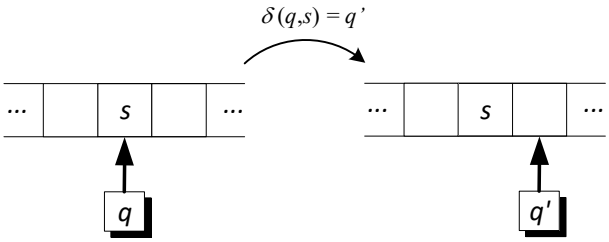
$$\delta : Q \times \Sigma \rightarrow \quad Q$$
$$(q, s) \rightarrow \delta(q, s)$$

En ese sentido, podemos decir que un automata finito es deterministico si por medio de la función de transicion $\delta : Q \times \Sigma \rightarrow Q$ es posible determinar claramente cual es el estado siguiente.

Una cadena de entrada w se coloca en la cinta de tal manera que el primer símbolo de w ocupa la primera casilla de la cinta. La unidad de control está inicialmente en el estado q_0 escaneando la primera casilla:



La función de transición δ indica el estado al cual pasa el control finito, dependiendo del símbolo escaneado y de su estado actual. Así, $\delta(q,s)=q'$ significa que, en presencia del símbolo *s*, la unidad de control pasa del estado *q* al estado *q*'.



Esta acción constituye un paso computacional.

1.1. Representación de Autómatas

Existen dos formas de representar un AFD, mediante tablas detransición o mediante diagramas de transición. Con el siguiente ejemplo mostraremos estas dos representaciones.

Ejemplo: Sea el siguiente AFD $M=(Q,\Sigma,\delta,q_0,F)$ donde: $Q=\{q_0,q_1\}$, $\Sigma=\{0,1\}$, $q_0=\{q_0\}$, $F=\{q_1\}$ y δ se defide de la siguiente forma:

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

a) Tabla de transición

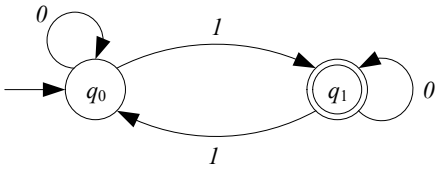
La función δ pueden representarse mediante una tabla que representa los valores de la función de transición, con tantas filas como estados y tantas columnas como entradas.

Se tiene la siguiente tabla de transición para el ejemplo:

δ	0	1
q_0	q_0	q_1
q_1	q_1	q_0

b) Diagrama de transición

El diagrama de transición es un grafo dirigido en el que cada nodo corresponde a un estado; y si $\delta(q_i, a) = q_j$ existe un arco dirigido del nodo q_i al correspondiente q_j, sobre el que se pone la etiqueta *a*. Para el ejemplo se tiene:



Los estados finales de aceptación se identifican por estar encerrados en un doble circulo. El estado inicial se representa con una flecha.

El AFD acepta cadenas que comienzan con 0 y terminan con 0. Por ejemplo, se podría escribir una expresión regular para las cadenas de 1's y 0's que tuvieran una cantidad impar de 1's.

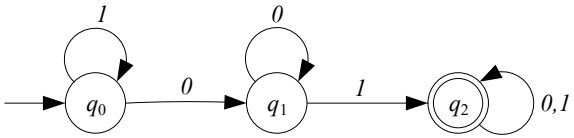
$$0^*\left(10^*10^*\right)^*10^* \qquad \text{ó} \qquad \left(0^*10^*10^*\right)^*10^*$$

De forma que las cadenas 0101010, 0001010100, 0100001010, pertenecen al lenguaje, pero las cadenas 0000, 0011, 011101 no son reconocidas por el AFD.

Ejemplo: Sea tiene $M=(Q,\Sigma,\delta,q_0,F)$ donde: $Q=\{q_0,q_1,q_2\}$, $\Sigma=\{0,1\}$, $q_0=\{q_0\}$, $F=\{q_2\}$ y cuyo tabla de transición esta dado por:

δ	0	1
q_0	q_1	q_0
q_1	q_1	q_2
q_2	q_2	q_2

El diagrama de transición es el siguiente:



Por tanto, las funciones de transición serían:

$$\delta(q_0,0)=q_1 \qquad \delta(q_0,1)=q_0$$
$$\delta(q_1,0)=q_1 \qquad \delta(q_1,1)=q_2$$
$$\delta(q_2,0)=q_2 \qquad \delta(q_2,1)=q_2$$

Observando el diagrama de transición del AFD se pude deducir que acepta todas las cadenas que continen la subcadena 01, de forma que la expresión regular sería:

$$1^*00^*1(0\,|\,1)^*$$

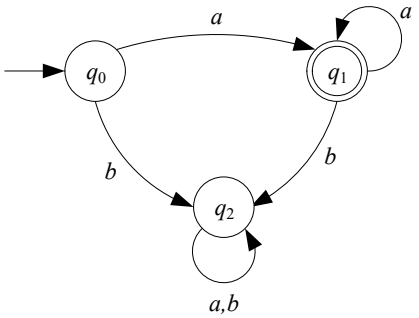
Ejemplo: Si $M=(Q,\Sigma,\delta,q_0,F)$ donde: $Q=\{q_0,q_1,q_2\}$, $\Sigma=\{a,b\}$, $q_0=\{q_0\}$, $F=\{q_1\}$ y δ se defide de la siguiente forma:

$$\delta(q_0,a)=q_1 \qquad \delta(q_0,b)=q_2$$
$$\delta(q_1,a)=q_1 \qquad \delta(q_1,b)=q_2$$
$$\delta(q_2,a)=q_2 \qquad \delta(q_2,b)=q_2$$

Se tiene la siguiente tabla de transición:

δ	a	b
q_0	q_1	q_2
q_1	q_1	q_2
q_2	q_2	q_2

Y cuyo diagrama de transición es:



Al analizar el AFD es evidente que sólo considera como cadenas aceptadas aquellas que están formadas únicamente por *a*'s. Cualquier otra cadena que contenga una *b* hará que el autómata termine en el estado q_2, que es un estado muerto o de abosorción.

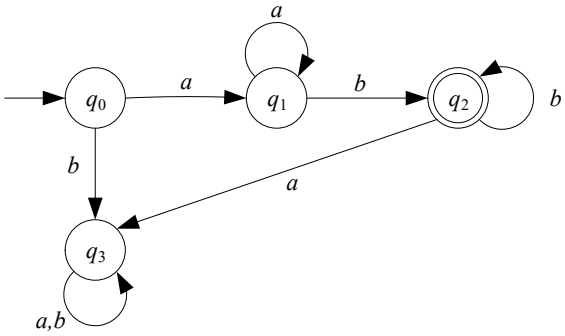
Las siguientes cadenas: *a*, *aa*, *aaa*, ..., *aaaa…a,* son aceptadas por el autómata y la expresión regular sería: aa^*

Mostramos algunas cadenas como *abbba*, *babaaa*, *aaaab*, *aaaabaaaa* que no son reconocidas o aceptadas por el AFD.

Definición. Se define un estado de ***absorción*** o ***muerte*** como aquel estado $q_i \in Q$ que no es un estado final de aceptación y no parte de él ninguna transición hacia otro estado. Es decir,

$$\delta(q,\mathrm{a}) = \varnothing \text{ ó } \delta(q,\mathrm{a}) = q\,,\ \forall a \in \Sigma$$

Ejemplo: Sea el autómata finito M donde: $Q = \{q_0, q_1, q_2, q_3\}$, $\Sigma = \{a, b\}$, $q_0 = \{q_0\}$, $F = \{q_2\}$ y cuyo diagrama de transición es el siguiente:



Se observa que, el estado q_3 no tiene ninguna transición, únicamente hay transiciones que inciden en él. Además, $q_3 \in F$, por tanto, q_3 es un estado de abosorción o muerte.

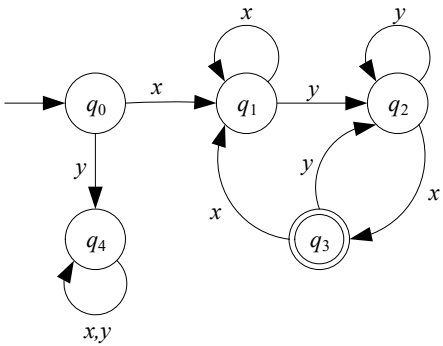
En cuanto a las cadenas producidas por el Autómata, reconce todas las cadenas que inicien con *a* y terminen en *b*, es decir:

ab, *aab*, …, *abbb*, *aabb*, …

y la expresión regular es: aa^*bb^*

En cambio, todas las cadenas que inicién con *b* o contengan la subcadena *ba*, tales como: *bbba*, *baaaba*, *abaaaa*, …, serán rechazadas por el autómata, pues q_3 no es un estado de aceptación.

Ejemplo: Sea tiene el autómata finito M en donde: $Q = \{q_0, q_1, q_2, q_3, q_4\}$, $\sum = \{x, y\}$, $q_0 = \{q_0\}$, $F = \{q_3\}$ cuyo diagrama de transición es:



Por tanto, la tabla de transición es como sigue:

δ	x	y
q_0	q_1	q_4
q_1	q_1	q_2
q_2	q_3	q_2
q_3	q_1	q_2
q_4	q_4	q_4

El autómata acepta cadenas que comienzan con *xy* y terminan con *yx*. La expresión regular estaría representada por:

$$x(x \cup y)^* yx$$

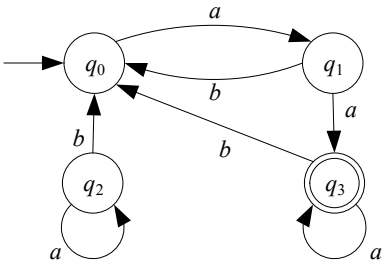
por consiguiente, tenemos algunas cadenas como *xxxyxyyyx*, *xyx*, *xyyyx*, son producidas por el autómata, pero las cadenas *yyxyx*, *xyxxy* no son reconocidas por el AFD.

Definición. Sea $M = (Q, \Sigma, \delta, q_0, F)$ un AFD. Un estado $q \in Q$ decimos que es ***inaccesible***, si no existe ninguna palabra sobre el alfabeto de entrada que partiendo desde q_0 llegue a q. Es decir,

$$q \text{ es inaccesible si, } \forall a \in \Sigma^*,\ \delta(q_0, a) \neq q$$

Los estados que no son inaccesibles decimos que son ***accesibles*** y todas sus transiciones, por tanto, el AFD obtenido es equivalente al dado.

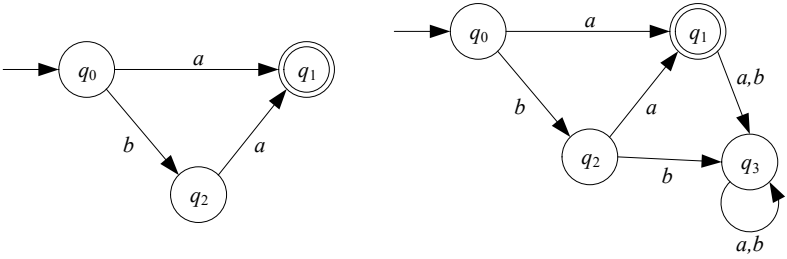
Ejemplo: En el diagrama del AFD, el estado q_2 es inaccesible porque no se puede llegar a partir del estado inicial.



Definiciones:

- **Autómata completo. –** Un AFD se dice completo cuando todos sus estados tienen transiciones con cada uno de los símbolos de entrada válidos.
- **Autómata conexo. –** Un AFD es conexo si todos sus estados son accesibles desde el estado inicial.

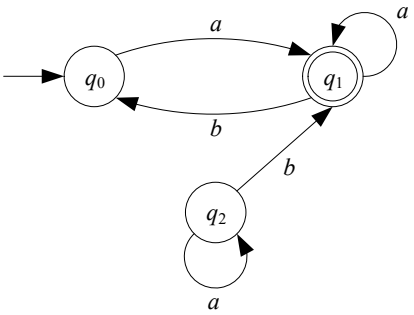
Ejemplo: Se tiene los siguientes autómatas:



a) b)

En los diagramas mostrados podemos ver un ejemplo de estasituación. El autómata de la izquierda a) está incompleto, pero podemos completarlo trasformándolo en el de la derecha b), al que hemos añadido el estado q_3, que es un estado muerto.

Ejemplo: El autómata presentado en el siguiente diagrama no es conexo y su parte conexa es la formada por los estados q_0 y q_1 y por las transiciones que hay entre ellos.



1.2. Lenguaje de un Autómata

Según Jurado (2008) define el lenguaje de un AFD, como:

Definición. El lenguaje que acepta un AFD es el conjunto de palabras definidas sobre Σ que hacen que el autómata llegue a un estado final de aceptación y se define:

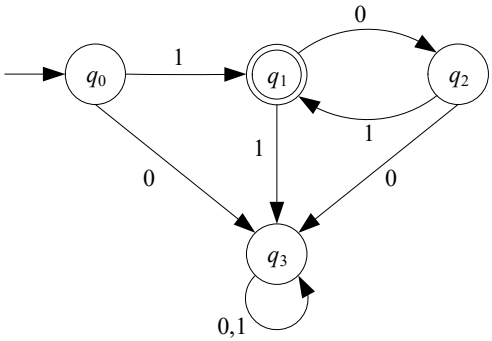
$$L(M) = \{w \in \Sigma^* / \delta(q_0, w) \in F\}$$

Ejemplo: Sea el autómata finito M donde: $\Sigma = \{0,1\}$, $Q = \{q_0, q_1, q_2, q_3\}$, $q_0 = \{q_1\}$, $F = \{q_1\}$ y δ se define por la siguiente tabla:

δ	0	1
q_0	q_3	q_1
q_1	q_2	q_3
q_2	q_3	q_1
q_3	q_3	q_3

Construya el diagrama de transición y determinar el lenguaje que reconoce, denotándola con su expresión regular.

De acuerdo a la función de transición se tiene el siguiente diagrama:



El lenguaje que genera el automata es:

$$L(M) = \{1,101,10101,...\} = \{1(01)^n / n \geq 0\}$$

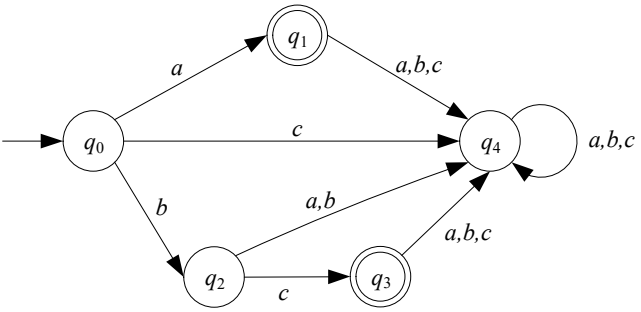
La expresión regular sería: $1(01)^*$

Ejemplo: Sea el AFD $M = (Q, \Sigma, \delta, q_0, F)$ donde: $\Sigma = \{a,b,c\}$, $Q = \{q_0, q_1, q_2, q_3, q_4\}$, $F = \{q_1, q_3\}$ y δ se define:

δ	a	b	c
q_0	q_1	q_2	q_4
q_1	q_4	q_4	q_4
q_2	q_4	q_4	q_3
q_3	q_4	q_4	q_4
q_4	q_4	q_4	q_4

Grafique el diagrama de transición, determinar el lenguaje que reconoce y denotarlo con una expresión regular.

Aligual que los ejemplos ateriores se construye el diagrama de transición.



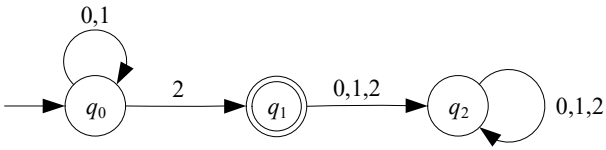
El lenguaje generado es $L(M) = \{a, bc\}$ y la expresión regular es $a \mid bc$.

Ejemplo: Sea el AFD $M=\left(\{q_0,q_1,q_2\},\{0,1,2\},\delta,\{q_0\},\{q_1\}\right)$ y δ se define por la siguiente tabla:

δ	0	1	2
q_0	q_0	q_0	q_1
q_1	q_2	q_2	q_2
q_2	q_2	q_2	q_2

Construya el diagrama de transición, el lenguaje que genera y expresión regular.

El diagrama de transición es:



El lenguaje reconocido por el automata es el siguiente:

$$L(M)=\{2,02,002,0002,...,012,01112,...,102,10002,...\}$$

$$L(M)=\left\{0^{n_1}1^{n_2}0^{n_3}1^{n_4}2/n_1,n_2,n_3,n_4\geq 0\right\}$$

La expresión regular es $(0\,|\,1)^*2$

1.3. Diseño de Autómatas

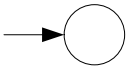
Para el diseño o diagramación de los autómatas deterministas se utiliza la siguiente convención adicional con respecto a los diagramas de transiciones: se supone que los arcos no dibujados explícitamente conducen a un estado de no-aceptación. Es decir, en el diagrama de transiciones se indican únicamente los arcos que intervienen en trayectorias de aceptación. Esto permite simplificar considerablemente los diagramas. (De Castro, 2004)

Trataremos dos tipos de problemas:

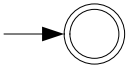
i. Dado un lenguaje regular L diseñar un autómata finito M que acepte o reconozca a L, es decir, tal que $L(M) = L$.
ii. Dado un autómata M determinar el lenguaje aceptado por M.

Ejemplo:

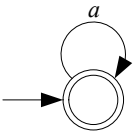
1) Si $L = \varnothing$. El lenguaje aceptado por AFD es: $L(M) = \varnothing$



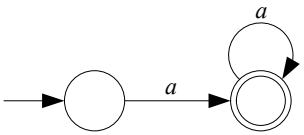
2) Si $L = \lambda = \{\lambda\}$. El lenguaje aceptado por AFD es: $L(M) = \lambda$



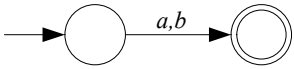
3) Si $L = a^* = \{\lambda, a, a^2, a^3, \cdots\}$. El lenguaje aceptado por AFD es: $L(M) = \{a\}^*$



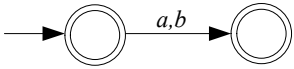
4) Si $L = a^+ = \{a, a^2, a^3, \cdots\}$. El lenguaje aceptado por AFD es: $L(M) = \{a\}^+$



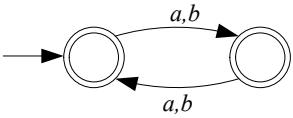
5) Si $L = a + b = a \mid b$. El lenguaje aceptado por AFD es: $L(M) = \{a, b\}$

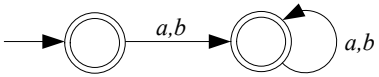


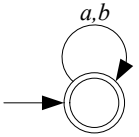
6) Si $L = \lambda + a + b$. El lenguaje aceptado por AFD es: $L(M) = \{\lambda, a, b\}$



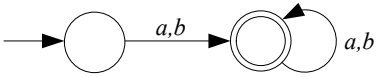
7) Si $L = (a + b)^*$. El lenguaje aceptado por AFD es: $L(M) = \{a, b\}^*$







8) Si $L = (a + b)(a + b)^*$. El lenguaje aceptado por AFD es: $L(M) = \{a, b\}\{a, b\}^*$



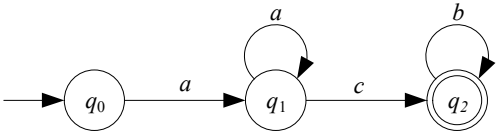
Ejemplo: Construya un autómata finito determinístico que acepte el lenguje $L = \{a^n c b^m / n > 0, m \geq 0\}$

Analizando el lenguaje podemos indicar que:

- se utiliza tres símbolos $\Sigma = \{a, b, c\}$
- tres estados como mínimo $Q = \{q_0, q_1, q_2\}$
- dado que $n > 0$, esto nos indica que inicia con al menos una *a* y luego pude haber varias *a*'s

- el lenguje tiene una sola *c*
- y como $m \geq 0$, nos indica que puede terminar sin ninguna *b*, una o varias b's

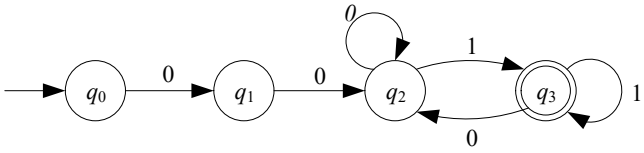
Por tanto, el diagrama de transición del AFD sería:



y el AFD estaría definido por $M = \left(\{q_0, q_1, q_2\}, \{a, b, c\}, \delta, \{q_0\}, \{q_2\}\right)$

Ejemplo: Construya un AFD que acepte el lenguje $L = \left\{00(0,1)^* 1\right\}$

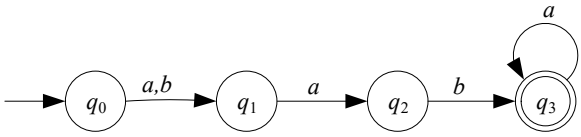
Analizando el lenguaje se tiene el diagrama de transición del AFD:



y el AFD sería $M = \left(\{q_0, q_1, q_2, q_3\}, \{0,1\}, \delta, \{q_0\}, \{q_3\}\right)$

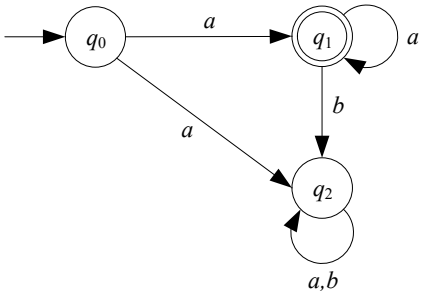
Ejemplo: Implemente el AFD que reconozca el siguiente lenguaje: $(a \mid b)aba^*$

El autómata que produzca dicho lenguaje es:

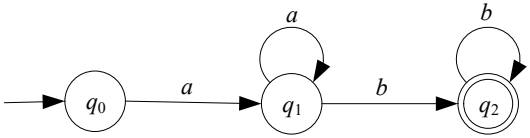


Ejemplo: Diseñe AFD sobre el alfabeto $\sum = \{a, b\}$, que reconozca el lenguaje:

a) $L = a^+ = \{a, aa, aaa, ...\}$

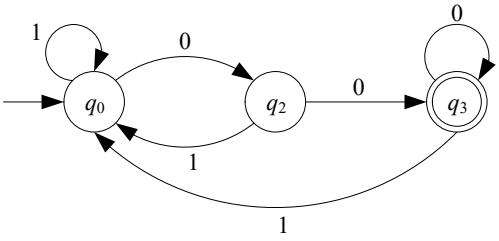


b) $L = a^+ b^+$

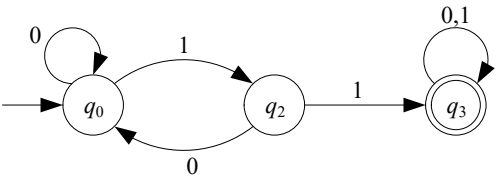


Ejemplo: Se tiene el alfabeto $\sum = \{0,1\}$, dibujar los diagramas de transición que reconozcan:

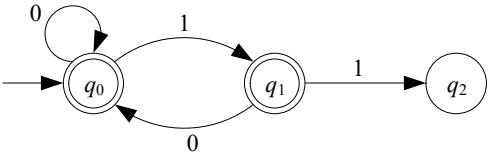
a) Cadenas terminadas en en 00



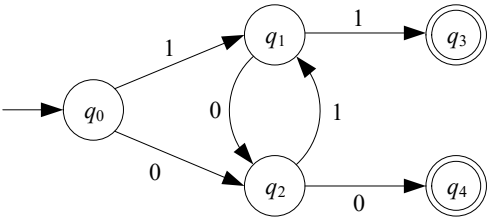
b) Cadenas con dos 1's consecutivos



c) Cadenas que no contengan dos 1's consecutivos

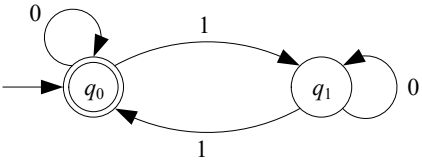


d) Cadenas con dos 0´s consecutivos o dos 1's consecutivos



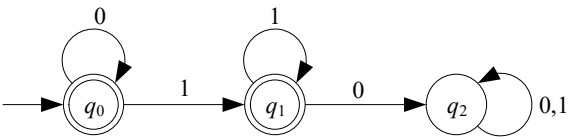
Ejemplo: Determinar los lenguajes aceptados por los siguientes AFD. Describir los lenguajes por medio de una expresión regular.

a)



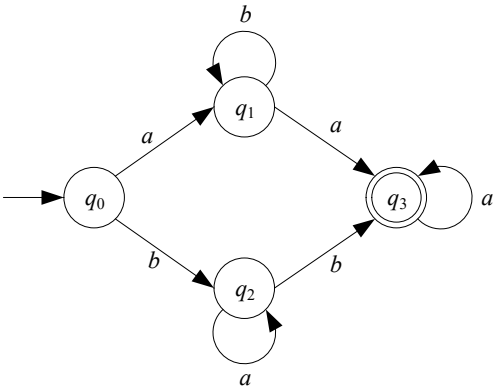
El lenguaje reconocido es: $0^*(10^*1)^*0^*$

b)



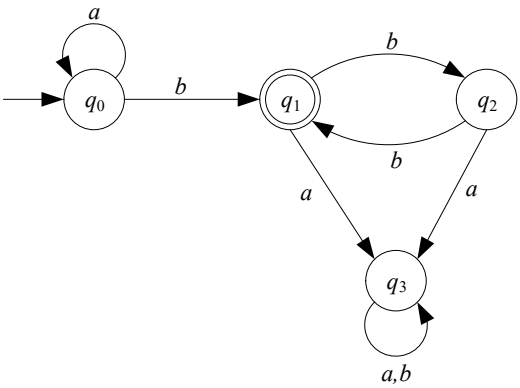
El lenguaje reconocido es: 0^*11^*

c)



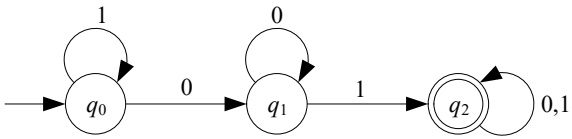
El lenguaje reconocido es el siguiente: $(ab^*a \mid ba^*b)a^*$

d)



El lenguaje reconocido es: $a^*b(bb)^*$

e)



El lenguaje reconocido es el siguiente: $1^*00^*1(0\,|\,1)^*$

2. AUTOMATA FINITO NO DETERMINISTA (AFND)

Es el autómata finito que tiene transiciones vacías (λ) o que por cada símbolo desde un estado de origen se llega a más de un estado destino, es decir, es aquel que, a diferencia de los autómatas finitos deterministas, posee al menos un estado, tal que, para un símbolo del alfabeto, existe más de una transición posible.

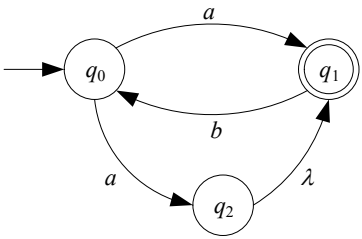
Haciendo la analogía con los AFDs, en un AFND puede darse cualquiera de estos dos casos:

- Que existan transiciones del tipo $\delta(q,a)=q_1$ y $\delta(q,a)=q_2$ siendo $q_1 \neq q_2$.

- Que existan transiciones del tipo, $\delta(q,\lambda)$ siendo un estado no-final, o bien un estado final, pero con transiciones hacia otros estados.

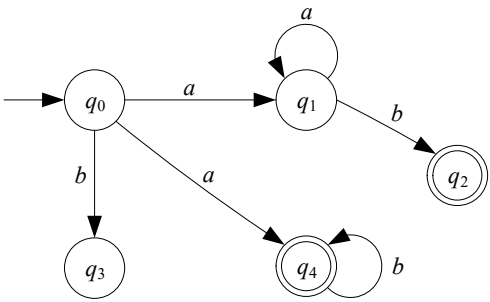
Cuando se cumple el segundo caso, se dice que el autómata es un autómata finito no determinista con transiciones vacías o transiciones λ. Estas transiciones permiten al autómata cambiar de estado sin procesar ningún símbolo de entrada.

Ejemplo: Considere el siguiente diagrama:



En este diagrama se observa que, el estado q_0, para el símbolo "*a*" tiene dos transiciones, una primera al estado q_1 y la otra hacia el estado q_2, es decir, $\delta(q_0, a) = \{q_1, q_2\}$. Así mismo, presenta una transición del estado q_2 al q_1 sin necesidad de leer ningún carácter de entrada, es decir, una transición λ.

Ejemplo: Se tiene el siguiente diagrama:



Al igual que en el ejemplo anterior, del estado q_0 salen con el símbolo "*a*" a dos estados, q_1 y q_4, es decir, $\delta(q_0, a) = \{q_1, q_4\}$.

Definición. Un Autómata Finito No Determinista (AFND) M se define como una quintupla.

$$M = (Q, \Sigma, \Delta, q_0, F)$$

Donde:

$Q = \{q_0, q_1, \cdots q_n\}$: Conjunto finito de estados.

Σ : Es un alfabeto finito.

$q_0 \in Q$: Estado inicial o estado de partida.

$F \subseteq Q$: Es el conjunto de estados finales o de aceptación.

Δ : Es la función de transición y se define:

$$\Delta : Q \times \Sigma^* \to \wp(Q)$$

$$\Delta(q, s) \subseteq Q$$

siendo $\wp(Q)$ el conjunto de las partes de Q.

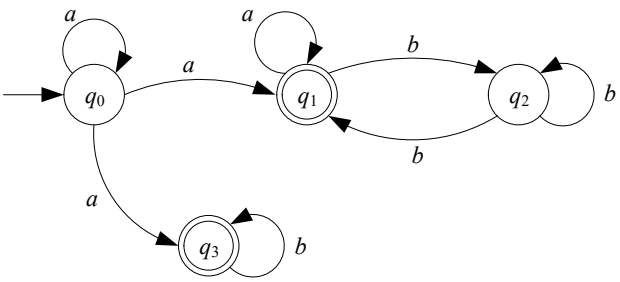
Ejemplo: Si $Q = \{q_0, q_1\}$ entonces, $\wp(Q) = \{\varnothing, \{q_0\}, \{q_1\}, \{q_0, q_1\}\}$

Además, un AFND puede estar en varios estados a la vez.

Ejemplo: Sea $M = (Q, \Sigma, \Delta, q_0, F)$ un AFND, donde: $Q = \{q_0, q_1, q_2, q_3\}$, $\Sigma = \{a, b\}$, $q_0 = \{q_0\}$, $F = \{q_1, q_3\}$ y cuya tabla de transición se defide:

Δ	a	b
q_0	$\{q_0, q_1, q_3\}$	$\varnothing$
q_1	$\{q_1\}$	$\{q_2\}$
q_2	$\varnothing$	$\{q_1, q_2\}$
q_3	$\varnothing$	$\{q_3\}$

A partir de la tabla de transición se construye el siguiente diagrama de transición:



Es importante mencionar que, hay cadenas que conducen al rechazo y también terminan en estados de aceptación, tal es el caso de la cadena $w = abb$.

Cómputo de rechazo: $q_0 \to q_1 \to q_2 \to q_2$

Computo de aceptación: $q_0 \to q_3 \to q_3 \to q_3$

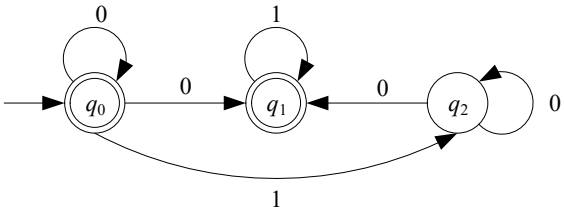
Computo de aceptación: $q_0 \to q_3 \to q_3 \to q_1$

Computo de abortado: $q_0 \to q_0$

Observación.

- En los ANFDs no existen los estados de muerte o absorción.
- Sean M_1 y M_2 dos AFNDs, $M_1 \equiv M_2 \Leftrightarrow L(M_1) = L(M_2)$.
- Los ANFDs reconocen los lenguajes regulares.

Ejemplo: Sea $M = (Q, \Sigma, \Delta, q_0, F)$ un AFND cuyo diagrama es:



donde: $Q = \{q_0, q_1, q_2\}$, $\Sigma = \{0,1\}$, $q_0 = \{q_0\}$, $F = \{q_1, q_2\}$ se pide:

a) Construir la tabla de trasición de estados

Δ	0	1
q_0	$\{q_0, q_1\}$	$\{q_2\}$
q_1	$\varnothing$	$\{q_1\}$
q_2	$\{q_1, q_2\}$	$\varnothing$

b) Defina las funciones de transición

$$\Delta(q_0,0)=\{q_0,q_1\} \qquad \Delta(q_0,1)=\{q_2\}$$
$$\Delta(q_1,0)=\varnothing \qquad \Delta(q_1,1)=\{q_1\}$$
$$\Delta(q_2,0)=\{q_1,q_2\} \qquad \Delta(q_2,1)=\varnothing$$

c) Establezca el camino de aceptación para las siguientes cadenas: 00100111, 1110, 11010, λ.

Una cadena es aceptada por un AFND si existe algún camino para esa cadena que comience en el estado inicial y termine en el estado aceptado o final.

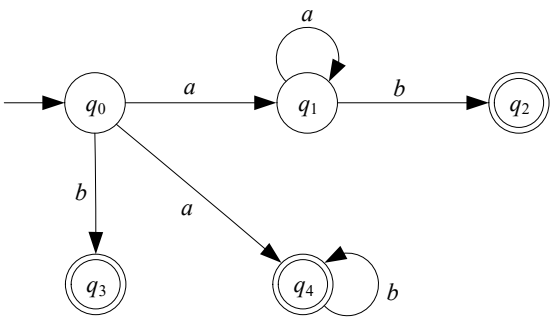
Cadena	Aceptación	Camino
00100111	Si	$q_0, q_0, q_2, q_2, q_1, q_1, q_1, q_1$
1110	No	q_0, q_2 abortado
11010	No	q_2 abortado
λ	Si	Dado que es estado q_0 es inicial y estado de aceptación a la vez.

d) Lenguaje y expresión regular aceptado por el autómata

$$L=\left\{0^n\left(\left\{01^m\right\}\cup\left\{10^p01^q\right\}\right)\Big/ n,m,p,q\geq 0\right\}$$

y su expresión regular es: $0^*\left(01^*\,|\,10^*01^*\right)$

Ejemplo: Se tiene el siguiente diagrama de un AFND:



donde: $Q=\{q_0,q_1,q_2,q_3,q_4\}$, $\Sigma=\{a,b\}$, $q_0=\{q_0\}$, $F=\{q_2,q_3,q_4\}$, se pide:

a) Lenguaje y expresión regular aceptado por el autómata

En un primer caso, podemos analizar la aceptación de una única cadena, con transición de q_0 a q_3: *b*

Luego, con la transición de q_0 a q_2 con las cadenas: *ab*, *aab*, *aa…ab*.

De estos dos análisis podemos establecer el siguiente lenguaje:

$$L_1=\{b\cup a^n b/n>0\}$$

Fijese que si $n \geq 0$, indicaría $\{b\cup b\}=\{b\}$, entonces podemos escribir $L_1=\{a^n b/n\geq 0\}$.

Finalmente, con la transición de q_0 a q_4 con las cadenas: *ab*, *abb*, *ab…bb*. Y el lenguaje aceptado sería:

$$L_2=\{ab^m/m\geq 0\}$$

Por tanto, el lenguaje aceptado por el AFND, es:

$$L = \left\{ a^n b \cup a b^m / n, m \geq 0 \right\}$$

y su expresión regular es: $a^* b \mid a b^*$

b) Construir la tabla de trasición de estados

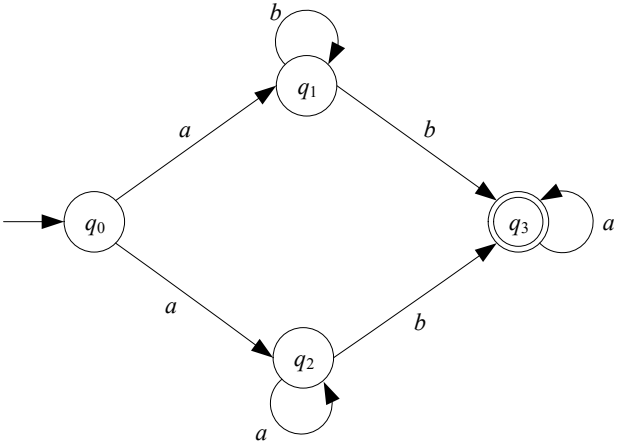
Δ	a	b
q_0	$\{q_1, q_4\}$	$\{q_3\}$
q_1	$\{q_1\}$	$\{q_2\}$
q_2	$\varnothing$	$\varnothing$
q_3	$\varnothing$	$\varnothing$
q_4	$\varnothing$	$\{q_4\}$

Ejemplo: Sea el AFND $M = (Q, \Sigma, \Delta, q_0, F)$, donde: $Q = \{q_0, q_1, q_2, q_3\}$, $\Sigma = \{a, b\}$, $q_0 = \{q_0\}$, $F = \{q_4\}$ y la unción Δ esta dada por la siguiente tabla:

Δ	a	b
q_0	$\{q_1, q_2\}$	$\varnothing$
q_1	$\varnothing$	$\{q_1, q_3\}$
q_2	$\{q_2\}$	$\{q_3\}$
q_3	$\{q_3\}$	$\varnothing$

Determinar el lenguaje que reconoce, y dar su expresión regular.

El diagrama de transición es el siguiente:

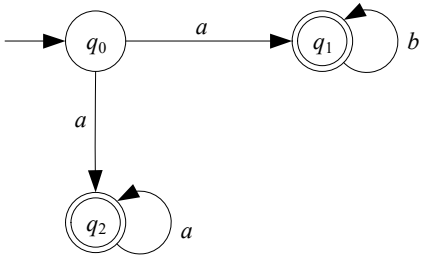


El lenguaje aceptado es: $L=\left\{a\left(b^{n}b\cup a^{m}b\right)a^{p}\,/\,n,m,p\geq 0\right\}$

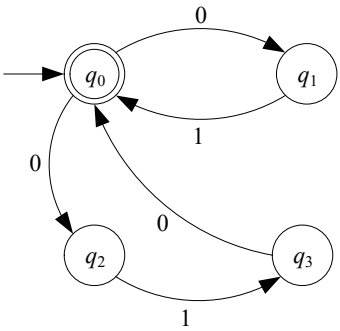
y la expresión regular sería: $a\left(b^{*}b\,|\,a^{*}b\right)a^{*}$ ó también $a\left(b^{*}\,|\,a^{*}\right)ba^{*}$

Ejemplo: Diseñe un AFND que considere el lenguaje $L=\left\{ab^{*}\cup a^{+}\right\}$ sobre el alfabeto $\sum=\{a,b\}$.

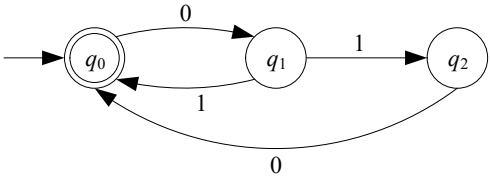
El AFND $M=\left(\{q_0,q_1,q_2\},\{a,b\},\Delta,\{q_0\},\{q_1,q_2\}\right)$ satisface $L(M)=L$.



Ejemplo: Se tiene el alfabeto $\sum=\{0,1\}$ y el lenguaje $L=\left\{(01\cup 010)^{*}\right\}$, Diseñe un AFND que acepte *L*.

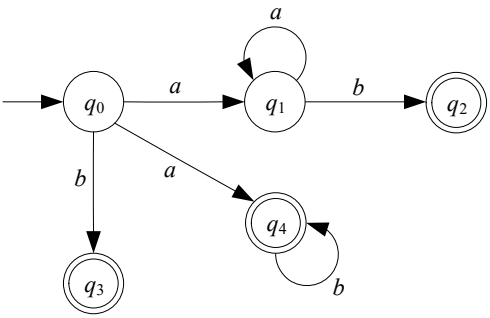


Otro AFND que satisface el mismo $L(M) = L$ y sólo tiene tres estados es:



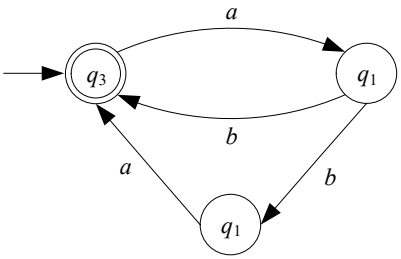
Ejemplo: Diseñe AFND y su respectiva tabla de transición sobre el alfabeto $\sum = \{a, b\}$, que reconozca el lenguaje:

a) $L = a^*b \cup ab^*$



Δ	a	b
q_0	$\{q_1, q_4\}$	$\{q_3\}$
q_1	$\{q_3\}$	$\{q_2\}$
q_2	$\varnothing$	$\varnothing$
q_3	$\varnothing$	$\varnothing$
q_3	$\varnothing$	$\{q_4\}$

b) $L = (ab \cup aba)^*$



Δ	a	b
q_0	$\{q_1\}$	$\varnothing$
q_1	$\varnothing$	$\{q_0, q_2\}$
q_2	$\{q_0\}$	$\varnothing$

2.1. Lenguaje aceptado por un AFND

Intuitivamente, un AFND acepta todas las palabras para las que puede transitar desde el estado inicial a un estado final.

Según (Díaz & Cañete, 2007) define el lenguaje de un AFND, como:

Definición. El lenguaje $L(M)$ aceptado por un AFND es el conjunto de todas las cadenas de símbolos terminales que pueden hacer que el AFND $M = (Q, \Sigma, \Delta, q_0, F)$ llegue a un estado final de aceptación y se define:

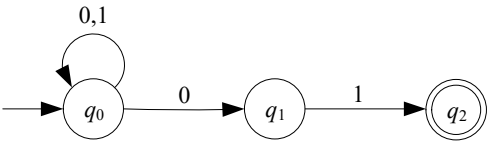
$$L(M) = \{w \in \Sigma^* / \Delta(q_0, w) \cap F \neq \varnothing\}$$

---

donde $\Delta : Q \times \Sigma^* \to \wp(Q)$ es la extensión de δ a cadenas definida:

$$\begin{cases} \hat{\Delta}(q, \lambda) = \{q\} \\ \hat{\Delta}(q, wa) = \bigcup_{p \in \hat{\Delta}(q,w)} \Delta(p, a) \end{cases} \qquad \forall q \in Q, w \in \Sigma^* \text{ y } a \in \Sigma$$

Ejemplo: Se tiene M un AFND sobre el alfabeto $\Sigma = \{0,1\}$ y cuyo diagrama de trasnsicón es:



a) Determine la tabla de transición

Δ	0	1
q_0	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\varnothing$	$\{q_2\}$
q_2	$\varnothing$	$\varnothing$

b) Determine si se aceptan las siguientes cadenas: $w_1 = 01$, $w_2 = 10$, y $w_3 = 10101$.

Para determinar si se aceptan las cadenas w_1, w_2 y w_3 se muestra la ejecución formal a partir de la unción de transición extendida $\hat{\Delta}$:

i) $\hat{\Delta}(q_0, 01)$

$$\hat{\Delta}(q_0, 01) = \delta\left(\hat{\Delta}(q_0, 0), 1\right) = \delta\left(\delta\left(\hat{\Delta}(q_0, \lambda), 0\right), 1\right)$$

Sustituuendo los estados se tiene:

$$\hat{\Delta}(q_0, 01) = \delta\left(\delta\left(\{q_0\}, 0\right), 1\right) = \delta\left(\{q_0, q_1\}, 1\right)$$

Entonce: $\delta\left(\{q_0, q_1\}, 1\right) = \delta(q_0, 1) \cup \delta(q_1, 1) = \{q_0\} \cup \{q_2\} = \{q_0, q_2\}$

Ahora aceptamos la cadena $w_1 = 01$ porque, $\{q_0, q_2\} \cap \{q_2\} \neq \varnothing$.

ii) $\hat{\Delta}(q_0, 10)$

$$\hat{\Delta}(q_0, 10) = \delta\left(\hat{\Delta}(q_0, 1), 0\right) = \delta\left(\delta\left(\hat{\Delta}(q_0, \lambda), 1\right), 0\right)$$

Sustituyendo:

$$\hat{\Delta}(q_0,01)=\delta\left(\delta\left(\{q_0\},1\right),0\right)=\delta\left(\{q_0\},0\right)=\{q_0,q_1\}$$

Ahora no aceptamos la cadena $w_2 = 10$ porque, $\{q_0,q_1\}\cap\{q_2\}=\varnothing$.

iii) $\hat{\Delta}(q_0,10101)$

$$\begin{aligned}\hat{\Delta}(q_0,10101)&=\delta\left(\hat{\Delta}(q_0,1010),1\right)=\delta\left(\delta\left(\hat{\Delta}(q_0,101),0\right),1\right)\\&=\delta\left(\delta\left(\delta\left(\hat{\Delta}(q_0,10),1\right),0\right),1\right)=\delta\left(\delta\left(\delta\left(\delta\left(\hat{\Delta}(q_0,1),0\right),1\right),0\right),1\right)\\&=\delta\left(\delta\left(\delta\left(\delta\left(\delta\left(\hat{\Delta}(q_0,\lambda),1\right),0\right),1\right),0\right),1\right)\end{aligned}$$

Sustituyendo se tiene:

$$\hat{\Delta}(q_0,10101)=\delta\left(\delta\left(\delta\left(\delta\left(\delta(q_0,1),0\right),1\right),0\right),1\right)=\delta\left(\delta\left(\delta\left(\delta\left(\{q_0\},0\right),1\right),0\right),1\right)$$

$$=\delta\left(\delta\left(\delta\left(\{q_0,q_1\},1\right),0\right),1\right),\ \delta\left(\{q_0,q_1\},1\right)=\delta\left(\{q_0\},1\right)\cup\delta\left(\{q_1\},1\right)$$

$$=\{q_0\}\cup\{q_2\}=\{q_0,q_2\}$$

$$=\delta\left(\delta\left(\{q_0,q_2\},0\right),1\right),\ \delta\left(\{q_0,q_2\},0\right)=\delta\left(\{q_0\},0\right)\cup\delta\left(\{q_1\},0\right)$$

$$=\{q_0,q_1\}\cup\varnothing=\{q_0,q_1\}$$

$$=\delta\left(\{q_0,q_1\},1\right),\ \delta\left(\{q_0,q_1\},1\right)=\delta\left(\{q_0\},1\right)\cup\delta\left(\{q_1\},1\right)$$

$$=\{q_0\}\cup\{q_2\}=\{q_0,q_2\}$$

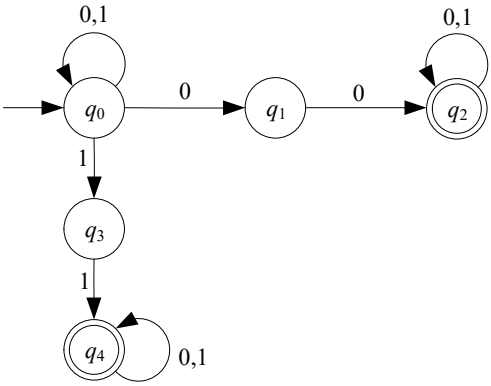
Se acerpta porque, $\{q_0,q_2\}\cap\{q_2\}\neq\varnothing$.

c) Cual es el lenguaje aceptado.

Se analizamos el diagrama de transición del AFND, se concluye que el automata reconoce acepta cadenas binarias que terminan en 01, por tanto, el lenguaje aceptado es:

$$L=\left\{(0\cup1)^n\,01/n\geq0\right\}$$

Ejemplo: Se tiene el siguiente diagrama de transición correspondiente a un AFND sobre el alfabeto $\sum = \{0,1\}$:



Determine si se acepta la siguiente cadena: $w = 01001$.

Primeramente, construimos la tabla de transición:

Δ	0	1
q_0	$\{q_0, q_1\}$	$\{q_0, q_3\}$
q_1	$\{q_2\}$	$\varnothing$
q_2	$\{q_2\}$	$\{q_2\}$
q_3	$\varnothing$	$\{q_4\}$
q_4	$\{q_4\}$	$\{q_4\}$

Ahora determinamos $\hat{\Delta}(q_0, 01001)$

$$\hat{\Delta}(q_0,01001) = \delta\left(\hat{\Delta}(q_0,0100),1\right) = \delta\left(\delta\left(\hat{\Delta}(q_0,010),0\right),1\right)$$
$$= \delta\left(\delta\left(\delta\left(\hat{\Delta}(q_0,01),0\right),0\right),1\right) = \delta\left(\delta\left(\delta\left(\delta\left(\hat{\Delta}(q_0,0),1\right),0\right),0\right),1\right)$$
$$= \delta\left(\delta\left(\delta\left(\delta\left(\delta\left(\hat{\Delta}(q_0,\lambda),0\right),1\right),0\right),0\right),1\right)$$

Sustituyendo los estados se tiene:

$$\hat{\Delta}(q_0,01001)=\delta\Big(\delta\Big(\delta\Big(\delta\Big(\delta\big(\{q_0\},0\big),1\Big),0\Big),0\Big),1\Big)=\delta\Big(\delta\Big(\delta\Big(\delta\big(\{q_0,q_1\},1\big),0\Big),0\Big),1\Big)$$

$$\delta\big(\{q_0,q_1\},1\big)=\delta\big(\{q_0\},1\big)\cup\delta\big(\{q_1\},1\big)=\{q_0,q_3\}\cup\varnothing=\{q_0,q_3\}$$

$$=\delta\Big(\delta\Big(\delta\big(\{q_0,q_3,q_4\},0\big),0\Big),1\Big),\ \delta\big(\{q_0,q_3\},0\big)=\delta\big(\{q_0\},0\big)\cup\delta\big(\{q_3\},0\big)$$

$$=\{q_0,q_1\}\cup\varnothing=\{q_0,q_1\}$$

$$=\delta\Big(\delta\big(\{q_0,q_1,q_4\},0\big),1\Big),\ \delta\big(\{q_0,q_1\},0\big)=\delta\big(\{q_0\},0\big)\cup\delta\big(\{q_1\},0\big)$$

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

$$=\delta\big(\{q_0,q_1,q_2\},1\big)=\delta\big(\{q_0\},1\big)\cup\delta\big(\{q_1\},1\big)\cup\delta\big(\{q_2\},1\big)$$

$$=\{q_0,q_3\}\cup\varnothing\cup\{q_2\}=\{q_0,q_2,q_3\}$$

Se acerpta porque, $\{q_0,q_2,q_3\}\cap\{q_2,q_4\}=\{q_2\}\neq\varnothing$.

2.2. Equivalencias entre los AFD y AFND

Los AFD y los AFND pueden resolver los mismos problemas. Por lo tanto, dado un AFND siempre es posible encontrar un AFD que sea equivalentea él. Es decir, que un AFD $M=(Q,\Sigma,\delta,q_0,F)$ puede ser considerado como un AFND $M'=(Q,\Sigma,\Delta,q_0,F)$ definiendo $\Delta(q,a)=\{\delta(q,a)\}$ para cada $q\in Q$ y cada $a\in\Sigma$. (De Castro, 2004)

Definición. Dado un AFND $M=(Q,\Sigma,\Delta,q_0,F)$ se pude construir un AFD *M'* equivalente a *M*, es decir,

$$L(M)=L(M')$$

***Observación:* AFND vs. AFD**

- Ambos permiten reconocer los mismos lenguajes.
- Unos son más comodos para algunas cosas y otros para otras.
- Si algo puede hacerse con un AFD también puede hacerse con un AFND y viceversa.

2.3. Conversión de un AFND a un AFD

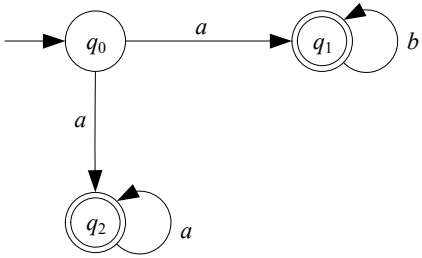
El AFD tendrá un estado por cada combinación de estados posibles en el AFND (Rubio & Rodríguez, 2019), denotamos:

$$AFND: M = (Q_N, \Sigma, \Delta_N, q_0, F_N)$$
$$AFD: M' = (Q_D, \Sigma, \delta_D, \{q_0\}, F_D)$$

donde

- $Q_D = \wp(Q_N)$: Conjunto de todos los subconjuntos de Q_N.
  Si $|Q_N| = n$, entonces $|Q_D| = 2^n$
- $F_D = \{S / (S \subseteq \wp(Q_N)) \wedge (S \cap F_N) \neq \varnothing\}$: Conjunto de subconjuntos de Q_N tales que alguno de los estados contenidos sea de aceptación. El subconjunto S de Q_N está en $F_D \Leftrightarrow (S \cap F_N) \neq \varnothing$
- $\delta_D(S,a) = \bigcup_{p \text{ en } S} \Delta_N(p,a)$: Miramos todos los estados p de S, vemos a qué estados del AFND pasan con la entrada a, y tomamos la unión de todos estos estados.

Ejemplo: Determine el AFD equivalente al AFND que se muestra en el siguiente diagrama de transición.



A partir del diagrama de transición, se tiene la siguiente tabla de transición:

Δ	a	b
q_0	$\{q_1, q_2\}$	$\varnothing$
q_1	$\varnothing$	$\{q_1\}$
q_2	$\{q_2\}$	$\varnothing$

i) Describimos el conjunto de todos los subconjuntos de Q_N.

$$Q_D = \{\varnothing, \{q_0\}, \{q_1\}, \{q_2\}, \{q_0, q_1\}, \{q_0, q_2\}, \{q_1, q_2\}, \{q_0, q_1, q_2\}\}$$

ii) Describimos el conjuto de los posibles estados de aceptación de Q_N.

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

iii) Generamos δ_D, iniciando con el estado de partida q_0 y únicamente con los nuevos estados que se van generado y dejando de lado los estados ya existentes.

$$\delta(\{q_0\}, a) = \{q_1, q_2\} \qquad \delta(\{q_0\}, b) = \varnothing$$

$$\delta(\{q_1, q_2\}, a) = \delta(\{q_1\}, a) \cup \delta(\{q_2\}, a) = \varnothing \cup \{q_2\} = \{q_2\}$$
$$\delta(\{q_1, q_2\}, b) = \delta(\{q_1\}, b) \cup \delta(\{q_2\}, b) = \{q_1\} \cup \varnothing = \{q_1\}$$

$$\delta(\{q_1\}, a) = \varnothing \qquad \delta(\{q_1\}, b) = \{q_1\}$$

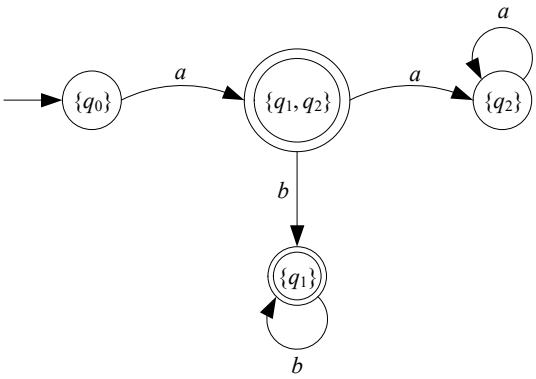
$$\delta(\{q_2\}, a) = \{q_2\} \qquad \delta(\{q_2\}, b) = \varnothing$$

Resumimos en una nueva tabla de transición definida por δ_D, donde el nuevo AFD *M*' construido a partir de *M* tiene un estado más, $\{q_1, q_2\}$.

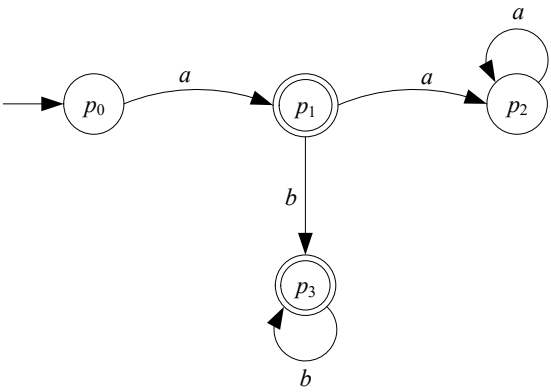
δ_D	a	b
q_0	$\{q_1, q_2\}$	$\varnothing$
q_1	$\varnothing$	$\{q_1\}$
q_2	$\{q_2\}$	$\varnothing$
$\{q_1, q_2\}$	$\{q_2\}$	$\{q_1\}$

iv) Ahora construimos el diagrama de transición de este autómata:

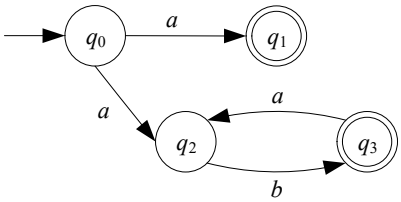
Los estados de aceptación son aquéllos en los que aparezcan q_1 ó q_2, que son los estados de aceptación del autómata original.



Para mayor simplicidad, podemos cambiar los nombres de los estados del nuevo autómata:



Ejemplo: Encontrar el equivalente AFD del AFND cuyo diagrama de transición es:



Se tiene la siguiente tabla de transición:

Δ	a	b
q_0	$\{q_1, q_2\}$	$\varnothing$
q_1	$\varnothing$	$\varnothing$
q_2	$\varnothing$	$\{q_3\}$
q_3	$\{q_2\}$	$\varnothing$

Generamos δ_D, iniciando con el estado de partida q_0:

$$\delta(\{q_0\}, a) = \{q_1, q_2\} \qquad \delta(\{q_0\}, b) = \varnothing$$

$$\delta(\{q_1, q_2\}, a) = \delta(\{q_1\}, a) \cup \delta(\{q_2\}, a) = \varnothing \cup \varnothing = \varnothing$$
$$\delta(\{q_1, q_2\}, b) = \delta(\{q_1\}, b) \cup \delta(\{q_2\}, b) = \varnothing \cup \{q_3\} = \{q_3\}$$

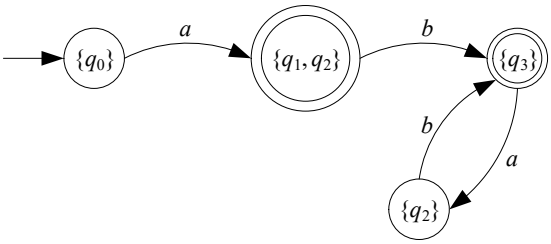
$$\delta(\{q_3\}, a) = \{q_2\} \qquad \delta(\{q_3\}, b) = \varnothing$$

$$\delta(\{q_2\}, a) = \varnothing \qquad \delta(\{q_2\}, b) = \{q_3\}$$

Se tiene la tabla de transición definida por δ_D:

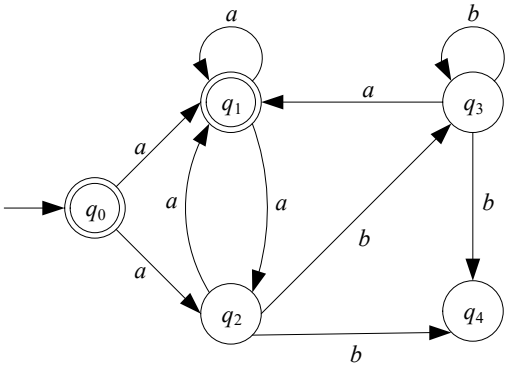
δ_D	a	b
q_0	$\{q_1, q_2\}$	$\varnothing$
q_2	$\varnothing$	$\{q_3\}$
q_3	$\{q_2\}$	$\varnothing$
$\{q_1, q_2\}$	$\varnothing$	$\{q_3\}$

Entonces el diagrama de transición de este autómata es:



Y el lenguaje que acepta es: $a \cup (ab)^*$

Ejemplo: Encontrar el AFD equivalente al siguiente AFND.



La tabla de transición es:

Δ	a	b
q_0	$\{q_1, q_2\}$	$\varnothing$
q_1	$\{q_1, q_2\}$	$\varnothing$
q_2	$\{q_1\}$	$\{q_3, q_4\}$
q_3	$\{q_1\}$	$\{q_3, q_4\}$
q_4	$\varnothing$	$\varnothing$

Generamos δ_D :

$$\delta(\{q_0\}, a) = \{q_1, q_2\} \qquad \delta(\{q_0\}, b) = \varnothing$$

$$\delta(\{q_1, q_2\}, a) = \delta(\{q_1\}, a) \cup \delta(\{q_2\}, a) = \{q_1, q_2\} \cup \{q_1\} = \{q_1, q_2\}$$
$$\delta(\{q_1, q_2\}, b) = \delta(\{q_1\}, b) \cup \delta(\{q_2\}, b) = \varnothing \cup \{q_3, q_4\} = \{q_3, q_4\}$$

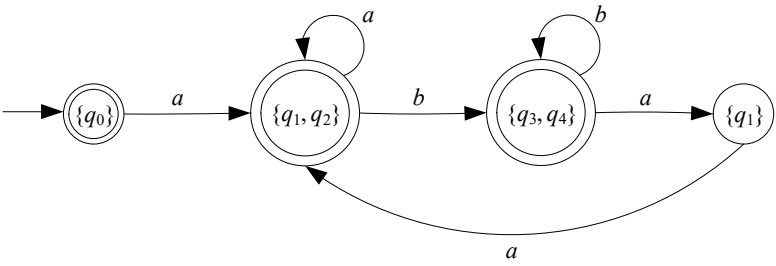
$$\delta(\{q_3, q_4\}, a) = \delta(\{q_3\}, a) \cup \delta(\{q_4\}, a) = \{q_1\} \cup \varnothing = \{q_1\}$$
$$\delta(\{q_3, q_4\}, b) = \delta(\{q_3\}, b) \cup \delta(\{q_4\}, b) = \{q_3, q_4\} \cup \varnothing = \{q_3, q_4\}$$

$$\delta(\{q_1\}, a) = \{q_1, q_2\} \qquad \delta(\{q_1\}, b) = \varnothing$$

Se tiene la tabla de transición definida por δ_D :

δ_D	a	b
q_0	$\{q_1, q_2\}$	$\varnothing$
$\{q_1, q_2\}$	$\{q_1, q_2\}$	$\{q_3, q_4\}$
$\{q_3, q_4\}$	$\{q_1\}$	$\{q_3, q_4\}$
$\{q_1\}$	$\{q_1, q_2\}$	$\varnothing$

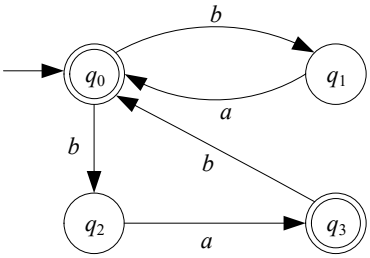
Entonces el diagrama de transición de este autómata es:



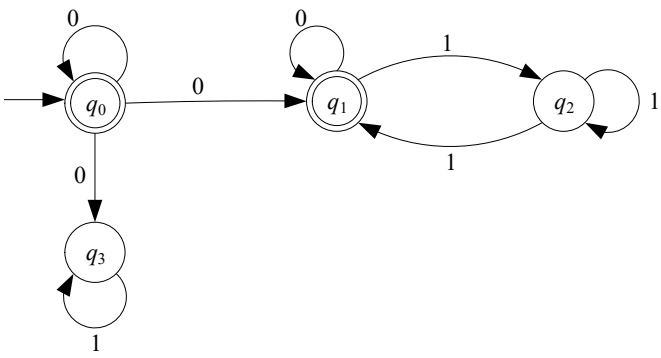
Ejercicios

Diseñar AFD equivalentes a los siguientes AFND:

a)



b)



3. AUTÓMATAS CON TRANSICIONES λ (AFND- λ)

Se puede ampliar la definición de autómata finito no determinista (AFND) para incluir transiciones de un estado a otro que no dependan de ninguna entrada. Tales transiciones se llaman transiciones-λ porque al realizarse no consumen ningún símbolo de la entrada. Se puede definir un AFND con transiciones-épsilon (AFND-λ) como:

Definición. Un autómata finito no determinista con transiciones λ (AFND-λ) es una quíntupla $M = (Q, \Sigma, \Delta, q_0, F)$ en el que la función de transición esta definido como:

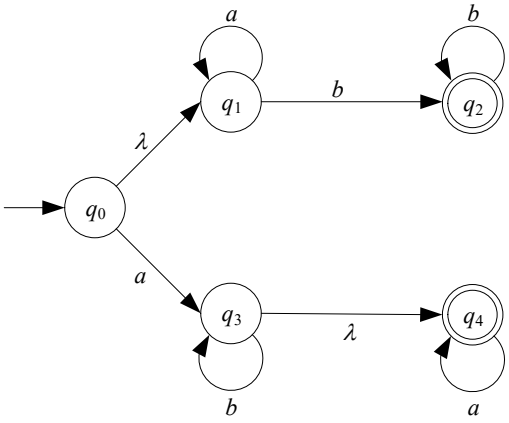
$$\Delta : Q \times (\Sigma \cup \{\lambda\}) \to \wp(Q)$$

donde, Q, Σ, q_0 y F están definidos como en el caso de un AFND.

Observación:

- Como ya sabemos, λ representa una cadena vacía, es decir $|w| = 0$.
- La transición $\Delta(q, \lambda) = \{p_i, \cdots p_{i_n}\}$ llamada también transición λ, transición nula o transición espontánea.

Ejemplo: Se tiene el siguiente diagrama de transición:



Se tiene la siguiente tabla de transición:

Δ	λ	a	b
q_0	$\{q_1\}$	$\{q_3\}$	$\varnothing$
q_1	$\varnothing$	$\{q_1\}$	$\{q_2\}$
q_2	$\varnothing$	$\varnothing$	$\{q_2\}$
q_3	$\{q_4\}$	$\varnothing$	$\{q_3\}$
q_4	$\varnothing$	$\{q_4\}$	$\varnothing$

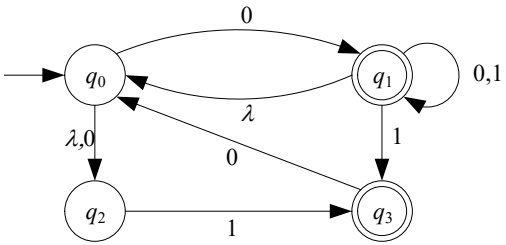
Algunoss ejemplos de cadenas aceptadas: *aab*, *abaa*, *abbaa*, …

Los AFND-λ permiten aún más libertad en el diseño de autómatas, especialmente cuando hay numerosas concatenaciones.

Ejemplo: Para la maquina $M_1 = \left(\{q_0, q_1, q_2, q_3\}, \{0,1\}, \Delta, \{q_0\}, \{q_1, q_3\}\right)$ y Δ se define por la siguiente tabla:

Δ	λ	0	1
q_0	$\{q_2\}$	$\{q_1, q_2\}$	$\varnothing$
q_1	$\{q_0\}$	$\{q_1\}$	$\{q_1, q_3\}$
q_2	$\varnothing$	$\varnothing$	$\{q_3\}$
q_3	$\varnothing$	$\{q_0\}$	$\varnothing$

A partir de la tabla se construye el siguiente diagrama de transición para el AFND-λ siguiente:



A continuación, se puede definir la λ-Clausura.

Definición. Se define la ***λ*-Clausura** (λ-cl) como un conjunto de todos los estados a los que se puede acceder desde un estado $q \in Q$ sin consumir símbolos de la entrada.

$\lambda - cl = \{p \in Q / p$ es accesible desde q sin consumir símbolos de entrada$\}$

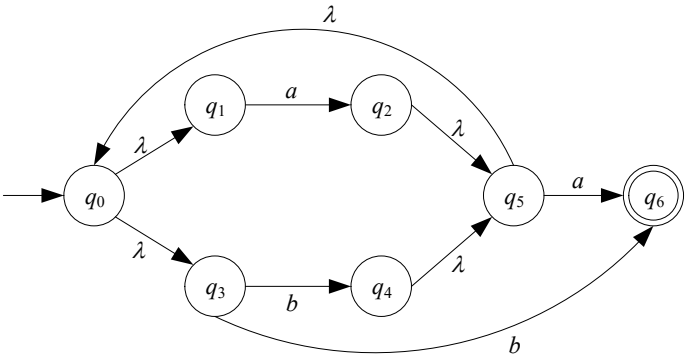
Ejemplo: En el ejemplo anterior del diagrama de transición de M_1.

$$\lambda - cl(q_0) = \{q_0\} \cup \{q_2\} = \{q_0, q_2\}$$
$$\lambda - cl(q_1) = \{q_1\} \cup \{q_0, q_2\} = \{q_0, q_1, q_2\}$$
$$\lambda - cl(q_2) = \{q_2\} \cup \varnothing$$
$$\lambda - cl(q_3) = \{q_3\} \cup \varnothing$$

Observación.

- $q \in \lambda - \text{cl}(q)$, $\forall q$. Es decir, la λ-cl nuenca puede ser vacía, contendrá al menos un estado sobre el que se aplica la λ-cl.
- $\lambda - \text{cl}(\{\varnothing\}) = \{\varnothing\}$
- Como sse vió con los AFND para cada subconjunto A de estados escribimos: $\Delta(A, a) = \bigcup_{r \in A} \Delta(r, a)$

Ejemplo: Se tiene el diagrama de transición de M_2.



Determine los λ-cl para los estados q_0, q_2 y $\{q_1, q_4\}$.

$$\lambda - cl(q_0) = \{q_0\} \cup \{q_1, q_3\} = \{q_0, q_1, q_3\}$$
$$\lambda - cl(q_2) = \{q_2\} \cup \{q_0, q_1, q_3, q_5\} = \{q_0, q_1, q_2, q_3, q_5\}$$
$$\lambda - cl(\{q_1, q_4\}) = \lambda - cl(q_1) \cup \lambda - cl(q_4) = \{q_1\} \cup \{q_0, q_1, q_3, q_4, q_5\} = \{q_0, q_1, q_3, q_4, q_5\}$$

3.1. Extensión de la función de transición

Se define una función que describe qué estados son alcanzables desde un estado q si a continuación en vez de entrar un solo símbolo, entrara una palabra $w \in \sum^*$.

Definición. La función $\Delta : Q \times \sum^* \to \wp(Q)$, y definimoscomo siempre $\hat{\Delta}(q, w)$ por recurrencia sobre la longitud de la palabra w.

- Si $|w| = 0$ entonces $w = \lambda$ y definimos $\forall q \in Q$, $\delta(q, \lambda) = \lambda - \text{cl}(q)$ (desde q, sin entrada de ningún símbolo, se alcanza los estados de $\lambda - \text{cl}(q)$).
- Supongamos definido $\hat{\Delta}(q, x)$ para cada x tal que $|x| \leq n$.
- Sea $w \in \sum^*$ tal que $|w| \leq n+1$, entonces w se puede escribir $w = xa$ con $|x| = n$.

Ahora definimos $\forall q \in Q$,

$$\hat{\Delta}(q, xa) = \lambda - \text{cl}\left(\delta\left(\hat{\Delta}(q, x), a\right)\right)$$

3.2. Lenguaje aceptado por un AFND-λ

Definición. Sea $w \in \sum^*$ decimos que w es una **palabra aceptada** por el AFND-λ si desde el estado inicial y leyendo la palabra completa alguno de los estados ue se alcanzan es final, es decir si

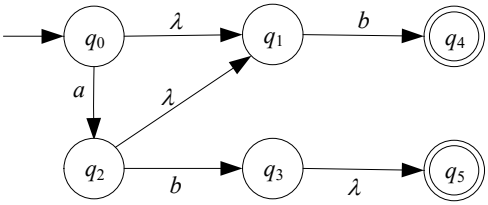
$$\hat{\Delta}(q_0, w) \cap F \neq \varnothing$$

Es importante indicar que, los AFND son un caso particular de los AFND-λ donde toadas las imágenes de $\delta(q,\lambda)$ son el conjunto vacio.

Definición. Es el conjunto de todas las palabras aceptadas, es decir,

$$L(M)=\left\{w\in\Sigma^{*}\,/\,\hat{\Delta}(q_0,w)\cap F\neq\varnothing\right\}$$

Ejemplo: Sea *M* un AFND-λ sobre el alfabeto $\Sigma=\{a,b\}$ y cuyo diagrama de transición es:



Determine si se aceptan las siguientes cadenas: $w_1 = a$, y $w_2 = ab$.

i) $\hat{\Delta}(q_0,a)=\lambda-\mathrm{cl}\left(\delta\left(\lambda-\mathrm{cl}(q_0),a\right)\right)$

Primeramente, hallamos $\lambda-\mathrm{cl}\left(\{q_0\}\right)=\{q_0,q_1\}$

Luego, $\delta\left(\{q_0,q_1\},a\right)=\delta\left(\{q_0\},a\right)\cup\delta\left(\{q_1\},a\right)=\{q_2\}\cup\varnothing=\{q_2\}$

Finalmente, $\lambda-cl\left(\{q_2\}\right)=\{q_1,q_2\}\cap\{q_4,q_5\}=\varnothing$, no se acepta la cadena

ii) $\hat{\Delta}(q_0,ab)$

$\lambda-\mathrm{cl}\left(\{q_0\}\right)=\{q_0,q_1\}$

$\delta\left(\{q_0,q_1\},a\right)=\delta\left(\{q_0\},a\right)\cup\delta\left(\{q_1\},a\right)=\{q_2\}\cup\varnothing=\{q_2\}$

$\lambda-\mathrm{cl}(q_2)=\{q_1,q_2\}$

$$\delta(\{q_1,q_2\},b)=\delta(\{q_1\},b)\cup\delta(\{q_2\},b)=\{q_4\}\cup\{q_3\}=\{q_3,q_4\}$$

$$\lambda-\mathrm{cl}(\{q_3,q_4\})=\lambda-\mathrm{cl}(\{q_3\})\cup\lambda-\mathrm{cl}(\{q_4\})=\{q_3,q_5\}\cup\{q_4\}=\{q_3,q_4,q_5\}$$

$\therefore\ \{q_3,q_4,q_5\}\cap\{q_4,q_5\}\neq\varnothing$, se acepta la cadena

4. EQUIVALENCIAS ENTRE LOS AFND-λ, AFND Y AFD

Primero observamos que cualquier AFND es obviamente también un AFND-λ y luego podemos construir a partir de un AFND un AFD equivalente.

Definición. Dado un AFND-λ $M=(Q,\Sigma,\Delta,q_0,F)$ se pude construir un AFND $M'=(Q',\Sigma,\Delta',q'_0,F')$ equivalente a *M*, es decir,

$$L(M)=L(M')$$

Para construir M' a partir de *M* se requiere la noción de **λ-Clausura de un estado.** Para un estado $q\in Q$, la λ-Clausura de *q,* denotada λ-cl(*q*), es el conjunto de estados de *M* a los que se puede llegar desde *q* por una, dos o más transiciones λ.

4.1. Conversión de un AFND-λ a un AFND

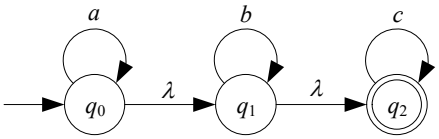
Sea $M=(Q,\Sigma,\Delta,q_0,F)$ un AFND-λ. Un AFND equivalente es el autómata $M'=(Q',\Sigma,\Delta',q'_0,F')$ donde:

- $Q'=Q$
- $\Delta'(q,a)=\bigcup_{r\in\lambda-\mathrm{cl}(q)}\lambda-\mathrm{cl}(\delta(r,a))$
- $q'_0=q_0$
- F' se define como:

$$F'=\begin{cases}F & si\ \ F\cap\lambda-\mathrm{cl}(q_0)=\varnothing\\ F\cup q_0 & si\ \ F\cap\lambda-\mathrm{cl}(q_0)\neq\varnothing\end{cases}$$

es decir, añadimos q_0 como estado final, si algún estado final del AFND-λ pertenece a la λ-Clausura del estado inicial. (Formella, 2014)

Ejemplo: Determine el AFND equivalente al AFND-λ cuyo diagrama de transición es el siguiente:



i) Hallamos los λ-Clausura para cada estado.

$$\lambda - cl(q_0) = \{q_0\} \cup \{q_1, q_2\} = \{q_0, q_1, q_2\}$$
$$\lambda - cl(q_1) = \{q_1\} \cup \{q_2\} = \{q_1, q_2\}$$
$$\lambda - cl(q_2) = \{q_2\} \cup \varnothing = \{q_2\}$$

ii) Tabla de transición AFND-λ.

Δ	λ	a	b	c
q_0	$\{q_0\} \cup \{q_1, q_2\}$	$\{q_0\}$	$\varnothing$	$\varnothing$
q_1	$\{q_1\} \cup \{q_2\}$	$\varnothing$	$\{q_1\}$	$\varnothing$
q_2	$\{q_2\} \cup \varnothing$	$\varnothing$	$\varnothing$	$\{q_2\}$

iii) Establecemos las transiciones Δ'.

$$\Delta'(q_0, a) = \lambda - \text{cl}\left(\delta\left(\lambda - \text{cl}(q_0), a\right)\right)$$
$$= \lambda - \text{cl}\left(\delta\left(\{q_0, q_1, q_2\}, a\right)\right) = \lambda - \text{cl}\left(\{q_0\}\right) = \{q_0, q_1, q_2\}$$
$$\Delta'(q_0, b) = \lambda - \text{cl}\left(\delta\left(\lambda - \text{cl}(q_0), b\right)\right)$$
$$= \lambda - \text{cl}\left(\delta\left(\{q_0, q_1, q_2\}, b\right)\right) = \lambda - \text{cl}\left(\{q_1\}\right) = \{q_1, q_2\}$$

$$\Delta'(q_0,c) = \lambda - \mathrm{cl}\big(\delta\big(\lambda - \mathrm{cl}(q_0),c\big)\big)$$
$$= \lambda - \mathrm{cl}\big(\delta\big(\{q_0,q_1,q_2\},c\big)\big) = \lambda - \mathrm{cl}\big(\{q_2\}\big) = \{q_2\}$$

$$\Delta'(q_1,a) = \lambda - \mathrm{cl}\big(\delta\big(\lambda - \mathrm{cl}(q_1),a\big)\big)$$
$$= \lambda - \mathrm{cl}\big(\delta\big(\{q_1,q_2\},a\big)\big) = \lambda - \mathrm{cl}(\varnothing) = \varnothing$$
$$\Delta'(q_1,b) = \lambda - \mathrm{cl}\big(\delta\big(\lambda - \mathrm{cl}(q_1),b\big)\big)$$
$$= \lambda - \mathrm{cl}\big(\delta\big(\{q_1,q_2\},b\big)\big) = \lambda - \mathrm{cl}\big(\{q_1\}\big) = \{q_1,q_2\}$$
$$\Delta'(q_1,c) = \lambda - \mathrm{cl}\big(\delta\big(\lambda - \mathrm{cl}(q_1),c\big)\big)$$
$$= \lambda - \mathrm{cl}\big(\delta\big(\{q_1,q_2\},c\big)\big) = \lambda - \mathrm{cl}\big(\{q_2\}\big) = \{q_2\}$$

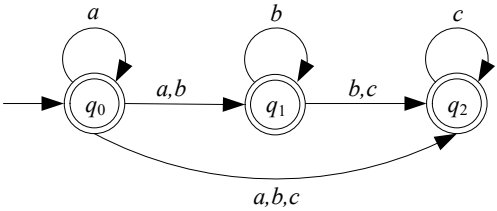
$$\Delta'(q_2,a) = \lambda - \mathrm{cl}\big(\delta\big(\lambda - \mathrm{cl}(q_2),a\big)\big)$$
$$= \lambda - \mathrm{cl}\big(\delta\big(\{q_2\},a\big)\big) = \lambda - \mathrm{cl}(\varnothing) = \varnothing$$
$$\Delta'(q_2,b) = \lambda - \mathrm{cl}\big(\delta\big(\lambda - \mathrm{cl}(q_2),b\big)\big)$$
$$= \lambda - \mathrm{cl}\big(\delta\big(\{q_2\},b\big)\big) = \lambda - \mathrm{cl}(\varnothing) = \varnothing$$
$$\Delta'(q_2,c) = \lambda - \mathrm{cl}\big(\delta\big(\lambda - \mathrm{cl}(q_2),c\big)\big)$$
$$= \lambda - \mathrm{cl}\big(\delta\big(\{q_2\},c\big)\big) = \lambda - \mathrm{cl}\big(\{q_2\}\big) = \{q_2\}$$

Resumimos en la siguiente tabla de función de distribución Δ':

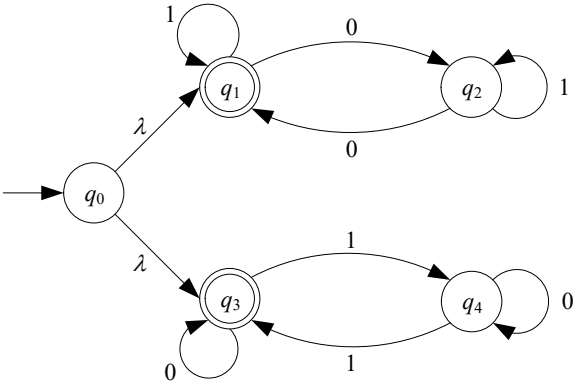
Δ'	*a*	*b*	*c*
q_0	$\{q_0,q_1,q_2\}$	$\{q_1,q_2\}$	$\{q_2\}$
q_1	$\varnothing$	$\{q_1,q_2\}$	$\{q_2\}$
q_2	$\varnothing$	$\varnothing$	$\{q_2\}$

iv) Finalmente, el autómata *M*' obtenido es el siguiente:

Los estados de aceptación serán aquellos estados que lo fueron en el AFND-λ, y además todos aquellos cuya λ-cl contenga algún estado de aceptación de AFND-λ, en este caso los tres estados serán de aceptación.



Ejemplo: Determine el AFND equivalente al siguiente AFND-λ:



Las funciones clausura para cada estado es:

$$\lambda - cl(q_0) = \{q_0\} \cup \{q_1, q_3\} = \{q_0, q_1, q_3\}$$
$$\lambda - cl(q_1) = \{q_1\}$$
$$\lambda - cl(q_2) = \{q_2\}$$
$$\lambda - cl(q_3) = \{q_3\}$$
$$\lambda - cl(q_4) = \{q_4\}$$

Tabla de transición AFND-λ.

Δ	λ	a	b
q_0	$\{q_1, q_3\}$	$\varnothing$	$\varnothing$
q_1	$\varnothing$	$\{q_2\}$	$\{q_1\}$
q_2	$\varnothing$	$\{q_1\}$	$\{q_2\}$
q_3	$\varnothing$	$\{q_3\}$	$\{q_4\}$
q_4	$\varnothing$	$\{q_4\}$	$\{q_3\}$

Las transiciones Δ' serían:

$$\begin{aligned}
\Delta'(q_0,0) &= \lambda - \mathrm{cl}\big(\delta\big(\lambda - \mathrm{cl}(q_0),0\big)\big) \\
&= \lambda - \mathrm{cl}\big(\delta\big(\{q_1,q_3\},0\big)\big) = \lambda - \mathrm{cl}\big(\{q_2,q_3\}\big) = \{q_2,q_3\} \\
\Delta'(q_0,1) &= \lambda - \mathrm{cl}\big(\delta\big(\lambda - \mathrm{cl}(q_0),1\big)\big) \\
&= \lambda - \mathrm{cl}\big(\delta\big(\{q_1,q_3\},1\big)\big) = \lambda - \mathrm{cl}\big(\{q_1,q_4\}\big) = \{q_1,q_4\} \\
\Delta'(q_1,0) &= \lambda - \mathrm{cl}\big(\delta\big(\lambda - \mathrm{cl}(q_1),0\big)\big) \\
&= \lambda - \mathrm{cl}\big(\delta\big(\{q_1\},0\big)\big) = \lambda - \mathrm{cl}\big(\{q_2\}\big) = \{q_2\} \\
\Delta'(q_1,1) &= \lambda - \mathrm{cl}\big(\delta\big(\lambda - \mathrm{cl}(q_1),1\big)\big) \\
&= \lambda - \mathrm{cl}\big(\delta\big(\{q_1\},1\big)\big) = \lambda - \mathrm{cl}\big(\{q_1\}\big) = \{q_1\}
\end{aligned}$$

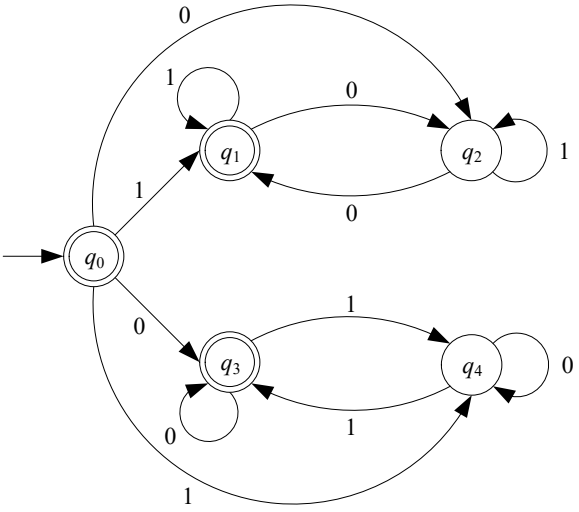
Como se observa los estados q_1, q_2, q_3 y q_4 no tienen λ-clausura, por tanto, las transiciones son iguales a la tabla anterior.

$$\begin{aligned}
\Delta'(q_2,0) &= \{q_1\} & \Delta'(q_2,1) &= \{q_2\} \\
\Delta'(q_3,0) &= \{q_3\} & \Delta'(q_3,1) &= \{q_4\} \\
\Delta'(q_4,0) &= \{q_4\} & \Delta'(q_4,1) &= \{q_3\}
\end{aligned}$$

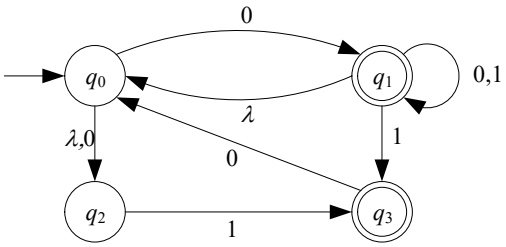
Resumiendo, se tiene la tabla de Δ':

Δ'	a	b
q_0	$\{q_2, q_3\}$	$\{q_1, q_4\}$
q_1	$\{q_2\}$	$\{q_1\}$
q_2	$\{q_1\}$	$\{q_2\}$
q_3	$\{q_3\}$	$\{q_4\}$
q_4	$\{q_4\}$	$\{q_3\}$

el autómata equivalente es:



Ejemplo: Construya un AFND equivalente al siguiente AFND-λ:



Este ejemplo se analizo anteriormente y sus funciones clausura son:

$$\lambda - cl(q_0) = \{q_0\} \cup \{q_2\} = \{q_0, q_2\}$$
$$\lambda - cl(q_1) = \{q_1\} \cup \{q_0, q_2\} = \{q_0, q_1, q_2\}$$
$$\lambda - cl(q_2) = \{q_2\} \cup \varnothing$$
$$\lambda - cl(q_3) = \{q_3\} \cup \varnothing$$

Tabla de transición AFND-λ.

Δ	λ	0	1
q_0	$\{q_2\}$	$\{q_1, q_2\}$	$\varnothing$
q_1	$\{q_0\}$	$\{q_1\}$	$\{q_1, q_3\}$
q_2	$\varnothing$	$\varnothing$	$\{q_3\}$
q_3	$\varnothing$	$\{q_0\}$	$\varnothing$

Las transiciones Δ' serían:

$$\Delta'(q_0, 0) = \lambda - \text{cl}\left(\delta\left(\lambda - \text{cl}(q_0), 0\right)\right)$$
$$= \lambda - \text{cl}\left(\delta\left(\{q_0, q_2\}, 0\right)\right) = \lambda - \text{cl}\left(\{q_1, q_2\}\right) = \{q_0, q_1, q_2\}$$
$$\Delta'(q_0, 1) = \lambda - \text{cl}\left(\delta\left(\lambda - \text{cl}(q_0), 1\right)\right)$$
$$= \lambda - \text{cl}\left(\delta\left(\{q_0, q_2\}, 1\right)\right) = \lambda - \text{cl}\left(\{q_3\}\right) = \{q_3\}$$

$$\Delta'(q_1,0) = \lambda - \mathrm{cl}\big(\delta\big(\lambda - \mathrm{cl}(q_1),0\big)\big)$$
$$= \lambda - \mathrm{cl}\big(\delta\big(\{q_0,q_1,q_2\},0\big)\big) = \lambda - \mathrm{cl}\big(\{q_1,q_2\}\big) = \{q_0,q_1,q_2\}$$
$$\Delta'(q_1,1) = \lambda - \mathrm{cl}\big(\delta\big(\lambda - \mathrm{cl}(q_1),1\big)\big)$$
$$= \lambda - \mathrm{cl}\big(\delta\big(\{q_0,q_1,q_2\},1\big)\big) = \lambda - \mathrm{cl}\big(\{q_1,q_3\}\big) = \{q_0,q_1,q_2,q_3\}$$

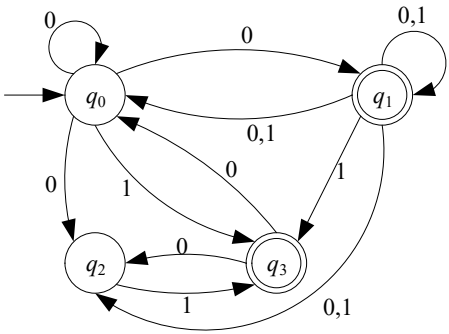
$$\Delta'(q_2,0) = \lambda - \mathrm{cl}\big(\delta\big(\lambda - \mathrm{cl}(q_2),0\big)\big)$$
$$= \lambda - \mathrm{cl}\big(\delta\big(\{q_2\},0\big)\big) = \lambda - \mathrm{cl}\big(\{\varnothing\}\big) = \varnothing$$
$$\Delta'(q_2,1) = \lambda - \mathrm{cl}\big(\delta\big(\lambda - \mathrm{cl}(q_2),1\big)\big)$$
$$= \lambda - \mathrm{cl}\big(\delta\big(\{q_2\},1\big)\big) = \lambda - \mathrm{cl}\big(\{q_3\}\big) = \{q_3\}$$

$$\Delta'(q_3,0) = \lambda - \mathrm{cl}\big(\delta\big(\lambda - \mathrm{cl}(q_3),0\big)\big)$$
$$= \lambda - \mathrm{cl}\big(\delta\big(\{q_3\},0\big)\big) = \lambda - \mathrm{cl}\big(\{q_0\}\big) = \{q_0,q_2\}$$
$$\Delta'(q_3,1) = \lambda - \mathrm{cl}\big(\delta\big(\lambda - \mathrm{cl}(q_3),1\big)\big)$$
$$= \lambda - \mathrm{cl}\big(\delta\big(\{q_3\},1\big)\big) = \lambda - \mathrm{cl}\big(\{\varnothing\}\big) = \varnothing$$

Ahora, se tiene la tabla de Δ':

Δ'	0	1
q_0	$\{q_0,q_1,q_2\}$	$\{q_3\}$
q_1	$\{q_0,q_1,q_2\}$	$\{q_0,q_1,q_2,q_3\}$
q_2	$\varnothing$	$\{q_3\}$
q_3	$\{q_0,q_2\}$	$\varnothing$

el autómata equivalente es:



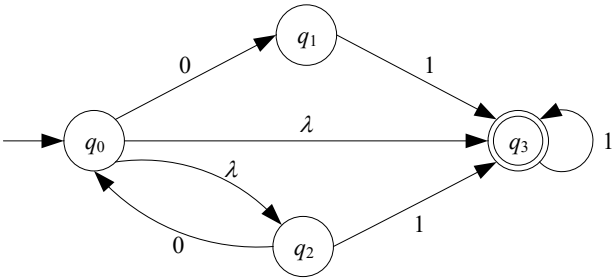
4.2. Conversión de un AFND-λ a un AFD

Tenemos un AFND-λ $M=(Q,\Sigma,\Delta,q_0,F)$ que puede tener λ-transiciones. Para cada $q\in Q$ ya definimos $\lambda-\mathrm{cl}(q)$, como el conjunto de estados alcanzables desde q mediante λ-transiciones.

Sea $M=(Q,\Sigma,\Delta,q_0,F)$ un AFND-λ. Definimos el equivalente un AFD $M'=(Q',\Sigma,\delta',q'_0,F')$ como sigue:

- $Q'=\wp(Q)$, es decir, cada estado de *M*' es un conjunto de estados de *M*.
- $\delta'(q,a)=\bigcup_{r\in\lambda-\mathrm{cl}(q)}\lambda-\mathrm{cl}(\delta(r,a)),\ \forall q\subseteq Q,\ \forall a\in\Sigma$
- $F'=\{q/q\cap F\neq\varnothing\}$
- $q'_0=\lambda-\mathrm{cl}(\{q_0\})$

Ejemplo: Determine el AFD equivalente al AFND-λ cuyo diagrama de transición es el siguiente:



i) Hallamos los λ-Clausura para cada estado.

$$\lambda - cl(q_0) = \{q_0\} \cup \{q_2, q_3\} = \{q_0, q_2, q_3\}$$
$$\lambda - cl(q_1) = \{q_1\} \cup \varnothing$$
$$\lambda - cl(q_2) = \{q_2\} \cup \varnothing$$
$$\lambda - cl(q_3) = \{q_3\} \cup \varnothing$$

ii) Tabla de transición AFND-λ.

Δ	λ	0	1
q_0	$\{q_2, q_3\}$	$\{q_1\}$	$\varnothing$
q_1	$\varnothing$	$\varnothing$	$\{q_3\}$
q_2	$\varnothing$	$\{q_0\}$	$\{q_3\}$
q_3	$\varnothing$	$\varnothing$	$\{q_3\}$

iii) Establecemos las λ-transiciones.

Sea $\lambda - cl(q_0) = \{q_0\} \cup \{q_2, q_3\} = \{q_0, q_2, q_3\} = p_0$

Ahora marcamos este nuevo estado
$\delta(p_0, 0) = \{q_0, q_1\}$ con

$$\lambda - cl\left(\{q_0, q_1\}\right) = \{q_0, q_1\} \cup \{q_2, q_3\} = \{q_0, q_1, q_2, q_3\} = p_1$$

$$\delta(p_0, 1) = \{q_3\} \text{ con } \lambda - cl\left(\{q_3\}\right) = \{q_3\} \cup \varnothing = p_2$$

$$\delta(p_1, 0) = \{q_0, q_1\} \text{ con } \lambda - cl\left(\{q_0, q_1\}\right) = \{q_0, q_1\} \cup \{q_2, q_3\}$$

$$\delta(p_1, 1) = \{q_3\} \text{ con } \lambda - cl\left(\{q_3\}\right) = \{q_3\} \cup \varnothing$$

$$\delta(p_2, 0) = \varnothing \text{ con } \lambda - cl(\varnothing) = \varnothing$$

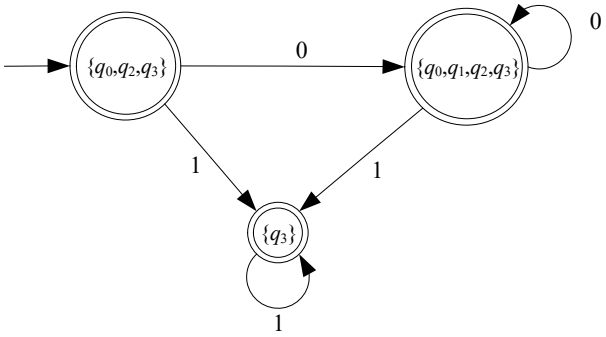
$$\delta(p_2, 1) = \{q_3\} \text{ con } \lambda - cl\left(\{q_3\}\right) = \{q_3\} \cup \varnothing$$

Se ha terminado, ya que todos los estados están identificados.
Los estados p_0, p_1 y p_2 serán estados de aceptación ya que continen al estado q_3, el cual era el estado de aceptación del AFND-λ.

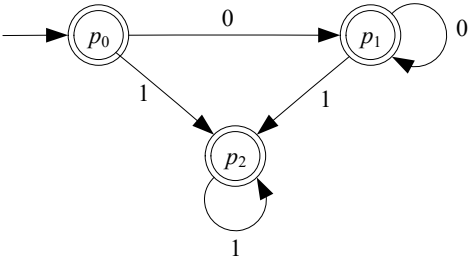
Resumimos en la siguiente tabla de función de distribución Δ':

δ'	0	1
$p_0: \{q_0\} \cup \{q_2, q_3\}$	$\{q_0, q_1\} \cup \{q_2, q_3\}$	$\{q_3\} \cup \varnothing$
$p_1: \{q_0, q_1\} \cup \{q_2, q_3\}$	$\{q_0, q_1\} \cup \{q_2, q_3\}$	$\{q_3\} \cup \varnothing$
$p_2: \{q_3\}$	$\varnothing$	$\{q_3\} \cup \varnothing$

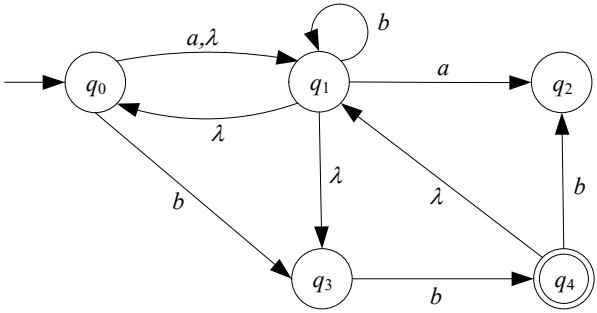
iv) Constriuimos el diagrama de transiciones equivalente.



Tambien puede ser:



Ejemplo: Construya el AFD equivalente al AFND-λ siguiente:



Determinamos los λ-Clausura.

$$\lambda - cl(q_0) = \{q_0\} \cup \{q_1, q_3\} = \{q_0, q_1, q_3\}$$
$$\lambda - cl(q_1) = \{q_1\} \cup \{q_0, q_1, q_3\} = \{q_0, q_1, q_3\}$$
$$\lambda - cl(q_2) = \{q_2\} \cup \varnothing = \{q_2\}$$
$$\lambda - cl(q_3) = \{q_3\} \cup \varnothing = \{q_3\}$$
$$\lambda - cl(q_4) = \{q_4\} \cup \{q_0, q_1, q_3\} = \{q_0, q_1, q_3, q_4\}$$

La tabla de transición AFND-λ es:

Δ	λ	a	b
q_0	$\{q_1\}$	$\{q_1\}$	$\{q_3\}$
q_1	$\{q_3\}$	$\{q_2\}$	$\{q_1\}$
q_2	$\varnothing$	$\varnothing$	$\varnothing$
q_3	$\varnothing$	$\varnothing$	$\{q_4\}$
q_4	$\{q_1\}$	$\varnothing$	$\{q_2\}$

Establecemos las λ-transiciones.

Sea $\lambda - cl(q_0) = \{q_0, q_1, q_3\} = p_0$

Ahora marcamos este nuevo estado

$\delta(p_0, a) = \{q_1, q_2\}$ con

$\lambda - cl(\{q_1, q_2\}) = \{q_0, q_1, q_3\} \cup \{q_2\} = \{q_0, q_1, q_2, q_3\} = p_1$

$\delta(p_0, b) = \{q_1, q_3, q_4\}$ con

$\lambda - cl(\{q_1, q_3, q_4\}) = \{q_0, q_1, q_3\} \cup \{q_3\} \cup \{q_0, q_1, q_3, q_4\} = \{q_0, q_1, q_3, q_4\} = p_2$

$\delta(p_1, a) = \{q_1, q_2\}$ con

$\lambda - cl(\{q_1, q_2\}) = \{q_0, q_1, q_3\} \cup \{q_2\} = \{q_0, q_1, q_2, q_3\}$

$\delta(p_1, b) = \{q_1, q_3, q_4\}$ con

$\lambda - cl(\{q_1, q_3, q_4\}) = \{q_0, q_1, q_3\} \cup \{q_3\} \cup \{q_0, q_1, q_3, q_4\} = \{q_0, q_1, q_3, q_4\}$

$\delta(p_2, a) = \{q_1, q_2\}$ con

$\lambda - cl(\{q_1, q_2\}) = \{q_0, q_1, q_3\} \cup \{q_2\} = \{q_0, q_1, q_2, q_3\}$

$\delta(p_2, b) = \{q_1, q_2, q_3, q_4\}$ con

$\lambda - cl(\{q_1, q_2, q_3, q_4\}) = \{q_0, q_1, q_3\} \cup \{q_2\} \cup \{q_3\} \cup \{q_0, q_1, q_3, q_4\}$

$= \{q_0, q_1, q_2, q_3, q_4\} = p_3$

$\delta(p_3, a) = \{q_1, q_2\}$ con

$\lambda - cl(\{q_1, q_2\}) = \{q_0, q_1, q_3\} \cup \{q_2\} = \{q_0, q_1, q_2, q_3\}$

$\delta(p_3, b) = \{q_1, q_2, q_3, q_4\}$ con

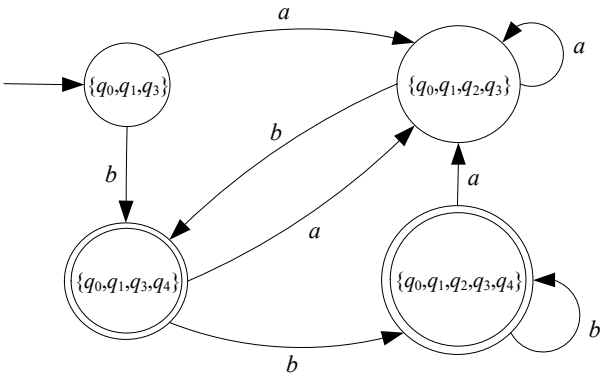
$\lambda - cl(\{q_1, q_2, q_3, q_4\}) = \{q_0, q_1, q_3\} \cup \{q_2\} \cup \{q_3\} \cup \{q_0, q_1, q_3, q_4\}$

$= \{q_0, q_1, q_2, q_3, q_4\}$

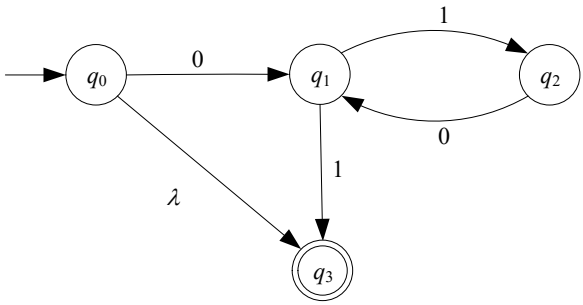
Los estados p_2 y p_3 serán estados de aceptación ya que continen al estado q_4, el cual era el estado de aceptación del AFND-λ.

Obtenemos la tabla de función de distribución δ' y el diagrama:

δ'	a	b
$p_0 : \{q_0, q_1, q_3\}$	$\{q_0, q_1, q_2, q_3\}$	$\{q_0, q_1, q_3, q_4\}$
$p_1 : \{q_0, q_1, q_2, q_3\}$	$\{q_0, q_1, q_2, q_3\}$	$\{q_0, q_1, q_3, q_4\}$
$p_2 : \{q_0, q_1, q_3, q_4\}$	$\{q_0, q_1, q_2, q_3\}$	$\{q_0, q_1, q_2, q_3, q_4\}$
$p_3 : \{q_0, q_1, q_2, q_3, q_4\}$	$\{q_0, q_1, q_2, q_3\}$	$\{q_0, q_1, q_2, q_3, q_4\}$



Ejemplo: Determine el AFD equivalente del siguiente AFND-λ:



Calculamos

$$\lambda - cl(q_0) = \{q_0\} \cup \{q_3\} = \{q_0, q_3\}$$
$$\lambda - cl(q_1) = \{q_1\} \cup \varnothing = \{q_1\}$$
$$\lambda - cl(q_2) = \{q_2\} \cup \varnothing = \{q_2\}$$
$$\lambda - cl(q_3) = \{q_3\} \cup \varnothing = \{q_3\}$$

Entonces:

Δ	λ	0	1
q_0	$\{q_0, q_3\}$	$\{q_1\}$	$\varnothing$
q_1	$\varnothing$	$\varnothing$	$\{q_2, q_3\}$
q_2	$\varnothing$	$\{q_1\}$	$\varnothing$
q_3	$\varnothing$	$\varnothing$	$\varnothing$

las λ-transiciones son:

$$\lambda - cl(q_0) = \{q_0, q_3\} = p_0$$
$$\delta(p_0, 0) = \{q_1\} \text{ con } \lambda - cl(\{q_1\}) = \{q_1\} = p_1$$
$$\delta(p_0, 1) = \varnothing \text{ con } \lambda - cl(\{\varnothing\}) = \varnothing$$

$$\delta(p_1, 0) = \varnothing \text{ con } \lambda - cl(\{\varnothing\}) = \varnothing$$
$$\delta(p_1, 1) = \{q_2, q_3\} \text{ con } \lambda - cl(\{q_2, q_3\}) = \{q_2\} \cup \{q_3\} = \{q_2, q_3\} = p_2$$

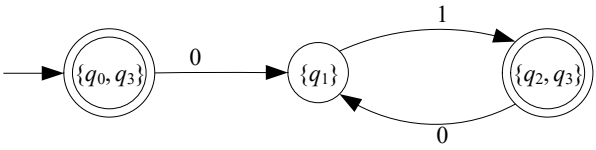
$\delta(p_2, 0) = \{q_1\}$ con $\lambda - cl(\{q_1\}) = \{q_1\}$

$\delta(p_2, 1) = \varnothing$ con $\lambda - cl(\{\varnothing\}) = \varnothing$

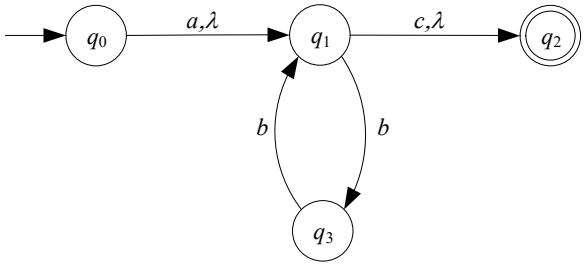
Los estados p_0 y p_2 serán estados de aceptación.

Por tanto, la tabla de función de distribución δ' y el diagrama son:

δ'	0	1
$p_0 : \{q_0, q_3\}$	$\{q_1\}$	$\varnothing$
$p_1 : \{q_1\}$	$\varnothing$	$\{q_2, q_3\}$
$p_2 : \{q_2, q_3\}$	$\{q_1\}$	$\varnothing$



Ejemplo: Determine el AFD equivalente del siguiente AFND-λ:



Calculamos

$\lambda - cl(q_0) = \{q_0\} \cup \{q_1, q_2\} = \{q_0, q_1, q_2\}$

$\lambda - cl(q_1) = \{q_1\} \cup \{q_2\} = \{q_1, q_2\}$

$\lambda - cl(q_2) = \{q_2\} \cup \varnothing = \{q_2\}$

$\lambda - cl(q_3) = \{q_3\} \cup \varnothing = \{q_3\}$

Tenemos:

Δ	λ	a	b	c
q_0	$\{q_1\}$	$\{q_1\}$	$\varnothing$	$\varnothing$
q_1	$\{q_2\}$	$\varnothing$	$\{q_3\}$	$\{q_2\}$
q_2	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$
q_3	$\varnothing$	$\varnothing$	$\{q_1\}$	$\varnothing$

las λ-transiciones son:

$$\lambda - cl(q_0) = \{q_0, q_1, q_2\} = p_0$$
$$\delta(p_0, a) = \{q_1\} \text{ con } \lambda - cl(\{q_1\}) = \{q_1, q_2\} = p_1$$
$$\delta(p_0, b) = \{q_3\} \text{ con } \lambda - cl(\{q_3\}) = \{q_3\} = p_3$$
$$\delta(p_0, c) = \{q_2\} \text{ con } \lambda - cl(\{q_2\}) = \{q_2\} = p_2$$

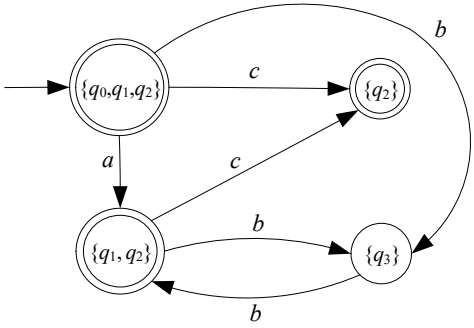
$$\delta(p_1, a) = \varnothing \text{ con } \lambda - cl(\{\varnothing\}) = \varnothing$$
$$\delta(p_1, b) = \{q_3\} \text{ con } \lambda - cl(\{q_3\}) = \{q_3\}$$
$$\delta(p_1, c) = \{q_2\} \text{ con } \lambda - cl(\{q_2\}) = \{q_2\}$$

$$\delta(p_2, a) = \varnothing \text{ con } \lambda - cl(\{\varnothing\}) = \varnothing$$
$$\delta(p_2, b) = \varnothing \text{ con } \lambda - cl(\{\varnothing\}) = \varnothing$$
$$\delta(p_2, c) = \varnothing \text{ con } \lambda - cl(\{\varnothing\}) = \varnothing$$

$$\delta(p_3, a) = \varnothing \text{ con } \lambda - cl(\{\varnothing\}) = \varnothing$$
$$\delta(p_3, b) = \{q_1\} \text{ con } \lambda - cl(\{q_1\}) = \{q_1, q_2\}$$
$$\delta(p_3, c) = \varnothing \text{ con } \lambda - cl(\{\varnothing\}) = \varnothing$$

Los estados p_0, p_1 y p_2 serán estados de aceptación. Por tanto, se tiene:

δ'	a	b	c
$p_0 : \{q_0, q_1, q_2\}$	$\{q_1, q_2\}$	$\{q_3\}$	$\{q_2\}$
$p_1 : \{q_1, q_2\}$	$\varnothing$	$\{q_3\}$	$\{q_2\}$
$p_2 : \{q_2\}$	$\varnothing$	$\varnothing$	$\varnothing$
$p_3 : \{q_3\}$	$\varnothing$	$\{q_1, q_2\}$	$\varnothing$



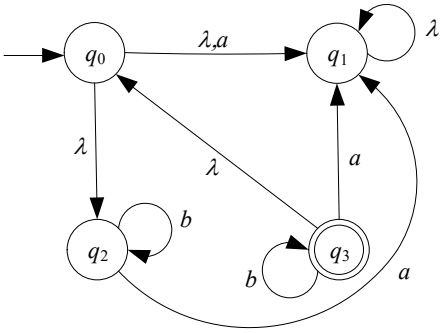
Ejercicios

1. Considere el siguiente AFND-λ, con: $q_0 = \{q_0\}$, $F = \{q_2\}$ y

Δ	λ	a	b	c
q_0	$\varnothing$	$\{q_0\}$	$\{q_1\}$	$\{q_2\}$
q_1	$\{q_0\}$	$\{q_1\}$	$\{q_2\}$	$\varnothing$
q_2	$\{q_1\}$	$\{q_2\}$	$\varnothing$	$\{q_0\}$

   a) Calcula las λ-closura para cada estado.
   b) Cuales son todas las cadenas de 3 o menos caracteres aceptadas por el automata.
   c) Convertirlo a un DFA

2. Contruya los AFD's euivalentes a los siguientes AFND-λ:



5. CONVERSIÓN DE EXPRESIONES REGULARES (ER) A AUTÓMATAS FINITOS

Dada una expresión regular existe un autómata finito capaz de reconocer el lenguaje que ésta define. Recíprocamente, dado un autómata finito, se puede expresar mediante una expresión regular el lenguaje que reconoce.

Para la transformación de una expresión regular en un autómata finito, se definirán en un principio las equivalencias entre las expresiones regulares básicas y sus autómatas finitos. Posteriormente se mostrará la construcción de Thompson que genera automáticamente un autómata finito a partir de una o varias expresiones regulares de cualquier complejidad. (Cueva, 2001)

Las expresiones regulares puden definirse como un equivalente algebraico para un autómata, es utilizado en muchos lugares como un lenguaje para describir patrones en textos que son sencillos pero muy útiles.

5.1. Conversión de ER a AFND-λ

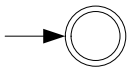
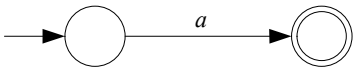
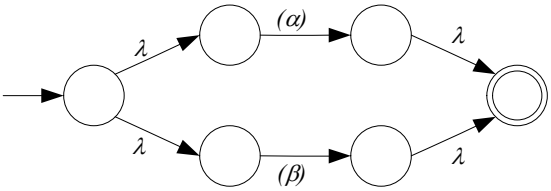
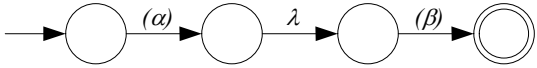
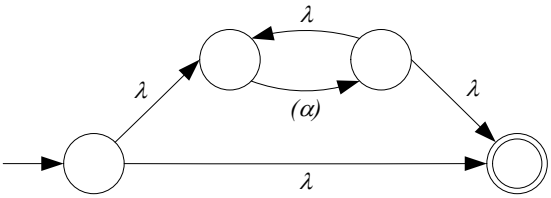
Existen distintas formas de convertir una expresión regular en un AFND-λ, de modo que $L(E) = L(M)$. La construcción de Thompson es el métos más utilizado y sencillo.

Definición. Sea una expresión regular E se recorre la misma según el orden de precedencia operacional (*, ., |) y los agrupadores para obtener un AFND-λ M definido por su diagrama de estado según los siguientes reglas:

***Entrada*:** Una expresión regular E en un alabeto Σ.

***Salida*:** un AFND-λ que acepte $L(E)$

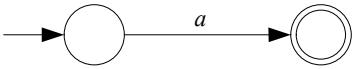
Sean α y β expresiones regulares:

Expresión regular (ER)	*Autómata finito (AF)*
$\varnothing$	$\forall a \in \sum : \delta(q_0, a) = \varnothing\ ,\ F = \varnothing$ 
λ	$\forall a \in \sum : \delta(q_0, a) = \varnothing\ ,\ F = \{q_0\} = Q$ 
a	$\delta(q_0, a) = \{q_1\}\ ,\ F = \{q_1\}$ 
$\alpha \mid \beta$	
$\alpha.\beta$	
α^*	

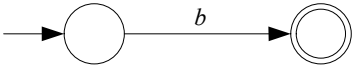
Ejemplo: Construya el AFND-λ que representa la ER: $\alpha = a^*b$

Tenemos los siguientes pasos:

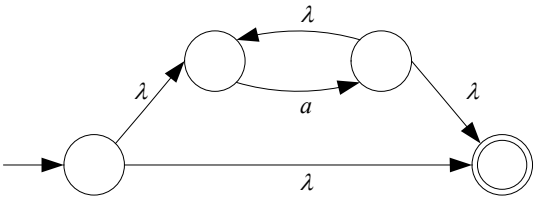
i) Para *a*:



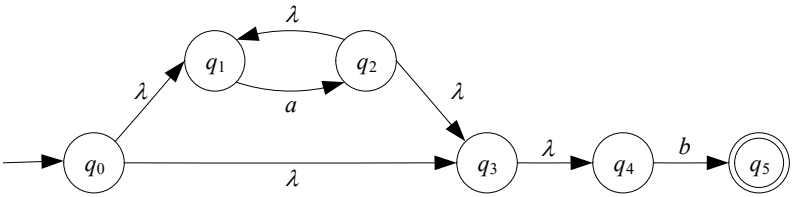
ii) Para *b*:



iii) Para *a**:



iv) Finalmente, para *a*b* se tiene:



y numeramos los estados de q_0 a q_5.

Formalizando el autómata se tiene: AFND-λ $M = (Q, \Sigma, \Delta, q_0, F)$, donde:

$Q = \{q_0, q_1, q_2, q_3, q_4, q_5\}$

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

$q_0 = \{q_0\}$

$F = \{q_5\}$

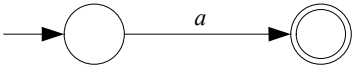
y la función Δ esta dada por la siguiente tabla:

Δ	λ	a	b
q_0	$\{q_1, q_3\}$	$\varnothing$	$\varnothing$
q_1	$\varnothing$	$\{q_2\}$	$\varnothing$
q_2	$\{q_1, q_3\}$	$\varnothing$	$\varnothing$
q_3	$\{q_4\}$	$\varnothing$	$\varnothing$
q_4	$\varnothing$	$\varnothing$	$\{q_5\}$
q_5	$\varnothing$	$\varnothing$	$\varnothing$

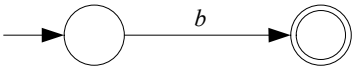
Ejemplo: Construya el AFND-λ de la expresión regular: $(a \mid b)^* a$

Tenemos los siguientes pasos:

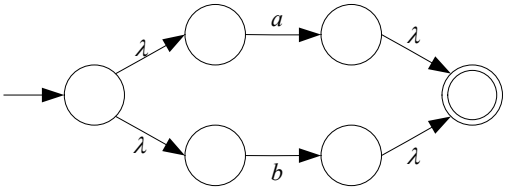
Para *a*:



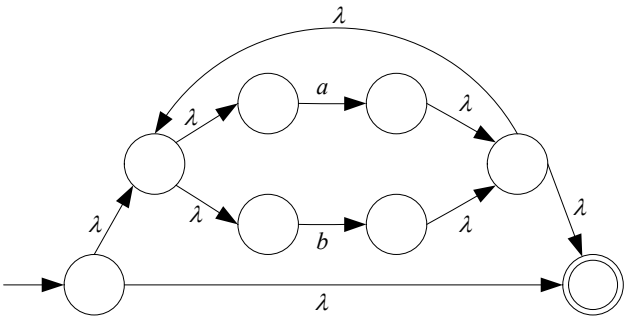
Para *b*:



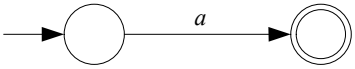
Ahora se puede combinar *a* y *b* utilizando la regla de la unión para obtener *a*|*b*.



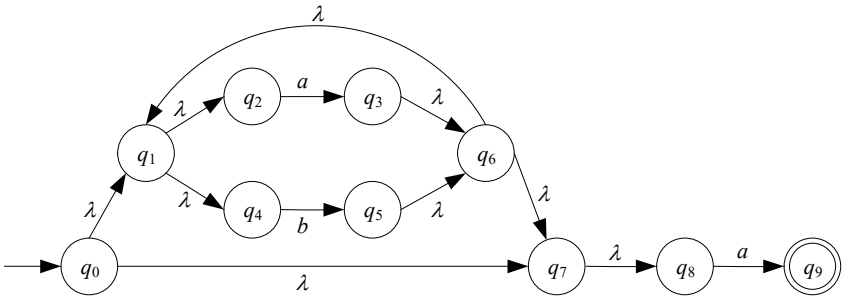
Para (*a*|*b*)* es:



Finalmente agregamos la segunda *a*:



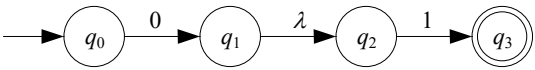
nos queda de la siguiente forma:



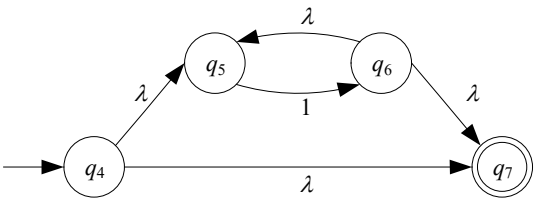
Ejemplo: Sea la expresión regular: $01|1^*$ se pide construir el AFND-λ.

Tambien podemos construir numerando los estados desde el inicio como se muestra a continuación:

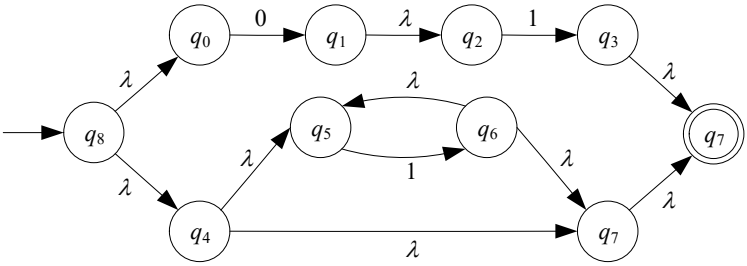
Para 01:



1*:

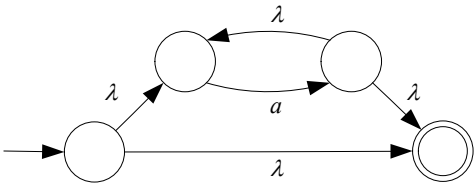


Se tiene 01|1*:

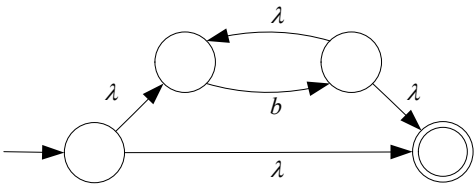


Ejemplo: Sea tiene $\left(a^* \mid b^*\right)c^*$ se pide construir el AFND-λ.

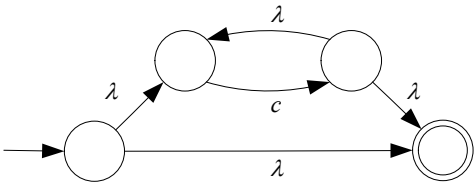
Para *a**:



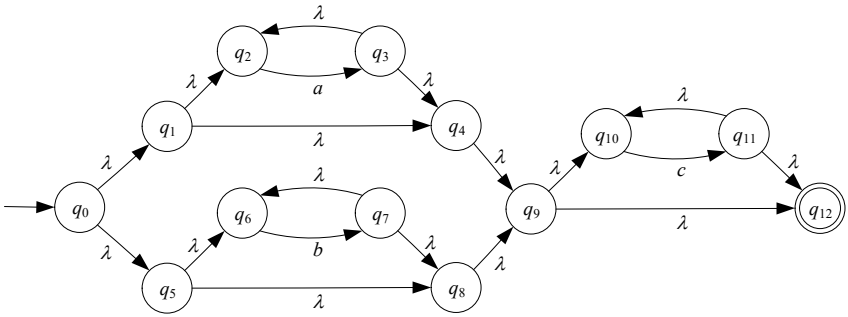
*b**:



c*:

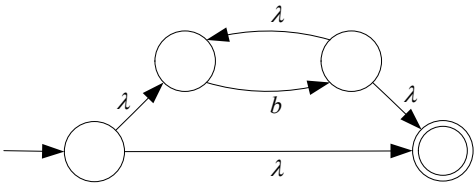


Se tiene $\left(a^* \mid b^*\right)c^*$:

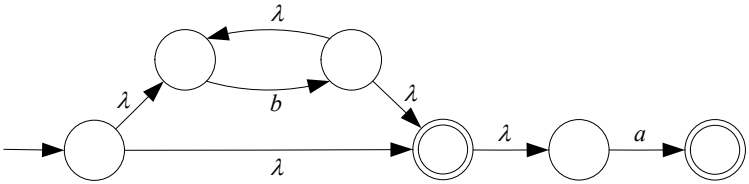


Ejemplo: Sea tiene $(b \mid (b^* a)^*)$ se pide construir el AFND-λ.

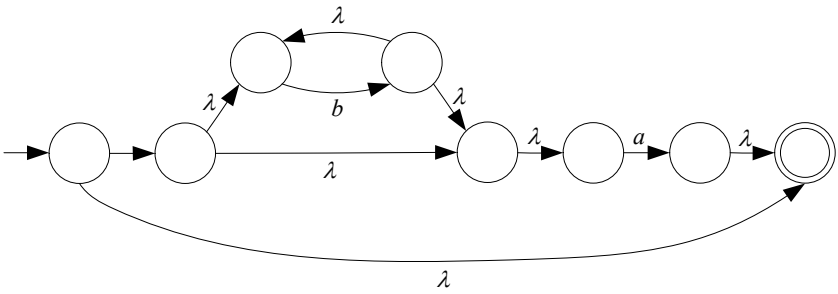
Para b*:



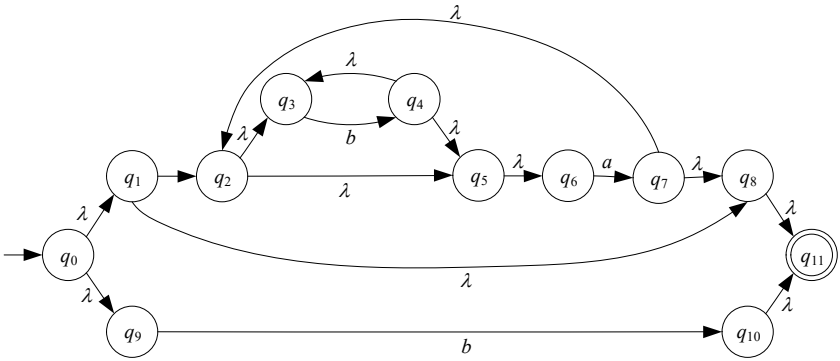
b*a:



(b*a)*:



Finalmente (*b* | (*b***a*)*):



Ejercicio Propuesto

1. Construya el diagrama de transición de los AFND-λ para las siguientes expresiones regulares utilizando las reglas de Thompson:
   a) (ab)*
   b) $(b|bc)^{+}$
   c) (ab|a)*
   d) (a*b*)*
   e) (c(a*b)*)*
   f) $(a^{*}|b^{+})^{+}$
   g) abcd|e

5.2. Conversión de AFD a ER

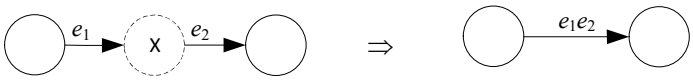
Como ya se vio, dada una expresión regular existe un autómata finito capaz de reconocer el lenguaje que ésta define. Recíprocamente, dado un autómata finito, se puede expresar mediante una expresión regular el lenguaje que reconoce, es decir, se pasar de AFD a una Expresión Regular.

Si se tiene un AFD $M = (Q, \Sigma, \delta, q_0, F)$ y se desea saber la expresión regular que este acepte, puede aplicarse un método de construcción muy sencillo descrito inormalmente como sigue:

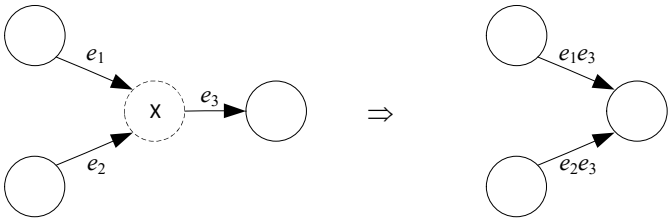
i) Agregar un nuevo estado inicial *i* conectado con una transición λ a *q*.
ii) Agregar un nuevo estado final *f* y agregar transiciones λ desde todos los $p \in F$ hasta *f*.
iii) Seleccione cualquier estado *q* del AFD extendido, salvo *i* o *f*, y eliminarlo del autómata. Al eliminarlo deben preservarse los caminos anteriores, sólo que las transiciones serán consideradas expresiones regulares.
iv) Repetir el paso anterior hasta que solamente quede una transición entre *i* y *f*. La expresión asociada a esa transición es la Expresión Regular (ER) reconocida por el AFD.

A continuación, se muestran ejemplos de eliminación de estados.

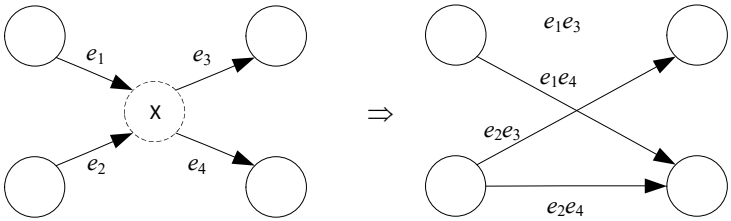
1.



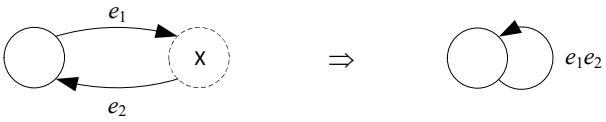
2.



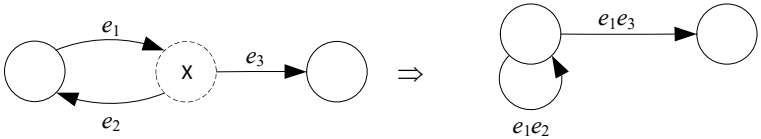
3.


e1
e3
X
e2
e4
⇒
e1e3
e1e4
e2e3
e2e4

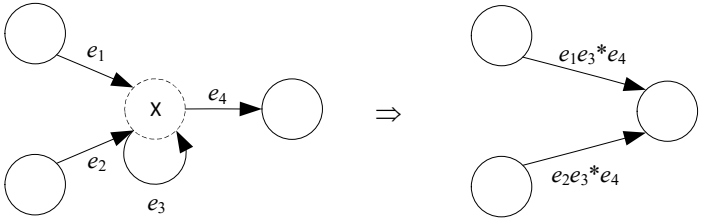
4.


e1
X
e2
⇒
e1e2

5.


e1
X
e3
e2
⇒
e1e3
e1e2

6.


e1
X
e4
e2
e3
⇒
e1e3*e4
e2e3*e4

Bibliografía

Cueva, J. M. (2001). *Lenguajes Gramáticas y Autómatas.* Oviedo - España: Universidad de Oviedo.

De Castro, R. (2004). *Teoría de la computación: Lenguajes, autómatas, gramaticas.* Bogotá - Colombia : UNIBIBLOS.

Díaz, V. J., & Cañete, J. M. (2007). *Lenguajes Formales y Automátas.* Sevilla - España.

Formella, A. (2014). *Teoría de Autómatas y Lenguajes Formales.* España: Universidad de Vigo.

Gómez, D., & Pardo, L. M. (2015). *Teoría de Autómatas y Lenguajes para ingenieros informáticos.* Santander - España: Universidad de Catambria.

Gutú, O. (2013). *Primer curso en teoría de autómatas y lenguajes formales.* México: Pearson.

Hopcroft, J. E., Motwani, R., & Ullman, J. D. (2008). *Introducción a la Teoría de autómatas lenguajes y computación* . Madrid - España: PEARSON EDUCACIÓN S.A.

Ibañez, V. (2011). *Lenguajes Formales y Autómatas.* Puno - Perú: Editorial Universitaria.

Jiménez, J. (2009). *Matemáticas para la computación.* México: Alfaomega Grupo.

Jurado, E. (2008). *Teoría de Autómatas y Lenguajes Formales.* España: Universidad de Extremadura.

Kolman, B., Busby, R., & Ross, S. (1997). *Estructuras de Matemáticas Discretas para la Computación.* México: Prentice-Hall.

Kovac, N. (2005). *Teória de Autómatas y Lenguajes Formales.* España.

Llopis, J. (23 de Mayo de 2019). *Matesfacil.com*. Obtenido de https://www.matesfacil.com/automatas-lenguajes/index.html

Rubio, F., & Rodríguez, I. (12 de Junio de 2019). *Teoría de Autómatas y Lenguajes Formales*. Obtenido de http://antares.sip.ucm.es:8180/webtalf/index.jsp

Sipser, M. (2006). *Introduction to the Theory of Computation.* United States: Thomson.