“TECNOLÓGICO NACIONAL DE MEXICO”

INSTITUTO TECNOLÓGICO DE IZTAPALAPA

INTEGRANTES:

ISC-6AM

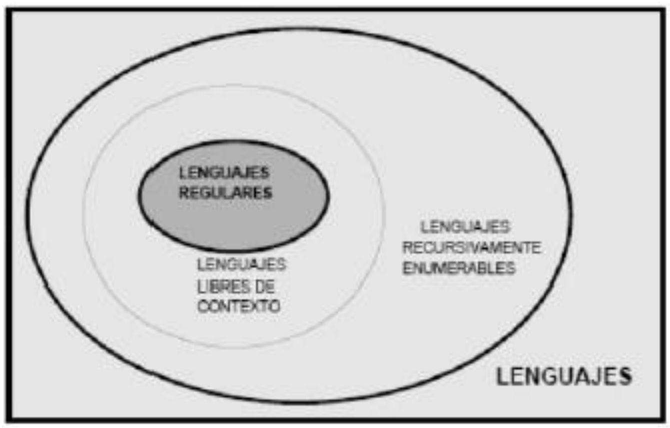
LENGUAJES Y AUTOMATAS I

M.C. ABIEL TOMÁS PARRA HERNÁNDEZ

SEP 2020 / FEB 2021

ACTIVIDAD SEMANA 6

Cuananemi Cuanalo Mario Alberto
Gramática Regular
Los LR en la jerarquía de Chomsky La clasificación de lenguajes en clases de lenguajes se
debe a N. Chomsky, quien propuso una jerarquía de lenguajes, donde las clases más
complejas incluyen a las más simples.



Los "lenguajes regulares" es la clase más pequeña, e incluye a los lenguajes más simples.
Por ejemplo, el conjunto de todos los números binarios. Los "lenguajes libres de contexto"
incluye a los LR. Por ejemplo, la mayoría de los lenguajes de programación. Los "lenguajes
recursivamente enumerables" que incluyen a los dos anteriores.
GRAMATICAS REGULARES - EXPRESIONES REGULARES Gramáticas Las gramáticas
formales definen un lenguaje describiendo cómo se pueden generar las cadenas del lenguaje.
Una gramática formal es una cuadrupla G = (N, T, P, S) donde
- N es un conjunto finito de símbolos no terminales
- T es un conjunto finito de símbolos terminales $N \cap T = \emptyset$
- P es un conjunto finito de producciones

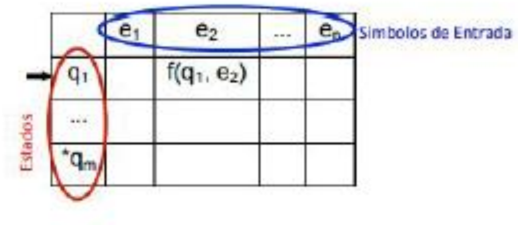
Cada producción de P tiene la forma
$\alpha \rightarrow \beta$, $\alpha = \varphi A \rho$ y $\beta = \varphi \omega \rho$ $\varphi, \omega, \rho \in (N \cup T)^*$ y A es S ó $A \in N$
- S es el símbolo distinguido o axioma $S \notin (N \cup T)$

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

lenguajes.
Gramáticas regulares (Tipo 3)
Generan los lenguajes regulares (aquellos reconocidos por un autómata finito). Son las
gramáticas más restrictivas.
El lado derecho de una producción debe contener un símbolo terminal y, como máximo, un
símbolo no terminal. Estas gramáticas pueden ser: - Lineales a derecha, si todas las
producciones son de la forma A ∈ N ∪ {S} A → aB ó A → a B ∈ N a ∈ T (en el lado
derecho de las producciones el símbolo no terminal aparece a la derecha del símbolo
terminal) - Lineales a izquierda, si todas las producciones son de la forma A ∈ N ∪ {S} A →
Ba ó A → a B ∈ N a ∈ T (en el lado derecho de las producciones el símbolo no terminal
aparece a la izquierda del símbolo terminal) En ambos casos, se puede incluir la producción
S → ε, si el lenguaje que se quiere generar contiene la cadena vacía. Por ejemplo las
siguientes gramáticas G1 y G2, son gramáticas regulares lineales a derecha y lineales a
izquierda respectivamente, que generan el lenguaje L = {a2n / n ≥ 0} G1 = ({A, B}, {a}, P1,
S1) G2 = ({C, D}, {a}, P2, S2) donde P1 es el cjto. donde P2 es el cjto. S1 → ε S2 → ε S1 →
aA S2 → Ca A → aB C → Da A → a C → a B → aA D → Ca
Autómatas Finitos
Autómatas

Autómatas Finitos

Autómatas Finitos.
Representación
· Se pueden representar mediante:
1. Diagramas de transición o
2. Tablas de transición
1. Diagramasde transición:
Nodos eDquetados por los estados ($q_i \in$ Conjunto de estados)
· Arcos entre nodos q_i a q_j eDquetados con e_i (e_i es un símbolo de entrada) si existe la transición de q_i , a q_j con e_i
· El estado inicial se señala con →
· El estado final se señala con * o con doble círculo
2. Tablas de transición:
· Filas encabezadas por los estados ($q_i \in$ Conjunto de estados)
· Columnas encabezadas por los símbolos de entrada ($e_i \in$ alfabeto de entrada)



Autómatas Finitos Deterministas (AFD)

Este tipo de autómatas admite su definición de dos maneras bien diferentes:: Como autómatas traductores o reconocedores. La definición como autómatas traductores continua a la definición de las máquinas secuenciales, y se los podría definir como una subclase de estas, ya que los autómatas finitos tendrían como limitante no poder iniciar desde cualquier estado como lo hacen en las máquinas secuenciales.

La forma que adoptaremos para la definición de los autómatas finitos deterministas es como autómatas reconocedores, ya que se ajusta con los contenidos de la informática teórica y utilización que se les da dentro del diseño de los analizadores léxicos.

Estos autómatas solo se limitarán a aceptar o no una determinada cadena recibida en la entrada, por lo tanto podemos decir que la salida de los mismos solo tendrá dos valores posibles aceptar o no aceptar a la palabra de entrada.

Al igual que en las máquinas secuenciales, estos autómatas transitarán entre un conjunto finito de estados posibles, a medida que reciban sucesivamente los caracteres de entrada, en un instante determinado de tiempo el autómata solo podrá estar en uno y solo uno de los estados posibles.

Una característica importante de este tipo de autómatas es el determinismo, lo cuál significa que estando en un estado y recibiendo una entrada del exterior el autómata tendrá la posibilidad de transitar a uno y solo un estado del conjunto de estados posibles.

Los autómatas finitos deterministas quedarán formalmente definida mediante una quintupla como sigue:

$$AFD = (\sum , Q, q_0, F, f)$$

donde:

$\sum$	Alfabeto de símbolos de entrada.
Q	Conjunto finito de estados
q_0	$q_0 \in Q$ – estado inicial previsto
F	$F \subseteq Q$ - es el conjunto de estado finales de aceptación.
f	Función de transición de estados definida como $f: Q \times \sum \longrightarrow Q$

Autómatas finitos no deterministas

Autómata finito no determinista. Es el autómata finito que tiene transiciones vacías o que por cada símbolo desde un estado de origen se llega a más de un estado destino.

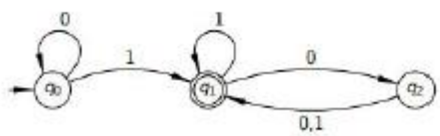
Los AFND son definiciones no tan deseables dentro de los lenguajes regulares porque dificultan su implementación tanto mecánica como informática; aunque en la mayoría de las transformaciones a lo interno de los LR (expresiones regulares a AF, gramáticas regulares a AF) conducen a AFND. Los AFND, por tanto, son imprescindibles en el análisis lexicográfico y el diseño de los lenguajes de programación.

Sea un autómata finito definido por la 5-tupla $A=<Q, T, g, F, q0>$, donde *Q* es el conjunto de estados, *T* el alfabeto de símbolos terminales, la relación de transiciones $g \subseteq Q \times (T \cup \{\xi\}) \times Q$ ó $g = \{(q_i, x, q_j) | q_i, q_j \in Q ; x \in T\}$ (léase: del estado *qi* mediante el terminal *x* se va a *qj*), *F* son los estados finales o de llegada dentro de *Q*, *q0* es el estado inicial o de partida; se dice que *A* es un **autómata finito no determinista** (AFND) si y sólo si existen en *g* al menos una de las siguientes transiciones:

Consecuencias

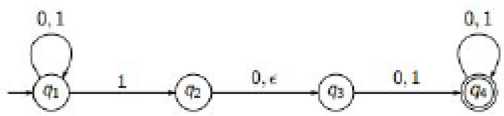
La definición formal de AFND se basa en la consideración de que a menudo según los algoritmos de transformación de expresiones y gramáticas regulares a AF terminan obteniéndose autómatas con transiciones múltiples para un mismo símbolo o transiciones vacías. Independientemente que sean indeseables, sobre todo para la implementación material, fundamentalmente mecánica, de los autómatas finitos, son imprescindibles durante la modelación de analizadores lexicográficos de los elementos gramaticales de los lenguajes de programación, llamados tókenes, como literales numéricos, identificadores, cadenas de texto, operadores, etc.

Automata finito Determinista:



- La computación del autómata con entrada 011 es (q_0, q_0, q_1, q_1) que me dice la secuencia de estados por los que pasa con entrada 011
- Cada entrada me da exactamente una computación. Tengo siempre como mucho una opción desde un estado si leo un símbolo ← Esto se llama determinismo

Automata finito no determinista



- Desde q_1 con el símbolo 1 hay dos opciones posibles
- Desde q_2 hay una posibilidad de moverse sin leer ningún símbolo (la marcada como ϵ)

A continuación se presentan algunos conceptos básicos necesarios para la comprensión de los ejercicios que se presentan en las secciones subsecuentes. Símbolo es un signo que representa algo abstracto. En este material, símbolo se referirá a un caracter alfanumérico. Ejemplos a, b, 1, 0, x, y, z, 9,

Alfabeto es un conjunto de símbolos y normalmente se denota con la letra Σ. Ejemplos Σ = {a,b,c,…z} Σ = {1,2,3,…9} Σ = {0,1} Σ = {a,b} Cadena o palabra es un conjunto de símbolos de algún alfabeto Σ concatenados entre sí, es decir uno enseguida del otro. Ejemplos Para el alfabeto Σ = {a,b,c,…z} algunas cadenas son: ab, z, cc, abc, abab Para el alfabeto Σ = {0,1} algunas cadenas son: 0, 1, 01, 000, 010

Cadena Vacía ε, es la cadena que no contiene ningún símbolo. Lenguaje es un conjunto de cadenas o palabras definido en un alfabeto Σ. Ejemplos Si Σ = {0,1} podríamos definir los lenguajes "conjunto de cadenas en Σ que terminan en 0" algunos de las palabras del lenguajes serían: 0, 10,00,010,100, 110…