Universidad Tecnológica Nacional
Facultad Regional Buenos Aires

Secretaría de Gestión Académica

LENGUAJES, GRÁMATICAS Y AUTÓMATAS

Unidad 7

LENGUAJES, GRÁMATICAS, AUTÓMATAS

Después de un "arduo" recorrido hemos llegado al último tema del programa, que está muy relacionado con las ciencias de la computación. Vivimos en la era de las comunicaciones, y día a día nos encontramos con problemas que podemos llamar "de entrada" y "de salida". Nos referimos a acciones que realizamos, por ejemplo, cuando subimos al colectivo: las monedas que introducimos en la máquina constituyen la " entrada " y el boleto la "salida". Otro ejemplo cotidiano lo constituye la computadora: la entrada son los datos, la información, y la salida es el resultado que se obtiene después de haberlos procesado. En todos esos casos, se utilizan tipos de lenguajes mediante los cuales las máquinas pueden procesar la información de entrada y realizar las acciones para la operación de salida.
En esta unidad recurriremos a muchos de los conceptos ya estudiados para abordar un tema nuevo que son las máquinas de estado finito o circuitos secuenciales (recordemos que ya vimos circuitos no secuenciales, las redes de compuertas, en álgebra de Boole)

Para ello abordaremos primero el concepto de lenguaje formal y de gramática.

> Según el Webster´s New Coollegiate Dictionary, un lenguaje es un *conjunto de palabras y métodos para combinar palabras, que es usado y entendido por un extenso grupo de personas*[1].

Con frecuencia a lenguajes de tal tipo se los llama *lenguajes naturales* para distinguirlos de los *formales*, que son los que se aplican para modelar un lenguaje natural y "comunicarse" con la computadora. Sin embargo, ambos tipos de lenguajes tienen algunas cuestiones en común como por ejemplo un alfabeto y reglas gramaticales.

Concretando, los lenguajes pueden clasificarse en *naturales y formales*.

Los *lenguajes naturales* se caracterizan por:
- una semántica, es decir, un significado
- una sintaxis, es decir, un conjunto de reglas gramaticales.

Son ejemplos de este tipo de lenguaje, los lenguajes de seres humanos, tales como: chino, español, inglés.

Los *lenguajes formales* se caracterizan por tener reglas gramaticales preestablecidas. Son ejemplo de este tipo de lenguaje los lenguajes de programación. Nos ocuparemos en esta unidad de tratar sus particularidades.

Para entender el significado del concepto *lenguaje* vamos a definir primero otros tales como *alfabeto, letra y palabra*.

> Un *alfabeto* es un conjunto no vacío y finito de símbolos. Los símbolos del alfabeto son las letras, strings o caracteres. Suele indicarse con la letra $\mathbf{V}$.

[1], USA, 2008, ISBN 978-0-87779-916-0

Tengamos en cuenta que un alfabeto, como todo conjunto, debe estar bien definido, es decir no deben quedar dudas acerca de cuáles son los elementos que lo componen, por ejemplo si el alfabeto fuera {a, b, ab} se plantearían dudas acerca del elemento ab, ¿es un elemento del conjunto?, ¿se construyó usando a, b?; esas dudas no pueden existir.

Los elementos que pertenecen al conjunto Alfabeto se denominan letras, por ejemplo la letra e forma parte de los alfabetos que se usan en los idiomas español, italiano, inglés, entre otros.

Para el alfabeto V= {a, b}, tanto la a, como la b son letras, simplemente por ser elementos del ese conjunto.

Toda secuencia finita de letras de un alfabeto se denomina *palabra.*

Por ejemplo para el alfabeto V={0,1} -que muchas veces se denomina alfabeto binario- son palabras 1101, 001110. Podemos indicarlas w=1101, z=001110.

Al conjunto de todas las palabras que se pueden construir con las letras de un alfabeto se lo indica V*; tengamos en cuenta que si bien el alfabeto es finito, V* no lo es.

Por ejemplo: Sea V = {a, b}, el alfabeto formado por los símbolos a y b.
v = ababbaaaw = bbbbx = aaaaason palabras de V*.
Llamamos λ a la palabra nula., es decir aquella que no tiene letras.

Algo para tener en cuenta es que convenimos que la palabra nula siempre pertenece a V^*, sin importarnos cual es el alfabeto V.

Las siguientes son operaciones para realizar con palabras sobre un alfabeto V. Estas operaciones son:

Concatenación: si x e y son palabras, la concatenación, x.y es una palabra formada por los símbolos de x seguidos por los símbolos de y.

Veamos algunos ejemplos

Sea V = {0,1} alfabeto binario, sean v, w e y palabras de V*
Sean v=0111, w=1110
La palabra y= vw que leemos "v contatenado con w" queda:
y = 01111110
Tengamos en cuenta, además, que la concatenación es una operación binaria y cerrada en el conjunto V*.

¿Te animás a analizar las propiedades de la concatenación?
Si tenés alguna dificultad, podés consultar con tu tutor o recurrir al libro de la cátedra, en el capítulo 19.

Potenciación: si concatenamos n veces una cadena x, es decir..., $\underbrace{x.x.x.x.x.....x}_{n}$ obtendremos x^n.

Tengamos en cuenta que si $n = 0$, se tiene la palabra nula y si $n = 1$ la palabra que se tiene es la dada.
En realidad la potenciación es un caso particular de la concatenación, ya que concatena una sola palabra n veces.

Podés consultar una definición por recurrencia en el capítulo 19 del libro de la cátedra.

Ejemplo1

Si concatenamos 2 veces la cadena x obtendremos x^2.
Si concatenamos 3 veces la cadena x, obtendremos x^3.

Ejemplo 2

Sean v=0111w=1110
la palabra que se obtiene es $y = vw$ que leemos y. "v concatenado con w", por lo que resulta
y = 01111110

- *Reflexión, inversa o trasposición:* si la palabra x está formada por los símbolos $a_1, a_2, ..., a_n$, entonces la palabra inversa de x, que indicamos x^R, se forma invirtiendo el orden de los símbolos en la palabra;
  $x^R = a_n, ..., a_2, a_1$.

Podrías reconocer cuáles de las operaciones definidas son binarias y cuáles son unitarias? Cuando reconozcas las binarias establecé si son cerradas en V^*. Si la respuesta es afirmativa, señalá qué propiedades cumplen.
Si no podés contestar, recurrí a tu tutor o al capítulo 19 del libro de la cátedra.

Para continuar, veamos el siguiente caso:

Sea $w = abc$ entonces $w^R = cba$
Como ya dijimos V^* denota a todas las palabras del alfabeto V, es decir que cada elemento de V^* es una secuencia finita de letras del alfabeto ya que es una palabra.

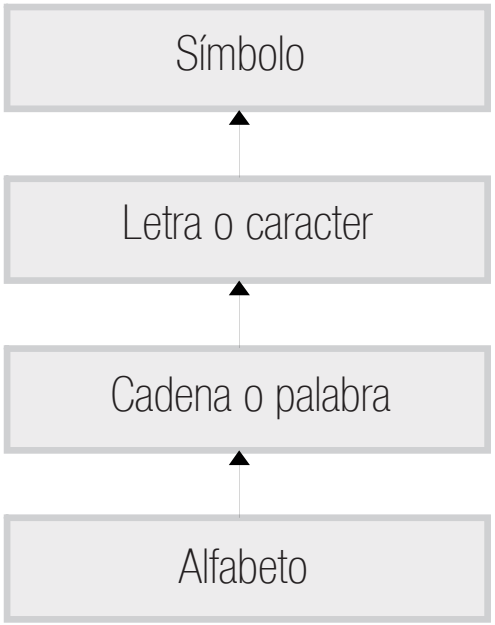
Ejemplo:

Si $V = \{a, b\}$, el alfabeto formado por los símbolos a y b se tiene que
v = ababbaaaw
w= bbbbx
z= aaaaason
son palabras de V^*

Imaginemos ahora que necesitamos enviar un mensaje que debe ser almacenado en un vector, para eso es útil conocer el número de letras que vamos a usar y por lo tanto es necesario dar una definición de longitud de una palabra, veamos entonces.
La **longitud** de una cadena w es la cantidad de símbolos que la forman.

Lo indicaremos long.w= |w|= " número de letras de w"
Tengamos en cuenta que la longitud de cualquier palabra será un número natural o cero.
Por ejemplo:
Teniendo en cuenta v=abba, w=bb, se tiene
Long v = |v| = 4
Long w = |w| = 2
Por lo tanto, si concatenamos v seguido de w, se tiene la palabra y =abbabb y entonces
Long y = long v + long w = 6
Como la palabra nula, λ, carece de caracteres o símbolos tiene longitud cero, | λ | = 0.

En síntesis, podemos esquematizar los componentes de un alfabeto de la siguiente manera:



Veamos ahora la definición de *lenguaje*

> Un lenguaje es un subconjunto de todas las palabras que se pueden formar con las letras de un alfabeto V, es decir es $L \subseteq V^*$.

Consultá la definición formal en el capítulo 19 del libro de la cátedra.

Las siguientes son operaciones que pueden realizarse con los lenguajes; en cada caso presentamos la definición y un ejemplo aclaratorio (tengamos en cuenta que un lenguaje es un conjunto y por lo tanto son válidas las operaciones para conjuntos):

- *Concatenación:* Dados dos lenguajes L_1 y L_2, su concatenación, $L_1 . L_2$ contendrá todas las palabras que se puedan formar por la concatenación de una palabra de L_1 y otra de L_2.

  Por ejemplo: Dados $L_1 = \{ \text{nana, napa, lana}\}$ y
  $L_2 = \{ \lambda , \text{nana, napa, pana, palabra, papa, pala}\}$
  $L_1 . L_2 = \{ \text{nana, napa, lana, nananana, napanana, ...}\}$

  Tengamos en cuenta que concatenamos cada palabra del primer lenguaje con cada palabra del segundo.

- *Potencia:* La potencia **i-ésima** de un lenguaje corresponde a la concatenación i veces del lenguaje en él mismo; lo indicamos

  $L^i = \underbrace{L.L.L.L....L}_{i}$ debería quedar $L^i = \underbrace{L.L.L.L....L}_{i}$

  Tengamos en cuenta que si $i=0$, obtenemos el lenguaje nulo que indicamos $\{\lambda\}$ y que si $i = 1$ se obtiene el mismo lenguaje.

  Por ejemplo: Dado $L_1 = \{ 0, 1\}$entonces
  $L_i^2 = \{ 00, 01, 10, 11 \}$

- *Reflexión, Inversión o Trasposición:* La reflexión de un lenguaje L está formada por la aplicación de la reflexión a cada una de las palabras del lenguaje; la indicamos
  $L^R = \{ x^R \text{ tal que } x \varepsilon L \}$

  Por ejemplo: Dado $L = \{ 0, 1 , 00, 10 \}$, entonces
  $L^R = \{ 0, 1, 00, 01 \}$

Recordemos las siguientes operaciones entre conjuntos, que ahora aplicaremos a los lenguajes.

- *Unión:* Dados dos lenguajes L_1 y L_2, su unión $L_1 \cup L_2$ contendrá todas las palabras que pertenezcan a cualquiera de los dos lenguajes,
  $L_1 \cup L_2 = \{ x \text{ tal que } x \varepsilon L_1 \vee x \varepsilon L_2 \}$

  Ejemplo: Dados $L_1 = \{ \text{nana, napa, lana}\}$ y
  $L_2 = \{ \lambda, \text{nana, napa, pana, palabra, papa, pala}\}$
  $L_1 \cup L_2 = \{ \lambda, \text{nana, napa, lana, pana, palabra, papa, pala} \}$

- *Intersección:* Dados dos lenguajes L_1 y L_2, su intersección
  $L_1 \cap L_2$ contendrá todas las palabras que pertenezcan a los dos lenguajes;
  $L_1 \cap L_2 = \{ x \text{ tal que } x \varepsilon L_1 \text{ y } x \varepsilon L_2 \}$
  Ejemplo: Dados $L_1 = \{ \text{nana, napa, lana}\}$ y

L_2 = { λ, nana, napa, pana, palabra, papa, pala}

$L_1 \cap L_2$ = { nana, napa }

- *Diferencia:* si
  L_1 y L_2 son lenguajes la resta de L_1 y L_2,
  L_1 - L_2, contendrá todas las palabras que pertenezcan a L_1 y no pertenezcan a L_2 ,
  L_1 - L_2 = { x tal que x ε L_1 y x ∉ L_2 }

  Ejemplo: Dados L_1 = { nana, napa, lana} y
  L_2 = { λ, nana, napa, pana, palabra, papa, pala}
  L_1 - L_2 = { napa, lana }
  L_2 - L_1 = { λ, pana, palabra, papa, pala

Recurre al libro de la cátedra en el capítulo 19 y trata de descubrir el conjunto en el que la concatenación de lenguajes y por lo tanto la potenciación es una operación cerrada y binaria. Discute con el tutor las propiedades.

Veamos ahora las siguientes definiciones que las aplicaremos al desarrollo de autómatas finitos

- *Clausura de Kleene:* sea V un alfabeto, sea N el conjunto de los números naturales, sea n ε N U {0} y sea L un lenguaje de V* entonces:

  $L^* = L^0 \cup L^1 \cup L^2 \cup L^3 \cup .... \cup L$ es la clausura de Kleene del lenguaje L.

$$L^* = \bigcup_{i=0}^{n} L$$

- *Clausura Positiva:* sea V un alfabeto, sea N el conjunto de los números naturales, sea n ε N y sea L un lenguaje de V* entonces: $L^+ = L^1 \cup L^2 \cup L^3 \cup .... \cup L$ es la clausura de positiva del lenguaje L.

$$L^+ = \bigcup_{i=1}^{n} L$$

Ahora estás en condiciones de resolver los siguientes ejercicios que te proponemos. Si te surge alguna pregunta o duda no dejes de consultar a tu tutor.

1. Sea V = {0,1}, indicá la longitud de las siguientes palabras del alfabeto V:
   a = 01011; b = 01; c = 00; d = λ

2. Sea $V = \{a, b\}$, el alfabeto formado por los símbolos **a** y **b**, indicá cuáles de las siguientes son palabras de V^*

   a). $v = ababbaca$

   b). $w = z$

   c). $z = a$

3. ¿Un Lenguaje formal puede ser infinito?

4. Indicá si las siguientes opciones son verdaderas o falsas: sea $V = \{1, 2, 3\}$

   a). $a=111 \in L(V)$

   b). $b=312 \in L(V)$

   c). $c=114 \in L(V)$

5. Sean $V = \{1, 2, 3\}$, $a=111 \in V$; $b=312 \in V$
   Concatená **ab** y **ba.**
   Indicá v^R y w^R, con $v = ab$ y $w = ba$.

6. ¿Es conmutativa la operación concatenación?

7. Sea $V = \{0, 1\}$, $w = 0101010$, hallá **w** con potencia **0, 1, 2** y **3**, establecé las longitudes de dichas palabras.

8. Probá que la concatenación en V^* es un semigrupo.

9. Probá que: $long\ w^a = a\ long\ w$ utilizando inducción matemática en **a**.

10. Sea $w \in V^*$, indicá los valores de:

    a). $\left(w^R\right)^R$

    b). λ^R

11. Dados los lenguajes

    $L_1 = \{abc, aa, aba\}$ y $L_2 = \{bbb, aaaaaaaa, abc\}$
    hallá L_1^* y L_2^*, L_1^+ y L_2^+.

12. Dados los lenguajes

    $L_1 = \{10, 00\}$ y $L_2 = \{10, 00, 0000\}$ indicá si ¿$L_1^* = L_2^*$? y ¿$L_1^+ = L_2^+$?

13. Sea el lenguaje $L = \{111, 000\}$ mostrá que la concatenación es cerrada en L^*.

Si tenés dudas sobre la resolución de estos ejercicios, planteáselas a tu tutor.

Sintetizando, hasta aquí:

- *Definimos los conceptos de alfabeto, letra y palabra*
- *Vimos que un alfabeto debe ser finito pero que el conjunto de todas las palabras que se pueden conseguir con las letras de cada alfabeto siempre es infinito*
- *Dimos el concepto de longitud de una palabra y caracterizamos a la palabra nula que es la que no tiene letras*
- *Aprendimos a operar con palabras con operaciones binarias (concatenación) y unitarias (inversión); notamos que la potenciación de palabras es un caso particular de la concatenación*
- *Vimos el concepto informal y formal de Lenguaje*
- *Aprendimos a operar con lenguajes y notamos que se opera con cada palabra de cada lenguaje.*
- *Vimos que las operaciones entre conjuntos son válidas en este caso y por lo tanto cumplen las mismas propiedades que en el caso de palabras*
- *Finalmente definimos las clausuras positivas y de Kleene de un lenguaje*

Gramáticas

Las gramáticas son descripciones de las sentencias de los lenguajes, establecen la estructura que deben tener las sentencias para que estén bien formadas y para que pueda entenderse su significado.
Las gramáticas formales son descripciones estructurales de las sentencias de los lenguajes, tanto formales (lenguajes de programación), como naturales (humanos). En este último caso, la gramática se ocupa de la sintaxis (es decir la forma que deben tener las sentencias)..
Las gramáticas permiten describir de forma intencional a los lenguajes; proporcionan reglas para la estructura de las frases y su significado.

Informalmente podemos decir que una gramática es un modelo matemático que consta de:

- un *alfabeto*, llamado alfabeto de elementos terminales que representa el conjunto de letras que tendrán las palabras del lenguaje que genera esa gramática
- un conjunto de *símbolos especiales,* denominados no terminales, que son elementos auxiliares y permiten representar estados intermedios antes de llegar al de la generación de las palabras del lenguaje.
- un símbolo inicial del que se partirá para la obtención de cualquiera de las palabras del lenguaje, denominado cabeza del lenguaje, que es uno de los elementos no terminales.
- un conjunto de producciones o reglas gramaticales que permitirán realizar las transformaciones desde los símbolos no terminales a las palabras del lenguaje.

Formalizando lo dicho hasta aquí:

Se define a la Gramática formal G, como la cuádrupla:
$G = (Vn; Vt ; P ; s)$ donde:

- V_n es un conjunto de elementos llamados no terminales, que suele llamarse vocabulario de elementos no terminales.
- V_t es un conjunto de elementos llamados terminales, suele denominarse vocabulario de elementos terminales.
- P es un conjunto de producciones (reglas de sustitución) y S es un elemento de V_n llamado símbolo inicial.

Algunas observaciones que tendremos que tener en cuenta son las siguientes:

1. El lenguaje generado por una gramática G se denomina $L(G)$.
2. Si la gramática G genera un lenguaje L se indica

$$G \Rightarrow L(G)$$

3. Ninguna regla gramatical puede comenzar con la palabra nula, λ
4. Dos gramáticas G_1 y G_2 son equivalentes si

$L(G_1) = L(G_2)$; es decir, si generan el mismo lenguaje.

5. El proceso para obtener las palabras del lenguaje que genera la gramática se llama derivación

5.1. Se comienza con el símbolo inicial **s.**
5.2. Se aplican las producciones de **P** al símbolo inicial **s** hasta obtener sólo elementos terminales.
5.3. Cualquier hilera que se consigue usando el 5.2. es un elemento del lenguaje **L(G).**

El que sigue es un ejemplo de gramática:

Ejemplo 1
Sea **G** una gramática en la cual:

$V_t = \{a, b\}$
$V_n = \{S, A, B\}$, donde **S** es el símbolo inicial.

Las reglas gramaticales o producciones son

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

Pueden escribirse de la forma siguiente:

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

Aplicando las reglas gramaticales y teniendo en cuenta que esta gramática genera palabras con las letras **a, b,** debemos plantearnos como objetivo eliminar las letras **S, A, B.** Tenemos que comenzar con **S**, eso no podemos elegirlo, pero si podemos seleccionar cualquiera de las opciones que **S** nos ofrece, por ejemplo:

$S \rightarrow aB$

A continuación, para eliminar B elegimos $B \rightarrow b$, reemplazamos en la expresión anterior y queda:

$S \rightarrow aB \rightarrow ab$ que es una palabra del lenguaje ya que ambas letras pertenecen al vocabulario terminal.

Veamos otra palabra (si hay que reemplazar una letra por más de una conviene poner un paréntesis),

$S \rightarrow aB \rightarrow a(bS) \rightarrow ab(bA) \rightarrow abb(bAA) \rightarrow abbb(AA) \rightarrow abbb(aa)$

Finalmente la palabra es $w=abbbaa$

¿Qué hicimos?¿ podés justificarlo? Si no estás seguro, consultalo con el tutor.

Recordemos que, cuando vimos lenguajes, señalamos que podían ser finitos o infinitos. Si en el caso anterior miramos las producciones, es decir las reglas gramaticales, vemos que tanto con A como con B volvemos al comienzo, es decir a S. Como eso lo podemos hacer tantas veces como queramos, resulta que el lenguaje que genera esta gramática es infinito.
Puede ocurrir que nos pidan que describamos ese lenguaje ya que es imposible listar todas sus palabras.
Si siguiéramos con el procedimiento podríamos observar que los enunciados del lenguaje son todas las cadenas de letras a y b en las que el número de letras a es igual al número de letras b.
Hilando más fino podemos decir que el no terminal A representa el conjunto de cadenas en las que el número de letras a es uno más que el número de letras b, y el no terminal B representa el conjunto de cadenas en las que el número de letras b es uno más que el número de letras a.

Una forma práctica de representar las derivaciones son los árboles de derivación que se utilizan en la construcción de compiladores para representar el análisis sintáctico de los programas fuente y sirven de base para la generación de un código.

Sin embargo, no siempre se pueden usar. Ya veremos cuando es posible usarlos y cuando no.

En los árboles de derivación:

- La cabeza del lenguaje se representa en la raíz del árbol
- Los nodos hojas del árbol son símbolos terminales de la gramática

Las derivaciones se representan creando tanto los sucesores del símbolo no terminal de la izquierda de las producciones como los símbolos (terminales y no terminales) aparezcan en la parte derecha de las producciones

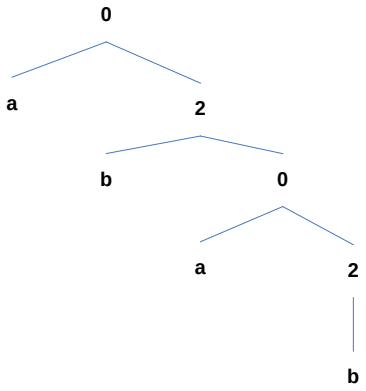
Veamos un ejemplo de árbol de derivación para la gramática

Ejemplo 2
Sea $G = (\{0,1,2\}; \{a, b\}; P; 0)$ con las siguientes producciones:
$0 \rightarrow a2 / b1$
$1 \rightarrow a / a0 / b11$
$2 \rightarrow b / b0 / a22$

Algunas de las palabras del lenguaje $L(G)$ son:
$0 \rightarrow a2 \rightarrow ab \in L(G)$

El árbol de derivación es el siguiente: $0 \rightarrow a2 \rightarrow ab0 \rightarrow aba2 \rightarrow abab \in L(G)$

Cuyo árbol de derivación es:



Tengamos en cuenta que para cada palabra hay un árbol de derivación

Finalmente podemos preguntarnos cuando una palabra no pertenece al lenguaje generado por una gramática, esto sucede cuando al aplicar las reglas gramaticales no se puede obtener la palabra en cuestión.
Si nos remitimos al ejemplo 1, donde las palabras tenían el mismo número de a que de b, la palabra $w = aabbb$ no podríamos encontrarla

La siguiente clasificación de las gramáticas dada por el lingüista Noam Chomsky nos resultará de gran utilidad para reconocer las gramáticas formales y usarlas para resolver distintos tipos de problemas.

Clasificación de gramáticas:

Noam Chomsky definió cuatro tipos de gramáticas formales, en función de las producciones, es decir de las reglas gramaticales.
Esta clasificación comienza con las más generales para finalizar en las que presentan más restricciones, siempre teniendo en cuenta las producciones.

A continuación daremos la clasificación, para eso consideraremos una producción genérica $x \rightarrow y$, donde llamaremos "parte izquierda x" y "parte derecha y"

- Gramáticas de Tipo 0 (sin restricciones):

En la parte izquierda tiene que haber al menos un símbolo no terminal. Respecto a las partes derechas de sus producciones no hay ningún tipo de restricción.

Ejemplos:

1. $S \rightarrow aSBC \mid aBC$
   $CB \rightarrow BC$
   $aB \rightarrow ab$
   $bB \rightarrow bb$
   $bC \rightarrow bc$
   $cC \rightarrow cc$

2. G = ({0, 1}; {A, B, S}, S, P), donde

P: S $\rightarrow$ A0
A0 $\rightarrow$ 1B1
1A $\rightarrow$ 0B0
B $\rightarrow \lambda$
B $\rightarrow$ 1
B $\rightarrow$ 0

El lenguaje generado por esta gramática es:

L(G) = {11, 101, 111}

- Gramáticas de Tipo I (dependientes o sensibles del contexto):

Se denominan gramáticas *dependientes del contexto* porque en ellas se tiene en cuenta que es lo que viene antes y después del símbolo que se está sustituyendo.
Este tipo de gramática consiste en que las derivaciones producidas por aplicación de sus reglas cumplen la condición de que las palabras siempre tienen longitud mayor o igual que las anteriores en la derivación, es decir
| x | ≤ | y |, de donde no generan la palabra nula.
Se llaman dependientes del contexto porque si hay una producción de la forma aXy donde debe ser reemplazado X: debe hacerse de la forma siguiente aTy, donde T es el reemplazo de X que respeta a lo que "rodea" a X.

Ejemplos:

1. S $\rightarrow$ abc | aAbc
   Ab $\rightarrow$ bA
   Ac $\rightarrow$ Bbcc
   bB $\rightarrow$ Bb
   aB $\rightarrow$ aa | aaA

2. G = ({0, 1}; {A, B}, A, P), donde
   P: A $\rightarrow$ 1B1
   A $\rightarrow$ 11
   1B1 $\rightarrow$ 101
   1B1 $\rightarrow$ 111

El lenguaje generado por esta gramática es:

L(G) = {11, 101, 111}

- Gramáticas de Tipo II (independientes del contexto)

Estas gramáticas se denominan de *contexto libre*, porque a la hora de transformar una palabra en otra, el símbolo no terminal que se transforma no depende de los que estén a la izquierda o derecha.

En estas gramáticas, la parte izquierda (x) de las producciones sólo pueden tener un símbolo no terminal, es decir $|x|=1$

Ejemplos:

1. $S \rightarrow aSb \mid ab$

2. $L = \{a^k b^k$ tal que $k \geq 1\}$ es un lenguaje de tipo 2, pues puede especificarse mediante la gramática de tipo 2:

$$A \rightarrow aAb$$
$$A \rightarrow ab$$

3. Sea $G = (\{0, 1\}, \{A, B\}, A, P)$, donde
$P = \{(A \rightarrow 1B1), (A \rightarrow 11), (B \rightarrow 1), (B \rightarrow 0)\}$
Esta gramática G, genera a $L(G) = \{11, 101, 111\}$

- Gramáticas de Tipo III (regulares o lineales)

Una gramática es *regular o tipo 3* si las producciones son de la forma $x \rightarrow y$ donde:

- x es un único símbolo no terminal $(x \in V_n)$
- y es concatenación de dos símbolos siendo uno de ellos terminal (V_t) o,
- y es un único símbolo terminal o
- y es la palabra nula

Algo para tener en cuenta es que si una producción es de la forma $V_n V_t$ en la misma gramática no puede haber una regla $V_t V_n$

Ejemplos:

1. $A \rightarrow aB \mid a \mid l$

2. Sea la gramática $G = (\{a, b, c\}; \{0, 1\}; P; a)$ con P dada por:

$$a \rightarrow b1 \mid c1$$
$$b \rightarrow b0 \mid l$$
$$c \rightarrow l \mid 0$$

3. Sea la gramática $G = (\{a, b, c\}; \{0, 1\}; P; a)$ con P dada por:

$$a \rightarrow 1b \mid 1c$$
$$b \rightarrow 0b \mid \lambda$$
$$c \rightarrow \lambda \mid 0$$

4. Sea la gramática $G = (\{a, b, c\}; \{0, 1\}; P; a)$ con P dada por:

$a \rightarrow 1b \mid 1c$
$b \rightarrow b0 \mid \lambda$
$c \rightarrow \lambda \mid 0$

No es una gramática tipo 3 ya que hay una producción 1b, de la forma $V_t\ V_n$ y otra producción b0 que es de la forma $V_n V_t$

Podemos sintetizar la clasificación con el siguiente cuadro:

Gramática Tipo 0
- ✓**Sin restricciones**
- ✓**Genera lenguajes recursivamente numerables**

Gramática Tipo I
- ✓**Dependientes del contexto (se tiene en cuenta lo que viene antes y después del símbolo que se sustituye).**
- ✓**Genera lenguajes sensibles (o dependientes) al contexto**
- ✓**Las producciones son del tipo: α → β donde α, β Є {Vn ∪ Vt}**
- ✓**Las producciones son no-contractivas: | α | ≤ | β |**
- ✓**Nunca genera a la palabra nula**

Gramática Tipo II
- ✓**Libres de contexto**
- ✓**Genera lenguajes libres al contexto**
- ✓**El lado izquierdo debe consistir en un solo no terminal**
- ✓**No hay restricciones al lado derecho**

Gramática Tipo III
- ✓**Regulares**
- ✓**Genera lenguajes regulares**
- ✓**El lado izquierdo debe consistir en un solo no terminal**
- ✓**El lado derecho debe ser un terminal seguido de un no terminal, o un solo terminal o la cadena vacía**

Sintetizando lo que hemos visto hasta ahora:

- *Las gramáticas nos dan las reglas para ir formando las palabras de un lenguaje*
- *Formalmente son cuádruplas G = (Vn; Vt; P, s)*
- *Para obtener palabras se deben ir aplicando las producciones o reglas grama ticales.*
- *Aveces se pueden utilizar árboles de derivación.*
- *Las gramáticas se clasifican en 4 tipos de acuerdo a sus producciones: tipo 0 o irrestrictas, tipo 1 o sensibles al contexto, tipo 2 o independientes del contexto y tipo 3 o regulares.*

En esta asignatura nos dedicaremos principalmente a los lenguajes regulares o tipo 3, es decir los que pueden ser generados por gramáticas tipo 3, y que como veremos en la sección siguiente van a ser reconocidos por máquinas abstractas llamadas autómatas finitos.
Sin embargo, para que tengas una idea, la mayoría de los lenguajes de programación son tipo 2 o independientes del contexto. Ellos son reconocidos por otro tipo de máquinas llamadas autómatas de pila.
Pero para poder comprender bien dichos lenguajes y dichas máquinas, primero es necesario conocer bien los lenguajes regulares y los autómatas finitos, que son las máquinas más simples.

A continuación los ejercicios resueltos te ayudarán a afianzar el tema:

1. Obtené las derivaciones de las palabras 002 y 0001 a partir de la siguiente gramática:

$G = (\{0, 1, 2\};\{A, B\}; P; A\})$
P:
$A \rightarrow 0B$
$A \rightarrow 2$
$B \rightarrow 0A$
$B \rightarrow 1$

Describí el árbol de derivación y obtené el lenguaje que genera.

Las palabras se pueden obtener de las siguientes derivaciones:

002: $A \rightarrow 0B \rightarrow 00A \rightarrow 002$
0001: $A \rightarrow 0B \rightarrow 000B \rightarrow 0001$

Los árboles de derivación aparecen en la figura siguiente:

Para obtener el lenguaje, habrá que analizar qué palabras pueden derivarse desde el axioma. Así que se pueden obtener las siguientes derivaciones:

$A \rightarrow 2$
$A \rightarrow 0B \rightarrow 01$
$A \rightarrow 0B \rightarrow 00A \rightarrow 002$
$A \rightarrow 0B \rightarrow 00A \rightarrow 000B \rightarrow 0001$
$A \rightarrow 0B \rightarrow 00A \rightarrow 000B \rightarrow 0000A \rightarrow 00002$

Por lo tanto, puede aparecer un 2 precedido de un número par de 0s ó un 1 precedido por un número impar de 1s.

Si se define un lenguaje L_1 como:

$L_1 = \{ 0^n 2 / n = 2.x \}$

y un lenguaje $L_2 = \{ 0^a 1 / a = 2.y + 1 \}$

entonces, el lenguaje generado por la gramática se puede definir como
$L(G) = L_1 \cup L_2$, o lo que es lo mismo: $L(G) = \{ 0^n 2 \cup 0^a 1 / n = 2.x a = 2.y +1 \}$

2. Dados los siguientes alfabetos:

$V_1 = \{ 1, 2, 3, 4, 5, 6, 7, 8 \}$
$V_2 = \{ a, b, c, d, e, f, g, h \}$

Y los lenguajes:

$L_1(V_1) = \{ x \mid x \in V_1 \}$ y $L_2(V_2) = \{ x \mid x \in V_2 \}$

Definir los lenguajes $L_1 \cup L_2$, $L_1 . L_2$ y $(L_1 . L_2)^2$

En este caso, una numeración muy utilizada para los tableros de ajedrez es numerar las filas del *1* al *8* y las columnas de la *a* a la *h*, con lo que el lenguaje resultante de $L_1 . L_2$ representaría todas las casillas de ajedrez.

$$(L_1 . L_2)^2 = \{ x^2 = x.x \mid x \in (L_1 . L_2) \} =$$
$$= \{ 1a1a, \ldots , 1a8a, 1a1b, \ldots , 1b1a, \ldots , 8h8h \}$$

El número de palabras del lenguaje resultante sería $| (L_1 . L_2)^2 | = 8^4$.

3. Determiná si la gramática dada es sensible al contexto, libre del contexto, regular o bien ninguna de éstas:

a. $T = \{ a, b \}$, $N = \{ \sigma^*, A \}$, con composiciones:
$\sigma^* \rightarrow b \sigma^*$, $\sigma^* \rightarrow aA$, $a \rightarrow a \sigma^*$,
$A \rightarrow bA$, $A \rightarrow a$, $\sigma^* \rightarrow b$,

y símbolo inicial σ^*.

Solución: regular, libre de contexto, sensible de contexto. Es importante siempre dar el último tipo alcanzado. Es decir en este caso no podemos dejar de decir que es regular

b. $T = \{ a, b \}$, $N = \{ \sigma^*, A, B \}$, con composiciones:
$\sigma^* \rightarrow b \sigma^*$, $\sigma^* \rightarrow AAB$, $Aa \rightarrow ABa$, $a \rightarrow aA$,
$Bb \rightarrow ABb$, $AB \rightarrow ABB$, $B \rightarrow b$,

y símbolo inicial σ^*.

Solución: es tipo I o sensible al contexto

c. <S> → b <S> | a<A> | a
<A> →a <S> | b<B>
<B> → b <A> | a<S> | b,

con símbolo inicial <S>.

Solución: regular, libre de contexto, sensible de contexto.Lo importante es que es tipo 2 o libre de contexto

Autómatas:

Lenguajes, Gramáticas Regulares y Autómatas

A continuación se demostrará que una gramática regular y un autómata de estado finitos son, en esencia, lo mismo pues cualquiera de ellos es una especificación de un lenguaje regular.

Comencemos por el concepto de lenguaje regular.

Podemos decir, informalmente, que un lenguaje es regular si existe una gramática regular que lo genere.

Una definición formal puede ser dada en forma recursiva, como se da a continuación

1. $\varnothing$ es un lenguaje regular.
2. $\{ \lambda \}$ es un lenguaje regular.
3. Si V es un alfabeto y $a \in V$ entonces $\{a\}$ es un lenguaje regular.
4. Si L_1 y L_2 son dos lenguajes regulares entonces:

   $L_1 \cup L_2$, $L_1 L_2$ y L_1^* son lenguajes regulares.

5. Ningún otro lenguaje sobre V es regular.

e

Veamos los siguientes ejemplos:

1. Sea $V = \{a, b\}$ los siguientes lenguajes son regulares:

$L_1 = \{ ab, ba \}$
$L_2 = \{ bb \}$
$L_1 \cup L_2 = \{ ab, ba, bb \}$
$L_3 = V^*$

2. Todos los números binarios que comienzan con un número par de 1s y terminan con un 0, es un lenguaje regular infinito sobre el alfabeto $\{0, 1\}$

$L = \{ 1^{2n} 0 \text{ con } n \geq 0 \}$

Algo para tener en cuenta:

Si un lenguaje formal es finito entonces es un Lenguaje Regular.
Un lenguaje es regular si puede ser reconocido por un Autómata Finito.

A continuación nos referiremos a las Expresiones Regulares que nos facilitarán la manera de describir un lenguaje regular y que se construyen utilizando los caracteres del alfabeto sobre el cual se define el lenguaje.

Veamos la definición formal, que la daremos por recurrencia:

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

λ (palabra nula): es una expresión regular.

Si $a \in A$ entonces a es una expresión regular.
Si x e y son expresiones regulares entonces xy 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.

Tengamos en cuenta lo siguiente: asociado a cada expresión regular sobre el alfabeto A hay un subconjunto de A^*, que se llama subconjunto o lenguaje regular.

Los siguientes Ejemplos: nos ayudarán a aclarar ideas

1. La Expresión Regular a*ba*ba representa al lenguaje de todas las palabras sobre el alfabeto {a, b} con 2 b y una a, comenzando o no con aes.
   Son palabras de este lenguaje: bba, abba, aabaaba....

2. Sea el alfabeto $A = \{0, 1\}$, la Expresión Regular $(1 \vee 0)\ 1^*$ representa al lenguaje de los números binarios que comienzan con un 1 ó un 0, el que puede o no ser seguido por 1 ó más 1s.
   Son palabras de este lenguaje: 1, 0, 11, 01, 111, 011, ...

Un Lenguaje Regular debe poder ser reconocido por un autómata finito.

Un autómata finito es una herramienta abstracta que se utiliza para reconocer un determinado Lenguaje Regular.
Es un modelo matemático de un sistema que recibe una cadena formada por caracteres de un determinado alfabeto y determina si esa cadena pertenece o no al lenguaje que reconoce.

Una máquina procesadora de información es un dispositivo que recibe un conjunto de señales de entrada que produce en correspondencia un conjunto de señales de salida.

Veamos los siguientes casos que seguramente nos ayudarán a comprender el tema.

e

Por ejemplo, podemos considerar que una lámpara de mesa es una máquina procesadora de información: la señal de entrada es la posición ENCENDIDO o la posición APAGADO del interruptor y la señal de salida es la ILUMINACIÓN o la OSCURIDAD.

Otro ejemplo de máquina procesadora de información es un sumador, cuyas señales de entrada son dos números decimales y la señal de salida, su suma.

Generalmente, las señales de entrada para una máquina procesadora de información cambian con el tiempo. Entonces en este caso, las señales de salida también variarán con el tiempo, en la forma correspondiente. Es decir, que una máquina procesadora de información recibe una serie (temporal) de señales de entrada y produce una correspondiente serie (temporal) de señales de salida.

Otro ejemplo es el de una máquina expendedora, que también es una máquina procesadora de información donde las señales de entrada son las monedas depositadas y la selección de la mercadería, y las señales de salida son las mercaderías y posiblemente el vuelto.

Advirtamos que existe una diferencia entre las máquinas citadas en los ejemplos anteriores. En el caso de la lámpara de mesa, siempre que la señal de entrada es ENCENDIDO, la señal de salida es ILUMINACIÓN, y siempre que la señal de entrada sea APAGADO, la señal de salida es OSCURIDAD. Esto indica que la señal de salida depende en cualquier momento solamente de la señal activada en ese instante y no de las señales de entrada anteriores a dicho instante.

En cambio, en el caso de la máquina expendedora, la señal de salida obtenida en cualquier instante dependerá no sólo de la señal de entrada dada en ese instante, sino además, de las señales de entrada precedentes.

Por ejemplo tomemos tres señales de entrada sucesivas:

1 peso	1 peso	1 peso

las correspondientes señales de salida son:

NADA	NADA	1 GASEOSA

La máquina expendedora es capaz de recordar la cantidad total que ha sido depositado.

Dividimos a los autómatas en dos clases, con memoria y sin memoria. Es muy importante tener en cuenta que:
Para una máquina sin memoria, su salida en cualquier instante, sólo depende de la entrada en tal instante.
y que para una máquina con memoria, su salida en cualquier instante depende de la entrada en dicho instante como de las entradas en instantes previos, debido a que la máquina puede recordar que "sucedió en el pasado".

Un *estado* representa un resumen de la historia pasada de una máquina. Por lo que, el estado de la máquina junto con las señales de entrada en un instante determinado, establecerán las señales de salida correspondientes a dicho instante.

Recordemos el concepto de autómata finito: una máquina con un número finito de estados, en cambio, una máquina con un número infinito de estados se llama autómata infinito.
Nosotros, como ya dijimos, nos ocuparemos de los autómatas finitos. Vamos a definirlos formalmente:

Una máquina con un número finito de estados se llama máquina de estado finito. Una máquina de estado finito es la 5-upla $M = (Q; V; \delta; q; F)$ donde:

- $Q = \{q_1, q_2, ..., q_n\}$ es el conjunto finito de estados de la máquina
- V es el alfabeto de entrada.
- $\delta: Q \times V \rightarrow Q$ es la relación de transición
- $q \in Q$ y se dice estado inicial.
- $\varnothing \neq F \subseteq Q$ es el conjunto de estados finales.

De la definición vamos a destacar que:

- La relación de transición $\delta: Q \times V \rightarrow Q$ es tal que $\delta(q;x) = f_{x(q)}$
- $\delta(q;x) = f_{x(q)}$ describe, para cada elemento de V, el efecto que esa señal de entrada tiene en los estados de la máquina.
- Indicamos $F = \{ f_x : Q \rightarrow Q$ con $x \in V\}$
- El estado inicial q es único.

Dentro de los autómatas finitos, encontraremos determinísticos y no determinísticos.
Los autómatas finitos determinísticos transitan entre estados de un conjunto de estados según reciben los símbolos que forman la palabra de entrada. Hay tres tipos de estados:

- Estado inicial, que permite empezar la ejecución del autómata.
- Estados finales, que permiten realizar la salida de aceptación de la palabra de entrada en el caso de que no haya más símbolos en la entrada.
- Estados intermedios.

Los autómatas finitos no determinísticos son aquellos en los que puede existir más de un transición por cada par (estado, entrada) o ninguna transición por cada par, de forma que, en cada momento, el autómata puede realizar varias transiciones diferentes entre las que deberá optar o no poder realizar ninguna. Otro rasgo que los diferencia de los determinísticos es que pueden realizar transiciones de un estado a otro sin leer ningún símbolo de entrada, mediante las denominadas transiciones λ.
Detengámonos ahora en los Autómatas Finitos Determinísticos.

Autómatas finitos determinísticos:

Un autómata finito determinístico (AFN) es una máquina de estado finito, que se define como: la 5-upla $M = (Q; V; \delta; q; F)$ donde:

- $Q = \{q_1, q_2, ..., q_n\}$ es el conjunto finito de *estados* de la máquina.
- V es el alfabeto de entrada.
- $\delta: Q \times V \rightarrow Q$ es la *relación de transición.*
- $q \rightarrow Q$ y se dice *estado inicial.*
- $\varnothing \neq F \subseteq Q$ es el conjunto de *estados finales.*

Algunas observaciones a considerar:

- Para cualquier estado, la lectura de una letra determina, sin ambigüedades, el estado de llegada.
- Para todo $(q; a) \in Q \times V \rightarrow \delta(q; a) = p$ si y sólo si la máquina puede pasar del estado q al estado p al "leer" el símbolo a.
- Para cada estado en un diagrama de transición debe existir a lo sumo una arista etiquetada para cada letra del alfabeto V.
- Ninguna arista está etiquetada con λ.
- Un A.F.D. es completo si cada estado tiene una transición por cada letra del alfabeto.

Veamos algunos de los ejemplos de Máquinas de Estado Finito como reconocedoras de Lenguaje.

1) El autómata $A = (\{q_o, q_1\}; \{0, 1\}; \delta; q_o; \{q_o\})$, donde δ se define como:

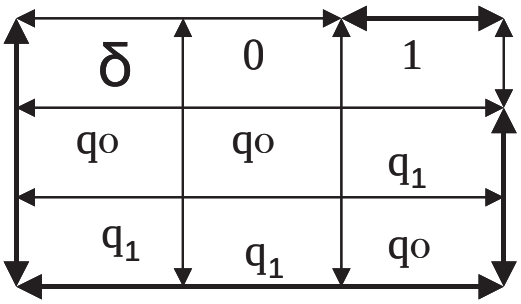
$\delta(q_o;0) = q_o$
$\delta(qo;1) = q_1$
$\delta(q_1;0) = q_1$
$\delta(q_1;1) = q_o.$

Si la palabra de entrada es 0110, el autómata pasará por los siguientes estados:

$0 \quad 1 \quad 1 \quad 0$
$q_o \rightarrow q_o \rightarrow q_1 \rightarrow q_o \rightarrow qo$

Vamos a armar la tabla de transiciones:

- En las filas estarán los estados $q \in Q$.
- El estado inicial se precederá del símbolo $\rightarrow$
- Cada estado final se precederá del símbolo *.
- En las columnas estarán los símbolos de entrada $a \in V$.
- En la posición $(q;a)$ estará el estado que determine $\delta(q;a)$.



Respecto del diagrama de transiciones, la siguiente es una de las convenciones:

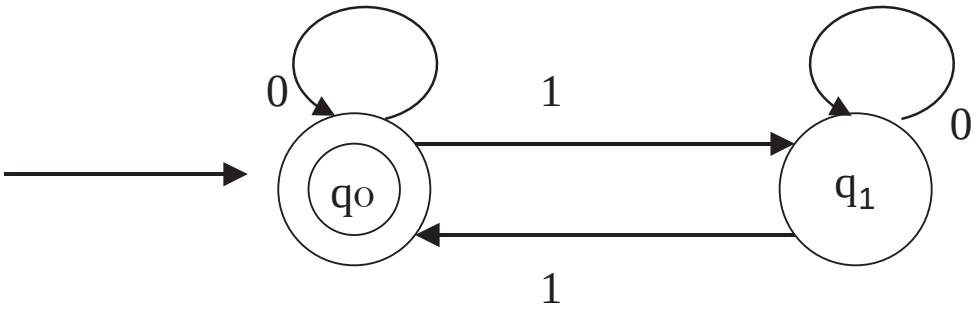
* En los nodos estarán los estados

* El estado inicial tendrá un arco entrante no etiquetado

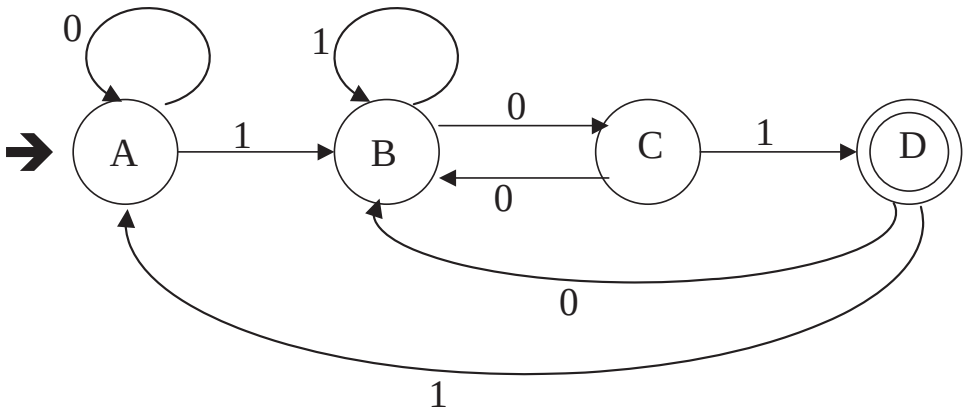
* Los estados finales estarán rodeados del doble círculo

* Habrá un arco etiquetado con a entre el nodo q_i y el nodo q_j si $\delta(q_i; a) = q_j$

y entonces el autómata A nos queda de la siguiente manera:



2).

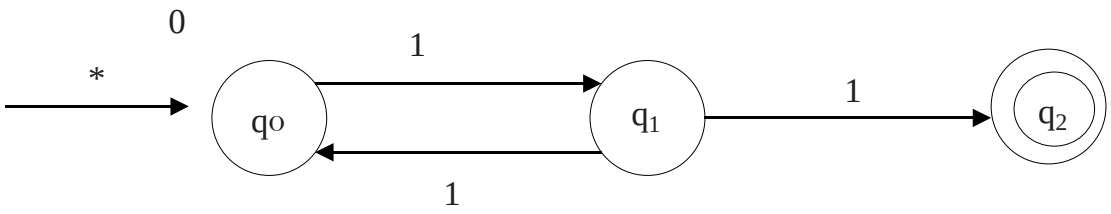


Esta máquina de estado finito es un Autómata Finito determinístico, ya que cumple con la definición del mismo. De cada estado {A, B, C, D} por cada letra del alfabeto (en este caso 0 o 1) existe una única transición hacia otro estado.

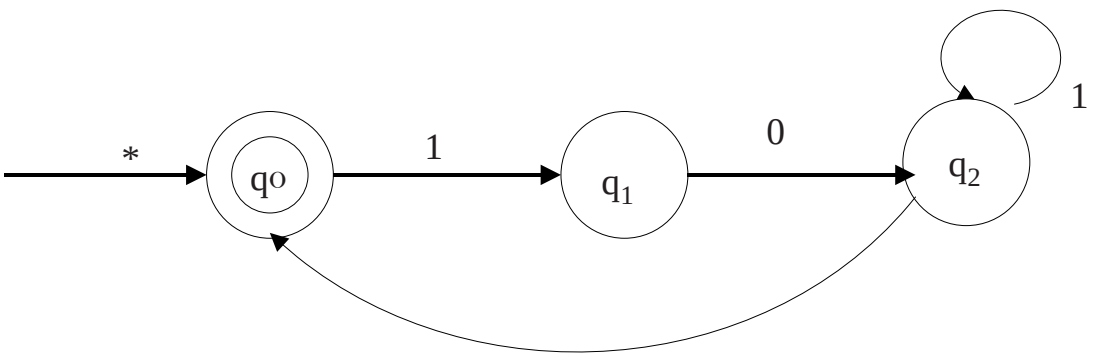
A continuación planteamos algunos ejercicios para que resuelvas; ante cualquier duda, podés consultar a tu tutor.

1. Determiná cuáles son las palabras aceptadas por los siguientes autómatas:

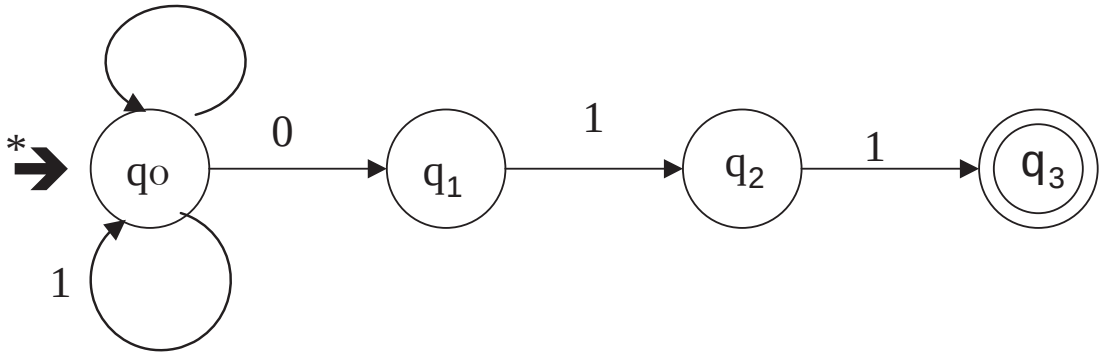
I.



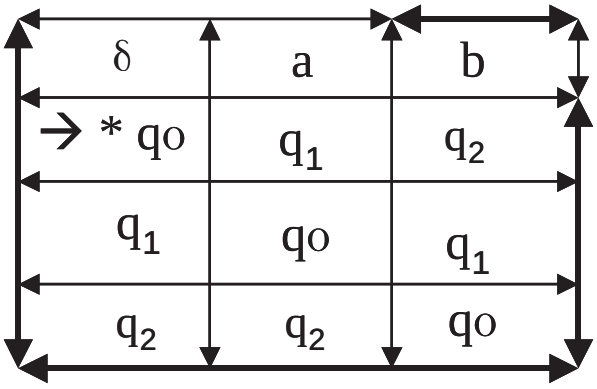
II.



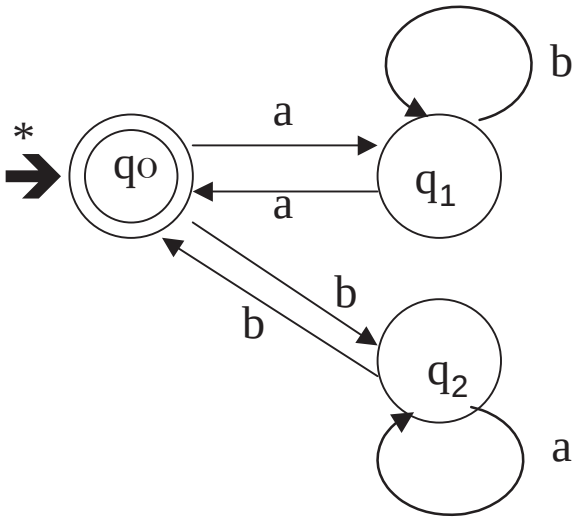
2. Indicá si el siguiente autómata finito es o no determinístico. Justificá tu respuesta.



3. Dibujá el diagrama de transición.



4. Dá la Tabla de trancisión



Ahora definamos formalmente los Autómatas finitos no determinísticos:

Un autómata finito no determinístico es un autómata finito que puede realizar transiciones por la palabra λ

$A = (Q; V; \delta; q; F)$ con $\delta = Q \times (V \cup \{ \lambda \}) \rightarrow P(Q)$

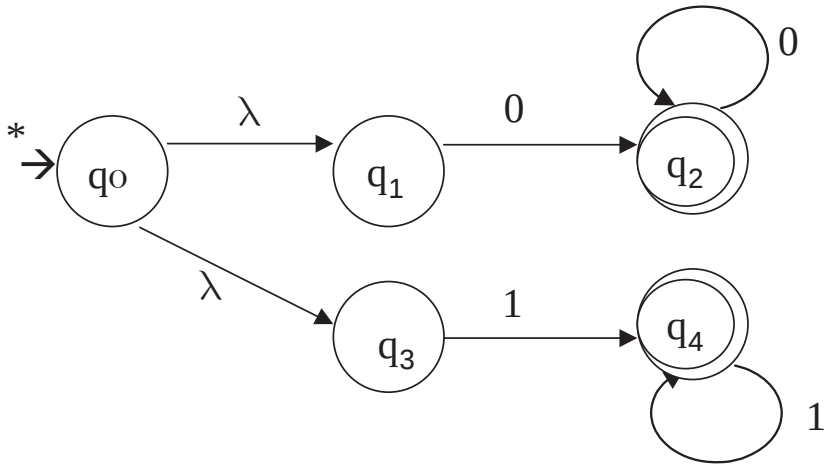
- Una transición por la palabra λ, es un cambio de estado sin la intervención de ningún carácter de la palabra en estudio.
- La transición que se ejecuta en una etapa dada puede ser incierta pues nada lo determina, por eso se dicen no determinístico.
- Cada estado puede tener, en el diagrama de transición, más de una arista etiquetada con cada letra del alfabeto V.
- Para cualquier estado q se pueden tener cero ó más alternativas a elegir como estado siguiente, todas para el mismo símbolo de entrada.
- Si $A = (Q; V; \delta; q; F)$ es un autómata finito no determinístico, el lenguaje que acepta A se indica $L(A)$ y es regular.

Ejemplos:

Sea $A = (\{q_o, q_1, q_2, q_3, q_4\}; \{0, 1\}; \delta; qo; \{q_2, q_4\})$, con δ dada por

$\delta(q_o; l) = \{q_1, q_3\}$
$\delta(q_1; 0) = \{q_2\}$
$\delta(q_3; 1) = \{q_4\}$
$\delta(q_2; 0) = \{q_2\}$
$\delta(q_4; 1) = \{q_4\}$

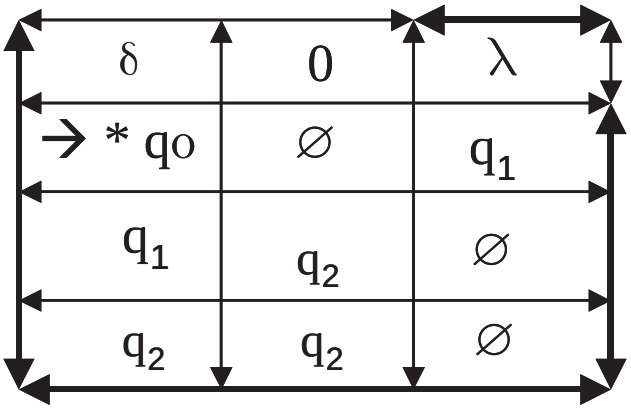
El diagrama de transición correspondiente es:



Tratá de ejercitar autómatas finitos no determinísticos; para ello te planteamos a continuación una serie de ejercicios:

1) Graficá los diagramas de transición de los autómatas finitos no determinísticos

a). $A = (\{q_o, q_1, q_2\}; \{0\}; \delta; \{q_2\})$, con δ dada por la siguiente tabla de transición:



δ	0	λ
→ * qo	∅	q_1
q_1	q_2	∅
q_2	q_2	∅

b). $A = (\{q_o, q_1, q_2, q_3\}; \{0, 1\}; q_o; \{q_3\})$, con δ dada por:

$\delta(q_o; 0) = \{q_o\}$ y $\delta(q_o; 0) = \{q_1\} \rightarrow \delta(q_o; 0) = \{q_o, q_1\}$ $\delta(q_o; 1) = \{q_o\}$

$\delta(q_1; 1) = \{q_2\}$

$\delta(q_2; 1) = \{q_3\}$

c). $A = (\{q_o, q_1, q_2, q_3\}; \{0, 1\}; q_o; \{q_3\})$, con δ dada por:

$\delta(q_o; 0) = \{q_o\}$ y $\delta(q_o; 0) = \{q_1\} \rightarrow \delta(q_o; 0) = \{q_o, q_1\}$ $\delta(q_o; 1) = \{q_o\}$

$\delta(q_1; 1) = \{q_2\}$

$\delta(q_2; 1) = \{q_3\}$

d). $A = (\{q_o, q_1, q_2, q_3, q_4\}; \{0, 1\}; \delta; q_o; \{q_2, q_4\})$, con δ dada por:

$\delta(q_o; \lambda) = \{q_1, q_3\} \delta(q_1; 0) = \{q_2\}$

$\delta(q_3; 1) = \{q_4\}$

$\delta(q_2; 0) = \{q_2\}$

$\delta(q_4; 1) = \{q_4\}$

e). $A = (\{a, b, c\}; \{0, 1\}; \delta; a; \{a, b\})$, con δ dada por:

$\delta(a; 0) = \{b\}$

$\delta(a; 1) = \{a\}$

$\delta(b; 0) = \{a\}$

$\delta(b; 1) = \{c\}$

δ (c; 0) = {a }
δ (c; 1) = {b }

Sintetizando, la diferencia entre autómatas finitos determinísticos y no determinísticos:

Una de las diferencias principales más importantes entre un autómata finito determínistico (A.F.D.) y un autómata finito no determinístico (A.F.N.) es que:

$\forall q \in Q, \forall a \in V => | \delta(q;a) | \leq 1$ *(A.F.D)*

Para los A.F.N.:

$\exists q \in Q, \exists a \in V \cup \{ \lambda \} y | \delta(q;a) | > 1$

Finalizamos aquí el recorrido por los contenidos de la asignatura.
Hemos abordado el programa vigente de Matemática Discreta de UTNFRBA abarcando temas que a veces resultan muy complejos.
Te recomendamos que aproveches las tutorías para consultar dudas pendientes y realizar los ejercicios de aplicación que se propongan.
Esperamos que la cursada haya resultado provechosa.