Matemática Discreta

CLASE N°9

Ing.Marcela Bellani
UNLAM | FLORENCIO VARELA 1903 .B1754.SAN JUSTO.BUENOS AIRES

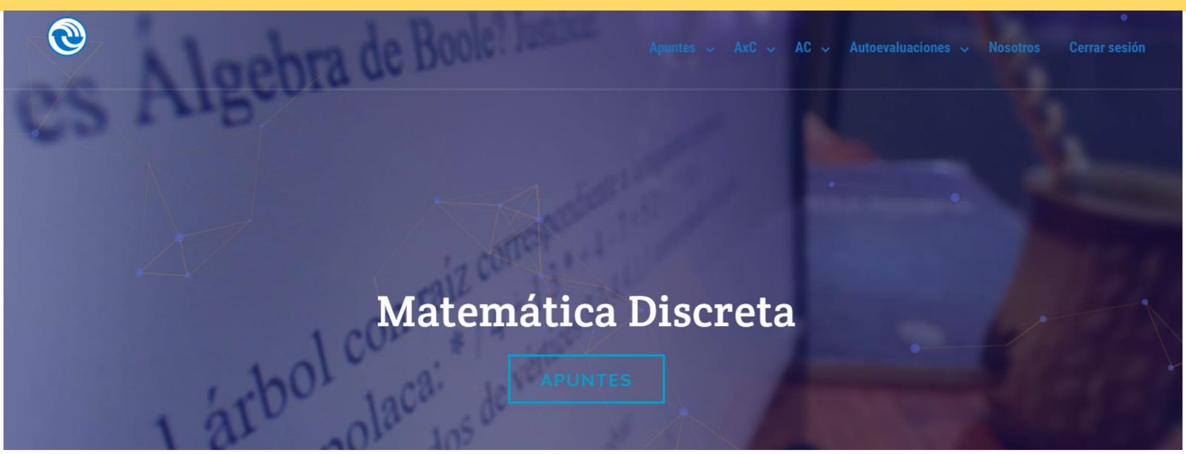
En esta clase se desarrollarán los siguientes temas:

-Gramáticas: definición, clasificación, operaciones.

-Expresiones regulares.

-Máquinas de estado finito: autómata finito y lenguajes regulares.

Para todas las clases contamos con la ayuda del sitio de la cátedra http://discretaunlam.net.ar donde encontrarás videos, ejercicios, explicaciones, autoevaluaciones de todos los temas de Matemática Discreta.



GRAMÁTICAS Y AUTÓMATAS FINITOS

Introducción

La Teoría de los Lenguajes Formales tiene su origen en un campo aparentemente bastante alejado de la Informática: la Lingüistica.

Los lingüistas de la llamada *escuela estructuralista americana* habían elaborado por los años 50 algunas ideas informales acerca de la gramática universal. Se entiende por gramática universal, una gramática que caracteriza las propiedades generales de cualquier lenguaje humano.

El primer trabajo que desarrolló teorías formales sobre gramáticas y lenguajes fue obra de *Avram Noam Chomsky*, quien es sin duda la figura más destacada de la lingüistica moderna, tanto por desarrollar sus fundamentos matemáticos, como por sus teorías sobre el origen y la naturaleza de los lenguajes naturales, aunque éstas últimas son más discutidas (*Chomsky, 1956; 1959; 1962; y 1963)*.

En el campo de la Informática, poco después de las primeras publicaciones de *Chomsky*, el concepto de Gramática Formal adquirió gran importancia para la especificación de lenguajes de programación; concretamente, se definió con sus teorías la sintaxis del lenguaje ALGOL 60 (con ligeras modificaciones sobre su versión primitiva), usándose una gramática libre de contexto. Ello condujo rápidamente al diseño riguroso de algoritmos de traducción y compilación.

Finalmente, y enlazando con el campo de la lingüistica, la Teoría de Lenguajes Formales es de gran utilidad para el trabajo en otros campos de la Informática como por ejemplo en Informática Teórica, Inteligencia Artificial, Procesamiento de lenguajes naturales (comprensión, generación, y traducción) y Reconocimiento del Habla.

La Teoría de los Lenguajes y Gramáticas Formales tiene una relación directa con la Teoría de Autómatas, siendo posible establecer entre ambas una correspondencia.

La Teoría de los Autómatas proviene del campo de la Ingeniería Eléctrica. El científico estadounidense *Claude Elwood Shannon (1916-2001)*, publicó varios trabajos, donde mostraba las bases para la aplicación de la Lógica Matemática a los circuitos combinatorios y secuenciales. A lo largo de las décadas siguientes, las ideas de *Shannon* se desarrollaron considerablemente, dando lugar a la Teoría de Autómatas (*Shannon 1949; 1954 y 1956*).

Los autómatas son sistemas que reciben información, la transforman y producen otra información que se transmite al entorno.

La Teoría de Autómatas tiene aplicación en campos muy diversos:

- Lógica de los Circuitos Secuenciales
- Teoría de Control de Sistemas
- Teoría de la Comunicación
- Arquitectura de Ordenadores
- Redes Conmutadoras y Codificadoras
- Teoría de los Sistemas Evolutivos y Auto-reproductivos
- Reconocimiento de patrones
- Redes Neuronales
- Reconocimiento y procesado de lenguajes de programación
- Traducción de lenguajes
- Teoría de Lenguajes Formales

Se aplican los lenguajes, gramáticas y autómatas de tipo 3 para la construcción de analizadores léxicos en el campo de los Traductores, Procesadores, Compiladores e Intérpretes, y los de tipo 2 para la construcción de analizadores sintácticos.

GRAMÁTICA

La gramática es un ente formal para especificar, de una manera finita, el conjunto de cadenas de símbolos que constituyen un lenguaje. Proporciona un conjunto de símbolos de varios tipos y un conjunto de reglas para construir las palabras. Se trata del sistema generador del lenguaje.

DEFINICIÓN FORMAL DE GRAMÁTICA: G = (V_{NT}, V_T, S, P)

Una gramática con estructura de frases G = (V_{NT}, V_T, S, P) consiste en:
un conjunto finito de símbolos no terminales, V_{NT},
un conjunto finito de símbolos terminales, V_T;
un *símbolo inicial* S que pertenece a V_{NT}
y un conjunto de producciones o de reglas de derivación P.

Todas las cadenas del lenguaje definido por la gramática están formadas con símbolos del ***vocabulario terminal*** **V_T**.
Los elementos del vocabulario terminal pueden ser representados por:
- letras minúsculas de comienzo del abecedario: *a, b, c,. . ., g.*
- operadores tales como: +, - , *, /,. . .
- caracteres especiales : # , @ , (,) , , ; , . . .
- los dígitos: 0, 1,. . ., 9
- las palabras reservadas de lenguajes de programación con letras minúsculas y en negrita : if, then, else, . . .

El ***vocabulario no terminal*** **V_{NT}** es el conjunto de símbolos introducidos como elementos auxiliares para la definición de la gramática, y que no figuran en las palabras del lenguaje. Permiten representar subconjuntos del lenguaje o estados intermedios de la generación de las palabras del lenguaje.
Los elementos del vocabulario no terminal pueden ser representados por:

- letras mayúsculas de comienzo del abecedario: A, B,. . ., G. La única excepción suele ser el símbolo inicial que se representa con S.
- nombres en minúscula, pero encerrados entre paréntesis angulares:
<expresión>,
<operador>, . . .

La **intersección** entre el **vocabulario terminal** y **no terminal** es el conjunto vacio:

$$V_N \cap V_T = \{ \}$$

La **unión** entre el **vocabulario terminal** y **no terminal** es el ***vocabulario***:

$$V_N \cup V_T = V$$

En ocasiones es importante distinguir si un determinado vocabulario V incluye o no la hilera nula, indicándose respectivamente con superíndice *, V^*, o superíndice +, V^+.

El ***símbolo inicial* S** es un símbolo no terminal a partir del cual se aplican las reglas de la gramática para obtener las distintas cadenas del lenguaje.

La ***producción* P** es un conjunto de reglas que se aplican desde el símbolo inicial para obtener las cadenas del lenguaje. Representan combinaciones cualesquiera de símbolos terminales y no terminales, con la restricción de que en la parte izquierda de la regla debe haber al menos un símbolo no terminal. El lado derecho de una regla puede contener la hilera nula. El conjunto de producciones P se define por medio de la enumeración de las distintas producciones, en forma de reglas o por medio de un metalenguaje por ejemplo BNF (*Backus Naur Form*) o EBNF (*Extended Backus Naur Form*).

Para simplificar el conjunto de reglas que forman la producción se suele agrupar todas las producciones de cada símbolo no terminal separando las partes derechas por el símbolo /.

⌦ *Ejemplo*

Dado el conjunto de producciones:

$$P\begin{cases} S \rightarrow 10 \\ S \rightarrow 1S0 \end{cases}$$

Simplificando las reglas de la producción se puede reescribir

P = {S→10 / 1S0

Su gramática es: G = (V_{NT}, V_T, S, P) siendo V_T = {1, 0}, V_{NT} = {S}

El mecanismo mediante el cual, a partir de una gramática, se obtienen las palabras del lenguaje se llama **derivación.**

Las palabras pertenecen al lenguaje si podemos construir un àrbol con raíz ordenado para ellas llamado **árbol de derivación**. El árbol de derivación también es llamado ***parse tree.*** Derivar el árbol de derivación se llama ***parsing.*** La tarea de determinar si un string/cadena/ palabra es válido se llama ***reconocimiento***. Es decir, el reconocimiento puede ser logrado mediante el parsing.

¿Cómo se derivación las palabras del lenguaje, L (G), a partir de su gramática G?

1. Se comienza con el símbolo inicial S.
2. Se aplican las producciones de P al símbolo inicial S hasta obtener solamente símbolos terminales.
3. Cualquier hilera que se obtenga es un elemento del lenguaje L (G).

⌦ *Ejemplos*

1. Sea la gramática: G = (V_{NT}, V_T, *S*, P) donde V_T = {1, 0}, V_{NT} = {S}, y el conjunto de producciones:

$$P\begin{cases} S \to 10 \\ S \to 1S0 \end{cases}$$

Reescribiendo P= {S→ 10 / 1S0

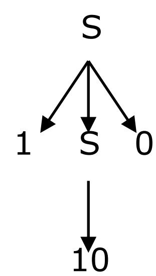
Se obtiene la siguiente derivación directa, al sustituir la primera regla en la segunda:

S→1**S**0→1**10**0

Árbol de derivación: es una manera de representar las derivaciones de una gramática. Sólo se pueden definir árboles de derivación para gramáticas de tipo 2 y tipo 3.

Para cada palabra hay un árbol de derivación.

La raíz del árbol de derivación representa al símbolo inicial. Los nodos internos del árbol representan los símbolos no terminales que aparecen en la derivación. Las hojas del árbol representan los símbolos terminales.



2.Obtener derivaciones de las palabras 004 y 0004 a partir de la siguiente Gramática G= ({0,4}, {S, S_1}, P, S)

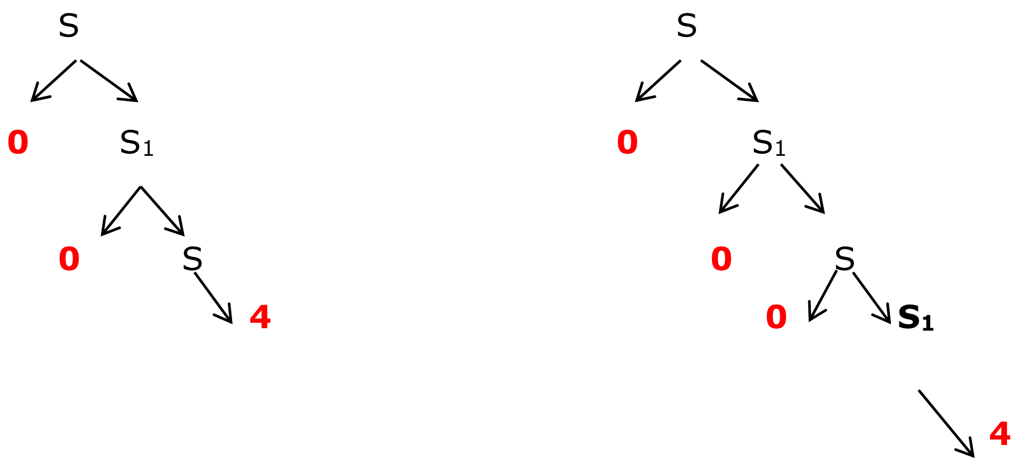
$$P = \begin{cases} S \to 0S_1 / 4 \\ S_1 \to 0S / 4 \end{cases}$$

Las palabras se pueden obtener con las siguientes derivaciones:

004: S → 0S_1 → 0**0S** → 0**04**

0004: S → 0S_1 → 0**0S** →0**00S_1**→ 0**004**

Cuyos árboles de derivación son los siguientes:



3.Sea la gramática G = ({*S, A, B*}, {*a, b, c, d*}, *S*, P) donde P son las siguientes reglas de producción

(1) $S \rightarrow ASB$

(2) $A \rightarrow b$

(3) $aaA \rightarrow aaBB$

(4) $S \rightarrow d$

(5) $A \rightarrow aA$

(6) $B \rightarrow dcd$

Las producciones se pueden reescribir de la siguiente manera:

$S \rightarrow ASB \,/ d$

$A \rightarrow b \,/\, aA$

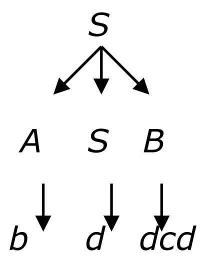
$aaA \rightarrow aaBB$

$B \rightarrow dcd$

Por aplicación de derivaciones inmediatas a partir del símbolo inicial obtenemos algunas palabras del lenguaje generado por esa gramática:

$$S \xrightarrow{(1)} ASB \xrightarrow{(2)} bSB \xrightarrow{(4)} bdB \xrightarrow{(6)} \boldsymbol{bddcd}$$

Su árbol de derivación es:



$$S \xrightarrow{(1)} ASB \xrightarrow{(5)} aASB \xrightarrow{(5)} aaAASBB \xrightarrow{(3)} aaBBASBB \xrightarrow{(6)} aadcddcdASdcddcd \xrightarrow{(2)}$$

$$aadcddcdbSddcddcd \xrightarrow{(1)} \boldsymbol{aadcddcdbdddcddcd}$$

$$S \xrightarrow{(1)} ASB \xrightarrow{(2)} bSB \xrightarrow{(6)} bSdcd \xrightarrow{(4)} \boldsymbol{bddcd}$$

El lenguaje generado por una gramática G, **L (G),** es el conjunto de todas las sentencias de la gramática.

Propiedad

Dos gramáticas son equivalentes si ambas generan el mismo lenguaje.

G1 y G2 son equivalentes si L(G1) =L(G2)

Ejercicios resueltos

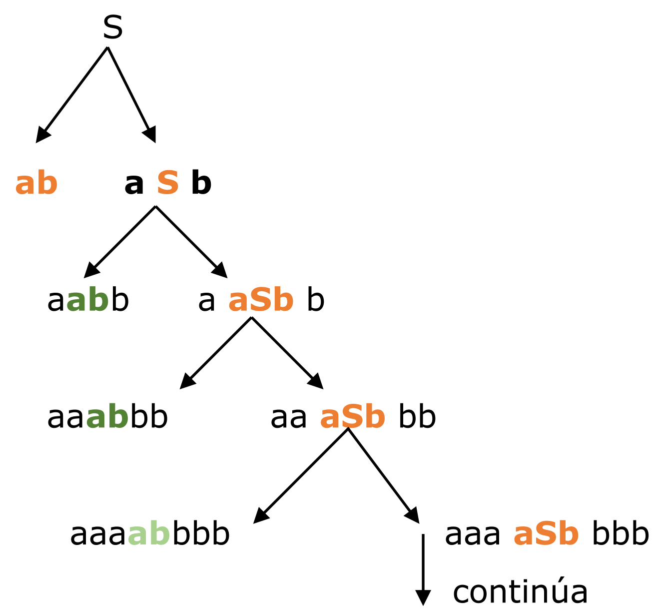
1.Sea la gramática G = ({S}, {a,b}, *S*, P) donde P={(S→ aSb), (S→ ab)}. Determinar el lenguaje que genera.

$$S \to \mathrm{aSb} \to \mathrm{aaSbb} \to \mathrm{a}^3\mathrm{Sb}^3 \to .... \to a^{(n-1)}S\ b^{(n-1)} \to a^n b^n$$

Aplicando la primera producción n-1 veces, seguida por la aplicación de la segunda producción, el lenguaje generado es:

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

Árbol de derivación del lenguaje



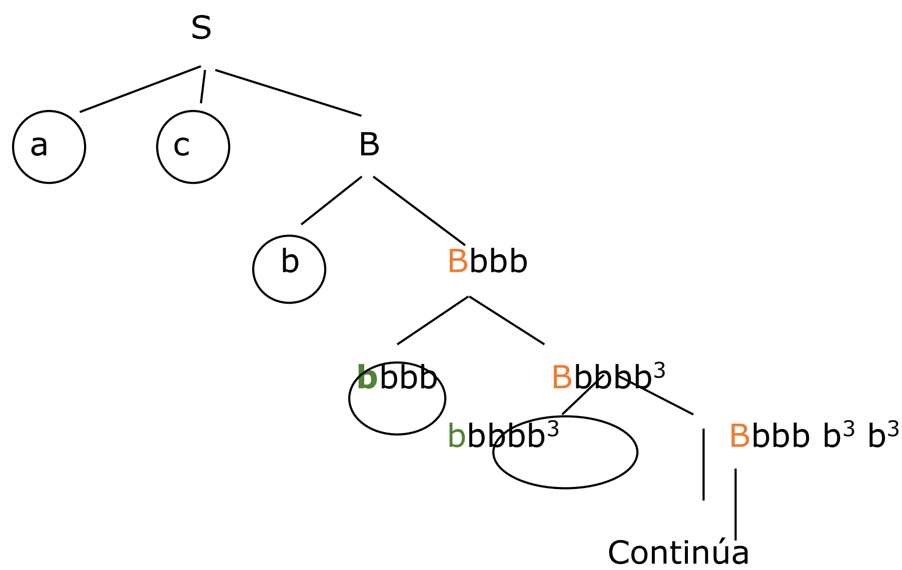
2.Sea la producción P = {(S → a), (S→ c), (S→ B), (B→b), (B→ Bbbb)}
Definir la gramática y hallar el lenguaje generado.

$$P = \begin{cases} \mathrm{S} \to \mathrm{a} \ / \mathrm{c} \ / \ \mathrm{B} \\ \mathrm{B} \to \mathrm{b} \ / \ \mathrm{Bbbb} \end{cases}$$

Gramática

G= ({S, B}, {a, b, c}, P, S)

Árbol de derivación del lenguaje



Lenguaje generado por la gramática G

L(G)={ w ϵ V^+/ w = a ∨ w = c ∨ w = b^{3k+1}, k ≥ 0}

Tipos de gramáticas con estructura de frases

Chomsky definió cuatro tipos distintos de gramáticas de acuerdo a la forma de las reglas de la producción P. La clasificación comienza con un tipo de gramáticas que pretende ser universal, aplicando restricciones a sus reglas de derivación se van obteniendo los otros tres tipos de gramáticas. Esta clasificación es jerárquica, es decir cada tipo de gramática engloba a todos los tipos siguientes.

Gramáticas de tipo 0

Las reglas de derivación son de la forma: $w_1 \rightarrow w_2$ siendo $w_1 \in \{V_T, V_N\}^+$ y $w_2 \in \{V_T, V_N\}^*$, es decir las únicas restricciónes son que no puede haber reglas de la forma:

$\lambda \rightarrow w_2$ y w_1 debe tener por lo menos un símbolo no terminal

⌦ ***Ejemplos de gramáticas de tipo 0***

Todas las gramáticas mostradas en los ejemplos anteriores son de tipo 0, pues en ninguna de ellas existe la producción $\lambda \rightarrow w_2$.

Gramáticas de tipo 1 o gramáticas sensibles al contexto

En ellas las reglas de producción son de la forma $w_1 \rightarrow w_2$ donde la longitud de w_2 es mayor que la longitud de w_1 o bien de la forma $w_1 \rightarrow \lambda$.

Este tipo de gramática también es llamada gramáticas sensibles al contexto ya que si una de las reglas de la producción $w_1 \rightarrow w_2$ es de la forma $\alpha A \beta \rightarrow \alpha \gamma \beta$; A puede ser reemplazada por γ sólo cuando está entre las cadenas α y β.

Por lo tanto, para ser una gramática de tipo 1 cada una de las reglas de la producción P cumplirá que la ***longitud de la parte derecha de la producción es mayor o igual a la de la parte izquierda o bien el lado derecho es la hilera nula pero además debe tenerse en cuenta que es lo que viene antes y después del símbolo que se está sustituyendo***.

⌦ ***Ejemplos de gramáticas de tipo 1***

- La gramática G = ({S, A, B}, {a, b}, S, P) cuyas producciones P se muestran a continuación es de tipo 1.

$$P = \begin{cases} S & \rightarrow ab \\ S & \rightarrow bA \\ A & \rightarrow \ a \\ A & \rightarrow aS \end{cases}$$

Cada una de las reglas de la producción P cumple con la condición para ser una gramática de tipo 1. Es decir:

long (S) ≤ long (ab)

long (S) ≤ long (bA)

long (A) ≤ long (a)

long (A) ≤ long (aS)

- La gramática G = ({A, S}, {a, b}, S, P) donde P son las producciones que se muestran a continuación es de tipo 1.

$$P = \begin{cases} S & \rightarrow aS \ ; \ \text{long } (S) \ \leq \ \text{long } (aS) \\ S & \rightarrow aS \ \ ; \ \text{long } (S) \ \leq \ \text{long } (aS) \\ A & \rightarrow bA \ ; \ \text{long } (A) \ \leq \ \text{long } (bA \\ A & \rightarrow \lambda \end{cases}$$

Cada una de las sentencias de la producción cumple con la condición para ser una gramática de tipo 1.

⌦ ***Ejemplos de gramáticas que No son de tipo 1***

- La gramática definida como G = ({S}, {a, b}, S, P) donde P son las siguientes producciones:

$$P = \begin{cases} \text{S} \rightarrow \text{aaaSbbb} \\ \text{aSb} \rightarrow \text{ab} \end{cases}$$

No es del tipo 1, ya que la producción aSb →ab no cumple con la condición:

longitud de w1≤ longitud de w2

Para que sea una gramática de tipo 1 la segunda regla debería ser:

aSb →aba o bien S →ab

- La gramática G = ({S, A}, {a, b}, S, P) con las producciones P siguientes:

$$P = \begin{cases} \text{S} \rightarrow \text{abSA} \\ \text{abA} \rightarrow \text{baab} \\ \text{S} \rightarrow \text{a} \\ \text{A} \rightarrow \text{b} \end{cases}$$

No es del tipo 1, ya que la producción abA →baab no mantiene constante lo que está antes del símbolo A.

Para que sea una gramática de tipo 1 la segunda regla debería ser abA →**ab**ab.

Gramáticas de tipo 2 o libre de contexto

Sus reglas de producción tan sólo admiten tener un símbolo no terminal en su parte izquierda, es decir son de la forma: $w_1 \rightarrow w_2$ donde w_1 es un único símbolo no terminal.

La denominación contexto libre se debe a que se puede reemplazar w_1 (símbolo no terminal) en una cadena siempre que aparezca independientemente de lo que figure en la cadena.

⌦ ***Ejemplos de gramáticas de tipo 2***

La gramática G = ({S, A, B}, {a, b}, S, P) cuyas producciones P se muestran a continuación es de tipo 2.

$$P = \begin{cases} S \to aB \\ S \to bA \\ A \to a \\ A \to aS \end{cases}$$

- La gramática G = ({S, A, B}, {a, b}, S, P) cuyas producciones P se muestran a continuación es de tipo 2.

$$P = \begin{cases} A \to aAA \\ B \to b \\ B \to aS \\ B \to aBB \end{cases}$$

Gramáticas de tipo 3 o regulares

Sus reglas de producción sólo pueden ser de la forma $w_1 \to w_2$ donde w_1 es un único símbolo no terminal y $w_2 = xA$ o $w_2 = x$ o bien $w_2 = \lambda$ donde A es símbolo no terminal y x es símbolo terminal.

Si $w_2 = xA$ se llama regular a derecha (el símbolo no terminal se encuentra a la derecha del símbolo terminal).

Si w_2 = Ax se llama regular a izquierda (el símbolo no terminal se encuentra a la izquierda del símbolo terminal).

⌦ *Ejemplos de gramáticas de tipo 3*

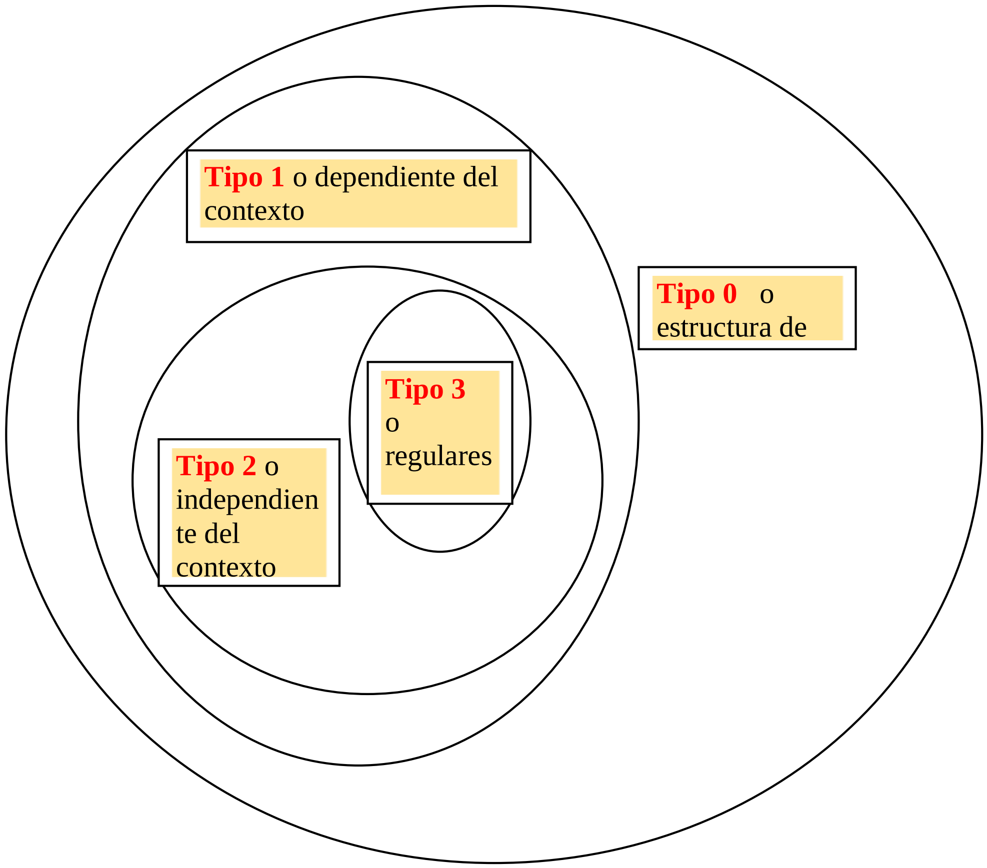
La gramática G = ({a, b}, {A, S}, S, P) donde P son las producciones que se muestran a continuación es de tipo 3.

$$P = \begin{cases} S \to aA \\ S \to bA \\ A \to aA \\ A \to a \end{cases}$$

Resumiendo:

Tipos de gramáticas	
Tipo	**Restricciones en las producciones $w_1 \to w_2$**
0	Única restricción $w_1 \neq \lambda$
1	longitud de $w_2 \geq$ la longitud de w_1 ó $w_2 = \lambda$ y se mantiene el contexto
2	w_1= A ,siendo A un símbolo no terminal
3	w_1= A y w_2 = aB ó w_2 = a,siendo A y B símbolos no terminales,a símbolo terminal ó bien $A \to \lambda$

JERARQUÍA DE LAS GRAMÁTICAS



Ejercicio resuelto

1.Clasificar las gramáticas de acuerdo a las derivaciones de las producciones indicadas:

G = ({X, Y, Z, T}, {0, 1}, P, X)

a) X → TYZ

T → 0 / 1

0Y → 1

1Y → 0

0Z → 1

1Z→ 0 / 1

b) X → ZT

ZT →TY

Y → 1

T → 0

c) $X \rightarrow ZY / YZ$

$Z \rightarrow 1$

$Y \rightarrow 0$

d) $X \rightarrow 1Y / 1Z$

$Y \rightarrow 0$

$Z \rightarrow 0 / 1$

Solución

Todas las gramáticas son de tipo cero ya que en todas se cumple que $w_1 \neq \lambda$.

La gramática a) no es de tipo 1 ya que la longitud (X) ≤ longitud (TYZ). Por lo tanto, la gramática a) es de tipo 0.

Las otras gramáticas son de tipo 1 ya que cumplen con la condición de que la longitud de $w_2 \geq$ la longitud de w_1 para cada una de las reglas que forman la producción.

Las gramáticas c) y d) son de tipo 2 pues el lado izquierdo de cada una de las sentencias de sus producciones está formado por un único símbolo no terminal. Esto no ocurre en la gramática b) en la cual el lado izquierdo de la segunda regla de su producción está formada por dos símbolos no terminales: $ZT \rightarrow TY$ .Por lo tanto la gramática b) es de tipo 1.

La gramática c) no es de tipo 3 ya que el lado derecho de la primera regla de su producción está formada por dos símbolos no terminales. Por lo tanto, la gramática c) es de tipo 2.

La gramática d) es de tipo 3 ya que el lado derecho de las reglas de su producción están formadas por la concatenación a derecha de un símbolo terminal con un símbolo no terminal: $X \rightarrow 1Y / 1Z$ y por símbolos terminales únicamente: $Y \rightarrow 0$; $Z \rightarrow 0 / 1$

CORRESPONDENCIA ENTRE GRAMÁTICAS Y LENGUAJES

Se denomina *lenguaje de tipo 0* al generado por una gramática de tipo 0. De la misma forma, se denominan *lenguajes de tipo 1, tipo 2, y tipo 3*, a los generados por las gramáticas de tipo 1, tipo 2, y tipo 3, respectivamente.

Si los lenguajes generados por los distintos tipos de gramáticas se relacionan entre sí con respecto a la relación de inclusión se obtiene:

$$\{L(G3)\} \subseteq \{L(G2)\} \subseteq \{L(G1)\} \subseteq \{L(G0)\}$$

Según lo visto anteriormente existe una correspondencia entre las gramáticas y los lenguajes de tal forma que se genera una jerarquía de lenguajes análoga a la mostrada para las gramáticas.

Aceptador de un lenguaje: Autómata

Un autómata es un modelo matemático de un sistema que manipula cadenas de símbolos que se le presentan a su entrada determinando si esa cadena pertenece al lenguaje que el autómata reconoce.

La salida en cada instante tiene dos posibilidades: aceptar la palabra de entrada; o no aceptarla.

El autómata recibe los símbolos de entrada, uno detrás de otro, es decir secuencialmente.

El símbolo de salida que en un instante determinado produce un autómata, no sólo depende del último símbolo recibido a la entrada, sino de toda la secuencia o cadena, que ha recibido hasta ese instante. Los autómatas transitan entre estados de un conjunto de estados según reciben los símbolos que forman la palabra de entrada.

Todo lo anterior conduce a definir un concepto fundamental: ***estado de un autómata***.

El **estado** de un autómata es toda la información necesaria en un momento dado, para poder deducir, dado un símbolo de entrada en ese momento, cual será el símbolo de salida.

Es decir, conocer el estado de un autómata, es lo mismo que conocer toda la historia de símbolos de entrada. Hay tres tipos de estados: **estado inicial**, permite empezar la ejecución del autómata; **estados finales**, permiten realizar la salida de aceptación de la palabra de entrada en el caso de que no haya más símbolos en la entrada; y **estados intermedios.**

El autómata tendrá un determinado *número de estados*, y se encontrará en uno u otro según sea la historia de símbolos que le han llegado.

Si un autómata se encuentra en un estado determinado, recibe un símbolo también determinado, producirá un símbolo de salida y efectuará un cambio o *transición* a otro estado (también puede quedarse en el mismo estado).

El campo de estudio de los Traductores, Procesadores e Intérpretes son los lenguajes y las gramáticas que los generan. Los elementos del lenguaje son sentencias, palabras, etc.… formadas a partir de un alfabeto o vocabulario, que no es otra cosa que un conjunto finito de símbolos. Establecidas las reglas gramaticales, una cadena de símbolos pertenecerá al correspondiente lenguaje si tal cadena se ha formado obedeciendo esas reglas. Entonces un **autómata reconocedor de ese lenguaje**, funciona de tal forma que cuando reciba a su entrada una determinada cadena de símbolos indica si dicha cadena pertenece o no al lenguaje.

Existe un tipo de autómata para reconocer cada uno de los tipos de lenguajes generados por las correspondientes gramáticas.

Lenguaje	Reconocedor de lenguaje
Tipo 0	Máquina de Turing o autómata de tipo 0
Tipo 1	Autómata ligado lineal
Tipo 2	Autómata de pila
Tipo 3	Autómata finito

Sólo estudiaremos los autómatas finitos que son los dispositivos que reconocen lenguajes regulares y ya vimos que los lenguajes regulares son generados por gramáticas regulares.

Para describir lenguajes de tipo 3 o lenguajes regulares se utilizan las expresiones regulares. Se trata de un metalenguaje para describir los lenguajes regulares.

Son lenguajes regulares:

- Lenguaje vacio $\varnothing$
- Lenguaje nulo $\Delta = \{ \lambda \}$
- Si V es un alfabeto y $a \in V$ entonces $\{a\}$ es un lenguaje regular
- Si L_1 y L_2 son lenguajes regulares entonces $L_1 \cup L_2$, $L_1 . L_2$, L_1^* son lenguajes regulares
- Ningún otro lenguaje sobre V es regular

⌧ Ejemplo

Sea V= {0,1} los siguientes lenguajes son regulares:

$L_1 = \{01,11\}$

$L_2 = \{111\}$

$L_1 \cup L_2 = \{01, 11,111\}$

$L_1^* = V^*$

Sea V un alfabeto. Una expresión regular sobre V es una secuencia de elementos de V conectados por los siguientes símbolos (,),concatenación **.**,$\vee$,*, que verifican:

λ es una expresión regular

Si x e y son expresiones regulares entonces x. y es una expresión regular

Si x e y son expresiones regulares entonces x $\vee$ y es una expresión regular

Si x es una expresión regular entonces (x)* es una expresión regular.

⌧ Ejemplos

- Sea el vocabulario {0,1} y la expresión regular 0 0* 1 1*. Indicar el lenguaje que denota, y algunas cadenas de dicho lenguaje.

Algunas cadenas son:

01

001

00001

0111

011

000001

El lenguaje que se describe es L= {cadenas que comienzan por una 0 y continuan con varios o ningún 0, y siguen con 1 y continúan con varios o ningún 1}

- Sea el vocabulario {a, b}, la expresión regular a (ab)* denota el conjunto de cadenas que empiezan por a y van seguidas por (ab) cualquier nº de veces o ninguna.

- Sea el vocabulario {a, b}, la expresión regular (a $\vee$ b) b* denota el conjunto de cadenas que empiezan con una a ó una b, la que puede o no estar seguida por una b ó más bs.

Los autómatas que reconocen a los lenguajes regulares son los autómatas finitos.

Autómata finito

Definición formal de autómata finito

Se puede definir como una quíntupla AF= (Q, V, δ, q0, F) donde:

Q = conjunto finito de estados

E = conjunto finito de símbolos de entrada, que constituye el vocabulario terminal de la gramática

δ = la *función de cambio o transición de estados δ*: Q x V → Q; define la transición de estado ante el procesamiento de una hilera de símbolos de entrada.

q_0 = el *estado inicial* (es único)

F = el *conjunto de estados finales* (puede haber más de uno)

Cuando un autómata finito transita a un estado final partiendo de un estado inicial, en varios movimientos, se dice que se ha producido aceptación o ***reconocimiento de la cadena*** de entrada. Es decir que dicha cadena, pertenece al lenguaje reconocido por el autómata. Se simboliza con L (AF) al conjunto de las hileras que acepta el autómata finito, por lo tanto w (hilera) $\in$ L (AF) si y sólo si existe un camino del estado inicial q_0 al estado final q.

Por el contrario, cuando el autómata finito no es capaz de llegar a un estado final, se dice que el autómata no reconoce dicha cadena y que por tanto no pertenece al lenguaje.

Los autómatas finitos se pueden representar mediante:

- Tabla de transición de estados
- Diagrama de transición o de estados

Tabla de transición de estados

La función de transición de estados δ puede representarse mediante una tabla, con tantas filas como estados y tantas columnas como entradas. Así por ejemplo se puede representar el autómata finito AF= (Q, V, δ, q0, F) donde:

Q = {q_1, q2, q3};

V= {0,1};

F= {q3}

Y la tabla de transición de estados:

δ	**0**	**1**
q1	q2	q1
q2	q3	q2
q3	____	____

O bien:

$\delta(q_1,0)$ = q2

$\delta(q_1,1)$ = q_1

$\delta(q_2,0)$ = q3

$\delta(q_2,1)$ = q2

$\delta(q_3,0)$ = ---

$\delta(q_3,1)$ = ---

Diagrama de transición o de estados

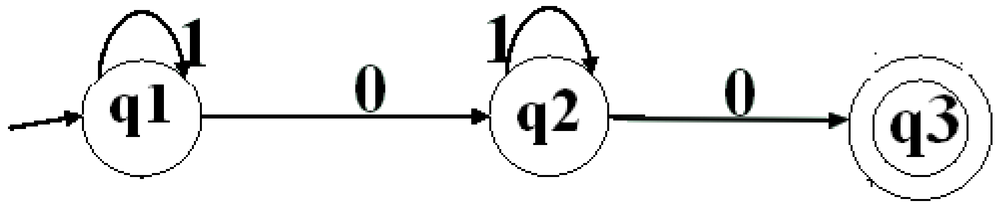
La forma habitual de representar los autómatas finitos es mediante un dígrafo o diagrama de transición de estados. Los vértices del dígrafo se llaman estados y se usan para señalar, en ese momento, hasta que lugar se ha realizado la cadena. Las aristas del grafo se etiquetan con caracteres del alfabeto y se llaman transiciones. Si el siguiente carácter a reconocer concuerda con la etiqueta de alguna transición que parte del estado actual, nos desplazamos al estado al que nos lleve la arista correspondiente.

Se comienza en un estado inicial, y cuando se hayan tratado todos los caracteres de la palabra (cadena) correspondiente, necesitamos saber si la cadena es "legal". Para ello se marcan ciertos estados como estados de aceptación o estados finales (doble círculo). Toda cadena que surja de una transición desde el estado inicial a un estado final de aceptación es "legal". Marcamos el estado inicial con una flecha (→) y alrededor de los estados de aceptación trazamos un círculo.

Resumiendo:

- ***Círculos***: estados.
- ***Estado inicial***: marcado con un puntero.
- ***Estados finales o de aceptación***: marcados por círculos dobles.
- ***Estados "intermedios".***
- ***Arcos dirigidos***: transiciones.
- ***Etiquetados*** con símbolos o categorías de símbolos.

Se transita de un estado a otro tras recibir a la entrada el símbolo de la etiqueta.

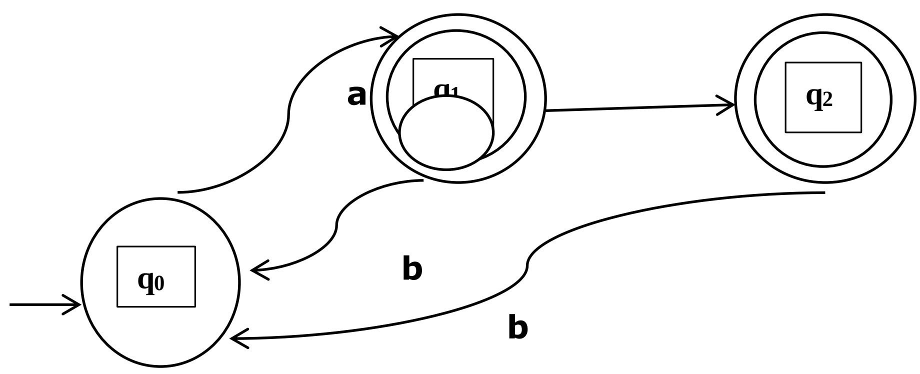


__Proceso de reconocimiento de una cadena:__

- Se parte del estado inicial (actual)
- Se lee la cadena símbolo a símbolo de izquierda a derecha.
- Por cada símbolo leído se produce una transición desde el estado actual a otro a través de la flecha cuya etiqueta coincide con el símbolo leído.
- La cadena es reconocida si tras realizar las transiciones correspondientes a la
  cadena completa se alcanza un estado de aceptación (final).

Ejemplo

¿La cadena **abaa** es aceptada por el autómata de estado finito cuyo diagrama de estado es el siguiente?



__Solución__

- Comenzando en el estado inicial **q_0** cuando entra **a** se pasa al estado **q_1**.
- En el estado **q_1** cuando **b** entra se pasa al estado **q_0**.

- En el estado **q_0** si la entrada es **a** se pasa al estado **q_1**.
- Estando en **q_1** cuando **a** entra se pasa al estado **q_2**, que es un estado final o de aceptación.

Por lo tanto, la palabra **abaa** es aceptada por el autómata finito.

Teoremas

Para toda gramática regular, G, existe un autómata finito, AF, tal que el lenguaje reconocido por el autómata finito es igual al lenguaje generado por la gramática.

$$L(AF) = L(G)$$

Para todo autómata finito, AF, existe una gramática regular, G, tal que el lenguaje
generado por la gramática es igual al lenguaje reconocido por el autómata finito

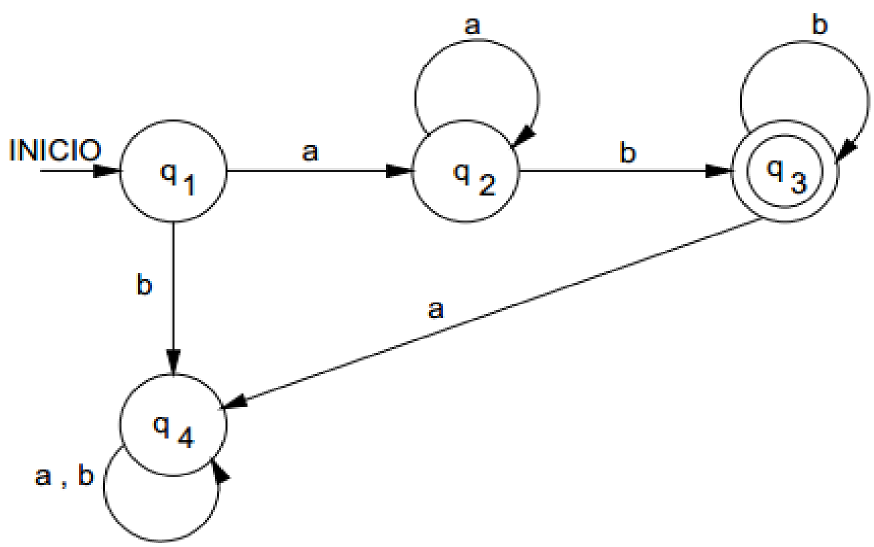
$$L(G) = L(AF)$$

⌦ Ejemplo

Sea el autómata finito AF=($\{q_1,q_2,q_3,q_4\},\{a, b\}, \delta , q_1,\{q_3\}$) donde la función de transición de estados está dada por la tabla siguiente:

δ	***a***	***b***
q1	q2	q4
q2	q2	q3
q3	q4	q3
q4	q4	q4

Determinar el lenguaje que reconoce, representar el diagrama de estados, e indicar la expresión regular que representa al lenguaje.
Se construye el diagrama de estados, colocando en primer lugar todos los estados dentro de círculos, marcando con doble círculo el estado final. El estado inicial se indica con una flecha. Para construir los arcos, nos situamos en el primer estado de la tabla de transiciones y se observa que f(q1, a) = q2,entonces se traza una flecha entre q1 y q2, apuntando a q2, y se coloca encima de la flecha el símbolo del vocabulario de entrada a. De igual forma se recorre la tabla de transiciones para cada estado y entrada completándose el diagrama de estados.



El lenguaje generado se obtiene partiendo del estado inicial y recorriendo todos los caminos posibles para alcanzar el estado final. Así se obtiene que este autómata reconoce el lenguaje:

L(G) ={ab,aab,…,abb,…,aabbb,aabbbb,…}

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

Por lo tanto, L(AF)= L(G)= $\{a^nb^m / n \geq 1, m \geq 1\}$

La expresión regular que denota el lenguaje es a^+b^+ o también aa^*bb^*.

¿Cuál es la gramática del lenguaje que reconoce el AF?

Sabemos que los AF reconocen lenguajes regulares los cuales son generados por gramáticas de tipo 3.Por lo tanto el lado izquierdo de cada una de las reglas de producción de la gramática está formado por un único símbolo no terminal y el lado derecho por un símbolo terminal seguido por un símbolo no terminal o bien la hilera nula λ.

Si AF = (Q, V, δ, q_0, F) y G = (V_N ; V_T ; P ; S) entonces V_N =Q ; V_T = V ; S = q_0 y las producciones de P están dadas por :

$qi \rightarrow a\ \delta\ (q_i, a)$

$qi \rightarrow \lambda$ si $qi \in F$

Para el ejemplo, la gramática G que genera el lenguaje L (G) =L (AF) es **G = ($V_{NT;}$ $V_{T;}$ P; S)** siendo:

V_{NT} = {S, S_1, S_2, S_3}

V_T = {a, b}

S = q_1

Y las reglas de la producción son:

$S \rightarrow aS1$	pues	$\delta\ (q_1, a) = q_2$
$S_3 \rightarrow bS3$	pues	$\delta\ (q4, b) = q4$
$S_1 \rightarrow aS1$	pues	$\delta\ (q_2, a) = q_2$
$S_1 \rightarrow bS_2$	pues	$\delta\ (q2, b) = q3$
$S_2 \rightarrow aS3$	pues	$\delta\ (q_3, a) = q4$
$S_2 \rightarrow bS2$	pues	$\delta\ (q3, b) = q3$
$S_2 \rightarrow \lambda$	pues	$q_3 \in F$
$S_3 \rightarrow bS3$	pues	$\delta\ (q_4, b) = q4$

La gramática de tipo 3 es G = ({S, S_1, S_2, S_3}, {a, b}, P; S)

Clasificación de los autómatas finitos

De acuerdo a la función de transición de estado δ los autómatas finitos se clasifican en autómatas *finitos deterministas AFD* y *autómatas finitos no deterministas AFND*.

Autómatas finitos deterministas AFD

En un autómata finito determinista AFD la función de transición no presenta ninguna ambigüedad en las transiciones de estados para una entrada dada. Es decir, por cada entrada *a* en un estado qi, existe una única transición hacia otro estado qj o a ningún otro estado.

Por lo tanto:

$$\forall\ q \in Q,\ \forall\ a \in V \text{ se verifica } |\ \delta(q, a)\ | \leq 1,$$

Además en un AFD no hay ninguna arista etiquetada con λ

El autómata finito del ejemplo anterior es un AFD ya las letras del alfabeto al salir de un estado se dirigen a un único estado o a ninguno. Esto lo podemos analizar desde la tabla de transición de estado:

$\delta(q_1,a) = q2 \rightarrow |\ \delta(q_1,a)\ | = 1$

$\delta(q_1,b) = q4 \rightarrow |\ \delta(q4,b)\ | = 1$

$\delta(q_2,a) = q2 \rightarrow |\ \delta(q_2,a)\ | = 1$

$\delta(q_2,b) = q3 \rightarrow |\ \delta(q3,b)\ | = 1$

$\delta(q_3,a) = q4 \rightarrow |\ \delta(q_3,a)\ | = 1$

$\delta(q_3,b) = q3 \rightarrow |\ \delta(q3,b)\ | = 1$

$\delta(q_4,a) = \text{--} \rightarrow |\ \delta(q_4,a)\ | = 0$

$\delta(q_4,b) = q4 \rightarrow |\ \delta(q_4,b)\ | = 1$

Observación: 'Determinista' significa que no tenemos opción ninguna para eligir.

Autómatas finitos no deterministas AFND

Si se permite que desde un estado se realicen cero, una o más transiciones mediante el mismo símbolo de entrada, se dice que el autómata finito es no determinista. A veces esmás conveniente diseñar autómatas finitos no determinista (AFN) en lugar de deterministas.

Por lo tanto:

$$\exists\ q \in Q,\ \exists\ a \in V \cup \{\ \lambda\ \} \text{ que verifica } \ |\ \delta\ (q,a)\ |\ > 1\ ,$$

Además en un AFND hay aristas que pueden estar etiquetadas con λ.Es decir puede hacer transiciones de un estado a otro sin leer ningún símbolo de entrada.

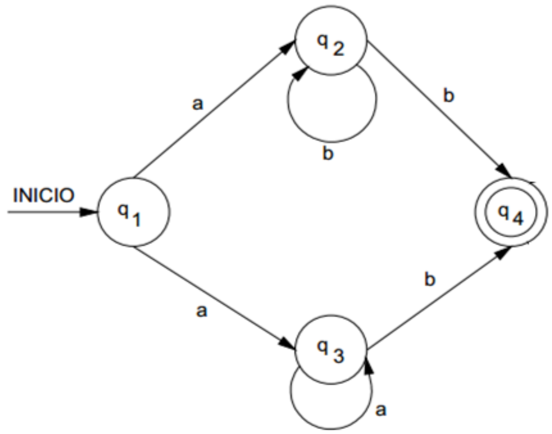
⌦ Ejemplo

Sea el autómata finito AFND = ({q_1, q_2, q_3, q_4}, {a, b}, δ, q_1, {q_4}), y la función δ viene dada por la siguiente tabla:

δ	***a***	***b***
q1	{q2,q3}	_____
q2	______.	{q2,q4}
q3	q3	q4
q4	_____	______

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

El diagrama de estados se construye al igual que en el ejemplo anterior de autómatas finitos, con la salvedad de que para una entrada a un estado puede salir más de una flecha de un determinado estado.



El lenguaje reconocido es el siguiente: L (G) = {ab, abb, abbb,…,aab, aaab, aaaab,…}

o también $L(G)=\{ab^n ,\ n \geq 1 \ \vee\ a^m b ,\ m \geq 1 \}$

Por lo tanto $T\ (AF)= L\ (G)= \{ab^n ,\ n \geq 1 \ \vee\ a^m b ,\ m \geq 1 \}$ (expresión del lenguaje como conjunto)

La expresión regular que denota el lenguaje es $ab^+ \vee a^+b$.

⌧ Ejemplo

Definir formalmente el autómata finito cuya función de transición es la siguiente:

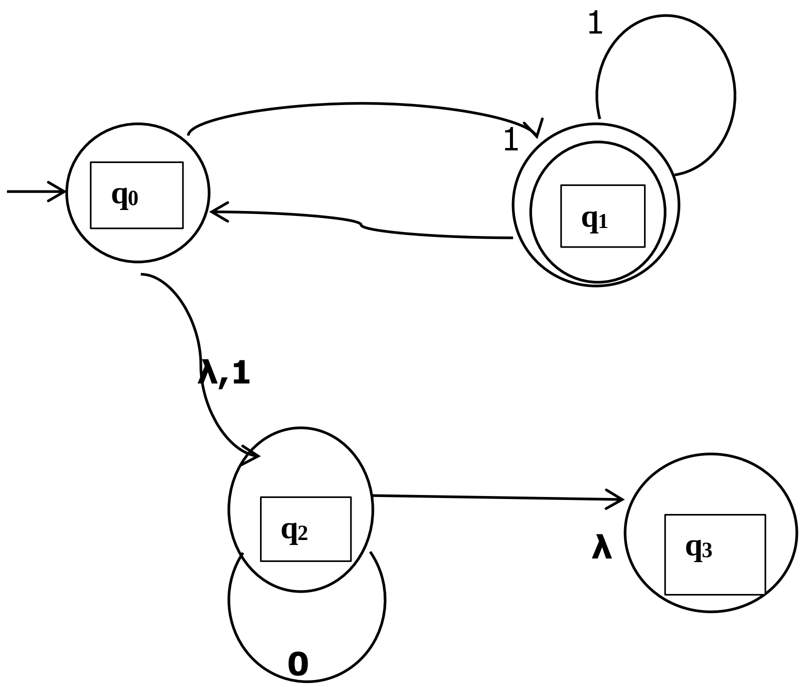
δ	**0**	**1**	**λ**
→$\mathbf{q_0}$	----	$\{q_1 , q_2\}$	q_2
*$\mathbf{q_1}$	q_0	$\{q_0 , q_1\}$	---
$\mathbf{q_2}$	q_2	--------	q_3
$\mathbf{q_3}$	----	--------	----

Realizar su diagrama de estados

¿Qué tipo de autómata finito es?

Solución

- AF = ({q_0, q_1, q_2, q_3}, {0,1}, δ, q_0, {q_1 })
- Diagrama de estados



- Se trata de un autómata finito no determinístico ya que tiene transiciones con λ.

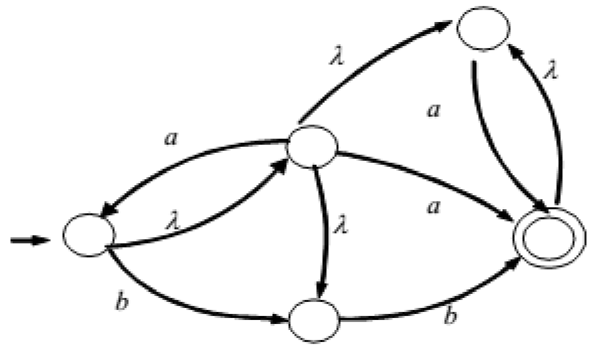
⌦ Ejercicios resueltos

1.Considere el autómata de la figura e indique cuál de las siguientes afirmaciones es falsa:

a) El autómata no es determinista

b) Existe un autómata sin transiciones lambda que reconoce el mismo lenguaje

c) El autómata reconoce un lenguaje finito



Solución:

C. Se trata de un autómata finito no determinista. Para todo autómata finito no determinista existe un autómata finito determinista (y por tanto sin transiciones lambda) que reconoce el mismo lenguaje. El lenguaje reconocido tiene un número infinito de cadenas (reconoce, entre otras, cadenas de la forma a^n).

2.Indique cuál de las afirmaciones siguientes es falsa, referida a la gramática

$$\begin{cases} S \rightarrow B1 \mid C1 \mid B0 \mid C0, \\ A \rightarrow A1 \mid D1 \mid \lambda, \\ B \rightarrow A0 \mid D0, \\ C \rightarrow B0 \mid C0, \\ D \rightarrow B1 \mid C1 \end{cases}$$

a) Es regular

b) Es independiente del contexto

c) Genera un lenguaje no regular

Solución:

C. Ya que es regular, por verificar las condiciones que definen una gramática regular, es también independiente del contexto. Toda gramática regular genera un lenguaje regular. (Atención, una gramática independiente del contexto también podría generar un lenguaje regular).

3. Indique cuál de las siguientes afirmaciones es cierta:

a) L1= L2, uno de los dos autómatas es determinista y el otro no lo es.

b) L1= L2, si consideramos que uno de los diagramas está incompleto.

c) L1≠ L2

Fig. 1

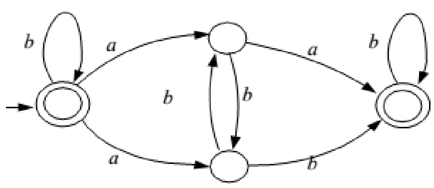
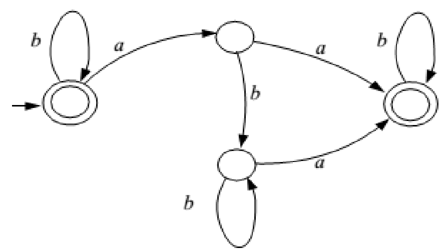


Fig. 2



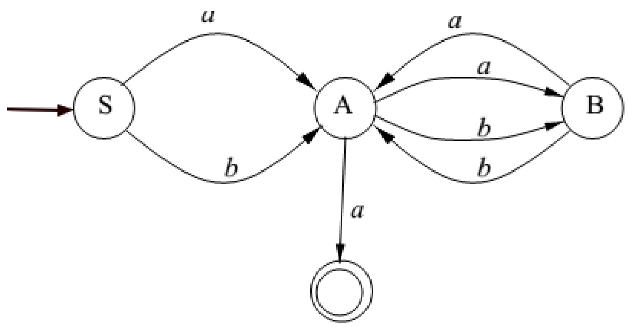
Solución:

C. El autómata de la figura 2 no acepta la cadena abb, mientras que el de la figura 1) sí la acepta.

3.Obtener un autómata finito para la gramática regular G siguiente:

1. S → aA
2. S → bA
3. A → aB
4. A → bB
5. A → a
6. B → aA
7. B → Ba

Solución:



4.Realice el diagrama correspondiente a partir de los siguientes datos

V= {a, b, c}, símbolo de entrada

Q= {q0, q1, q2, q3}, estados

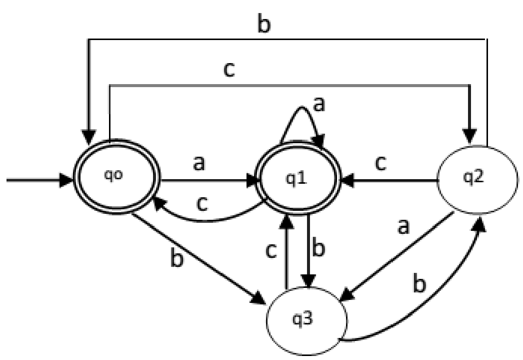
T= {q0, q1}, estados de aceptación

q0 estado inicial

La función de estado próximo **δ** definida por la siguiente tabla:

	a	b	c
q0	q1	q3	q2
q1	q1	q3	q0
q2	q3	q0	q1
q3		q2	q1

Solución:



IBLIOGRAFÍA

Kenneth. Rosen, MATEMÁTICA DISCRETA Y SUS APLICACIONES.

Cueva Lovelle, Juan Manuel, LENGUAJES, GRAMÁTICAS Y AUTÓMATAS

Departamento de Informática, Universidad de Oviedo

Para finalizar con el tema "Gramáticas-Autómatas Finitos" te proponemos que ingreses al sitio de la cátedra " https://discretaunlam.net.ar " para releer el tema.

Luego comienza a hacer los ejercicios de la guía de ejercicios para el segundo parcial. Y finaliza haciendo la autoevaluación "Gramáticas-Autómatas Finitos".

Tené en cuenta que todas las actividades que realices forman parte del seguimiento académico que hará tu tutor.