TECNOLÓGICO NACIONAL DE MEXICO

INSTITUTO TECNOLÓGICO DE IZTAPALAPA

INTEGRANTE

PEREZ ARMAS FAUSTO ISAAC

181080037

ISC-6AM

LENGUAJES Y AUTOMATAS I

M.C. ABIEL TOMÁS PARRA HERNÁNDEZ

SEP 2020 / FEB 2021

ACTIVIDAD SEMANA 5

REPRESENTACIÓN FINITA.

Un AFD tiene un conjunto finito de estados y un conjunto finito de símbolos de entrada. El término “determinista” hace referencia al hecho de que para cada entrada sólo existe uno y sólo un estado al que el autómata puede hacer la transición a partir de su estado actual. Un estado se diseña para que sea el estado inicial, y cero o más estados para que sean estados de aceptación. Una función de transición determina cómo cambia el estado cada vez que se procesa un símbolo de entrada.

Un autómata finito determinista consta de:

1. Un conjunto finito de estados, a menudo designado como Q.
2. Un conjunto finito de simbolos de entrada, a menudo designado como Σ
3. Una función de transición que toma como argumentos un estado y un símbolo de entrada y devuelve un estado. la funcion de transicion se designa habitualmente como δ. En nuestra representación gráfica informal del autómata, δ se ha representa mediante arcos entre los estados y las etiquetas sobre los arcos. Si q es un estado y a es un símbolo de entrada, entonces δ(q,a) es el estado p tal que existe un arco etiquetado a que va desde q hasta p.
4. Un estado inicial, uno de los estados de Q.
5. Un conjunto de estados finales o de aceptación F. El conjunto F es un subconjunto de Q.

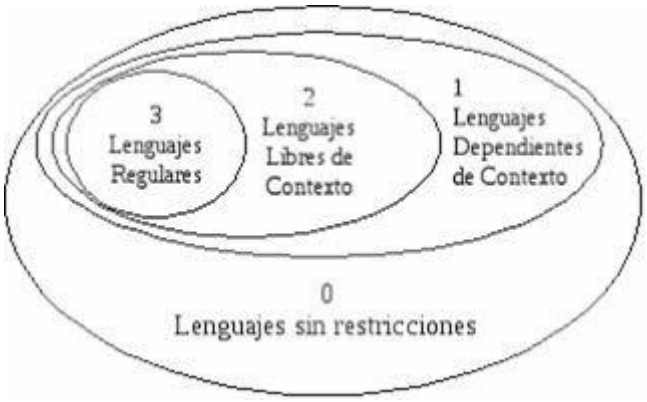
La representación más sucinta de un AFD consiste en un listado de los cinco componentes anteriores. Normalmente, en las demostraciones, definiremos un AFD

utilizando la notación de “quíntupla” siguiente: $A = (Q, \Sigma, \delta, q_0, F)$

Donde:

A *Es el nombre del AFD*
Q *es un conjunto finito de estados*
Σ *son lo símbolos de entrada*
q_0 *es el estado inicial*
$F \subseteq Q$ *es el conjunto de estados finales*
$\delta : Q \times V \rightarrow Q$ *es la funcion de transición*

GRAMÁTICA



TIPO "0" O "No restringida o recursivamente enumerables":

Incluye a todas las gramáticas formales. Estas gramáticas generan todos los lenguajes de ser reconocidos por una maquina de Turing. Y con este lenguaje se hacen los parser de un compilador

CARACTERÍSTICAS: De este tipo es que del lado derecho de cada producción puede empezar con un símbolo terminal o con un no terminal y del lado izquierdo puedes empezar con más de un símbolo no terminal.

RESTRICCIONES: Es que no tiene solamente que el del lado izquierdo debe haber por lo menos un símbolo no terminal

$$x \rightarrow y$$
$$x \in (NT/T)^{+}$$
$$y \in (NT/T)^{*}$$

NOTA: "+" significa "*sin incluir la cadena vacía"* y significa "*incluyendo la cadena vacía". "l"* significa *"o".* Estos lenguajes también son denominados "recursivamente enumerables"

TIPO “1” O “Sensible al contexto”:

Estos tipos de lenguajes se resuelven mediante autómatas lineales limitados. Con este tipo se hacen los parser (analizador sintáctico) de un compilador que transforma su entrada en un árbol de derivación. El analizador sintáctico convierte el texto de entrada en otras estructuras (comúnmente árboles), que son más útiles para el posterior análisis y capturan la jerarquía implícita de la entrada

$$\alpha \rightarrow \beta ; |\alpha| \leq |\beta|$$
$$\alpha = z_1 x\, z_2$$
$$\beta = z_1 y\, z_2$$
$$z_1, z_2 \in T^*$$
$$x \in NT$$
$$y \in (NT/T)^+$$

CARACTERÍSTICAS: Del lado derecho de cada producción puede empezar con un símbolo terminal o con un no terminal y del lado izquierdo puede empezar con más de un símbolo no terminal.

RESTRICCIONES: el número de no terminales del lado izquierdo de la producción debe ser menor o igual al número de símbolos del lado derecho

NOTA: Los lenguajes regulares y los libres de contexto también se puede resolver mediante autómatas lineales limitados

TIPO “2” O “Libres o Independientes de contexto”:
Estos tipos de lenguajes se resuelven mediante autómatas descendentes y con este tipo de lenguaje se programa los parser en un compilador, permiten describir la mayoría de los lenguajes de programación, de hecho, la sintaxis de la mayoría de los lenguajes de programación está definida mediante gramáticas libres de contexto.

$$x \rightarrow y$$
$$x \in NT$$
$$y \in (NT/T)^*$$

CARACTERISTICAS: Del lado derecho de cada producción puede empezar con símbolo terminal o con un no terminal. Los lenguajes regulares también se pueden resolver mediante autómatas descendentes

IPO "3" O "Lenguajes regulares":

Estos tipos de lenguajes se resuelven mediante autómatas finitos y con este tipo de lenguaje se hacen los scanners. Estas gramaticas se restrigen a aquellas reglas que tienen en la parte izquierda un no terminal, y en la parte derecha un solo terminal, posiblemente seguido de un no terminal y también esta familia de lenguajes pueden ser obtenidas por medio de expresiones regulares

$$\alpha \rightarrow \beta$$
$$\alpha \in NT$$
$$\beta \in \begin{cases} aB \\ Ba \\ b \end{cases}$$
$$B \in NT$$
$$a \in T^{+}$$
$$b \in T^{*}$$

CARACTERÍSTICAS: Del lado derecho de cada producción debe empezar con un símbolo terminal. Aquí le dejo una tabla donde se representan los tipos de lenguaje según sus características.

Arboles de Derivación

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

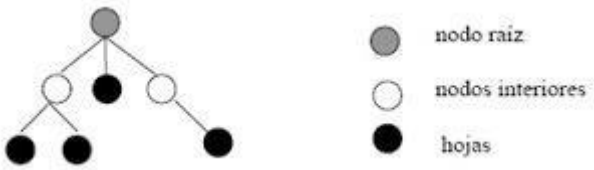
Un árbol es un conjunto de puntos, llamados nodos, unidos por líneas, llamadas arcos. Un arco conecta dos nodos distintos. Para ser un árbol un conjunto de nodos y arcos debe satisfacer ciertas propiedades:

- Hay un único nodo distinguido, llamado raíz (se dibuja en la parte superior) que no tiene arcos incidentes.

- Todo nodo c excepto el nodo raíz está conectado con un arco a otro nodo k, llamado el padre de c (c es el hijo de k). El padre de un nodo, se dibuja por encima del nodo.
- Todos los nodos están conectados al nodo raíz mediante un único camino.
- Los nodos que no tienen hijos se denominan hojas, el resto de los nodos se denominan nodos interiores.

El árbol de derivación tiene las siguientes propiedades:

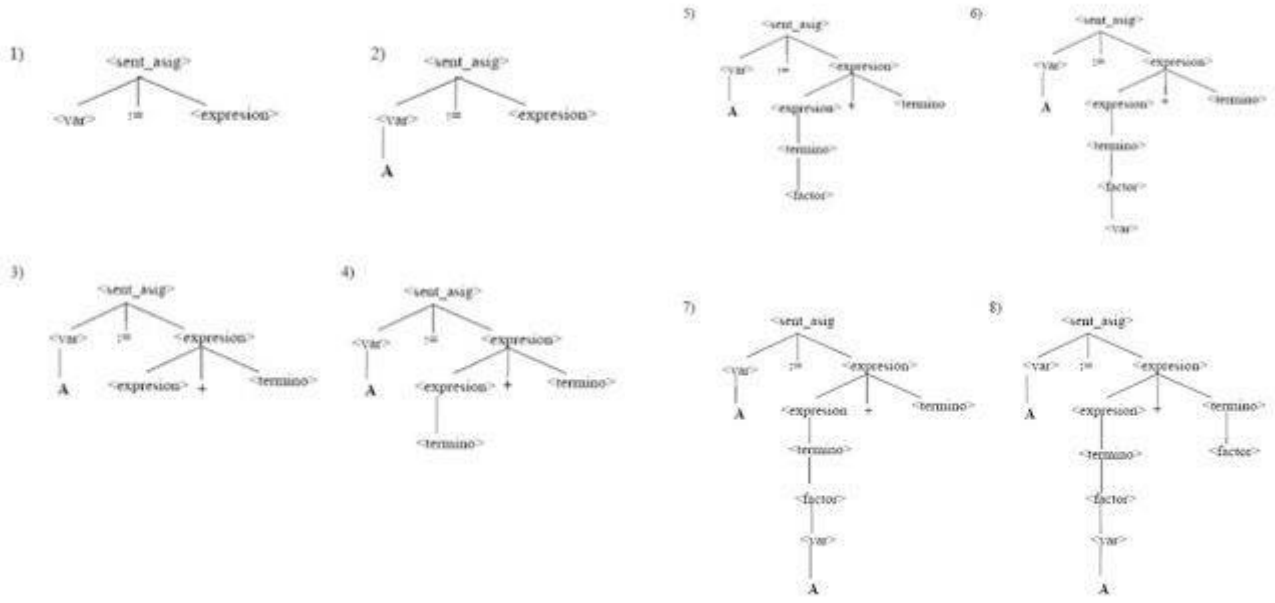
- El nodo raíz está rotulado con el símbolo distinguido de la gramática
- Cada hoja corresponde a un símbolo terminal o un símbolo no terminal; ● Cada nodo interior corresponde a un símbolo no terminal.



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.

La siguiente definición BNF describe la sintaxis (simplificada) de una sentencia de asignación de un lenguaje tipo Pascal:

Por ejemplo, la sentencia A:= A + B es una sentencia de asignación que pertenece al lenguaje definido por la definición BNF dada, y cuyo árbol de derivación se construye como se muestra a continuación:



El árbol de derivación correspondiente a la sentencia C:= D – E * F es el siguiente: