AUTOMATAS DE ESTADOS FINITOS Y MAQUINAS DE TURING

Contenido Conceptual

Autómatas finitos deterministas y no deterministas, tablas de transición, aceptación de cadenas. Obtención de autómatas a partir de expresiones regulares. Conversión de AFN en AFD, Diseño de analizadores léxicos, estructura del analizador.
Máquinas de Turing. Definición formal. Diagrama de estados. Lenguajes asociados. Restricciones. Máquinas de Turing no deterministas.

Objetivos

- Identificar y diferenciar autómatas finitos deterministas y no deterministas.
- Lograr la capacidad para representar e implementar autómatas que reconozcan patrones en cadenas de entrada para obtener los componentes léxicos a partir de expresiones regulares.
- Obtener la habilidad para construir autómatas finitos no deterministas mediante "Construcción de Thompson" y convertirlos en autómatas finitos deterministas.
- Adquirir la capacidad para representar las configuraciones de máquinas de Turning definidas, para las cadenas de entrada.
- Lograr la habilidad para crear máquinas de Turing.

PARTE "A": AUTOMATAS DE ESTADOS FINITOS

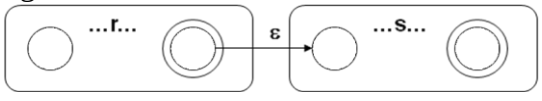
Diagrama de transiciones representa las acciones que tienen lugar cuando el analizador léxico es llamado por el analizador sintáctico para obtener el siguiente componente léxico.
Construcción del diagrama de transiciones:

- Las *posiciones* en un diagrama de transición se representan con un círculo y se llaman *estados*.
- Los *estados* se conectan mediante flechas, llamadas *aristas.*
- Las *aristas* tienen etiquetas que indican los caracteres de entrada.
- El *estado de inicio* es el estado inicial del diagrama de transición.
- El *estado de aceptación* es el estado en el cual se ha encontrado un token y se indica con un círculo doble.

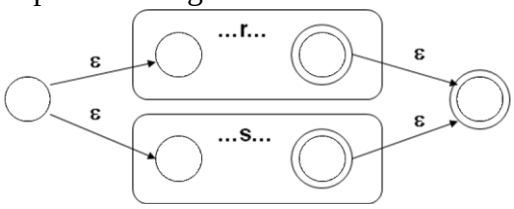
Construcción de Thompson permite la obtención de cada operación de la expresión regular al conectar entre sí los AFN de las subexpresiones.
Representaciones de la Construcción de Thompson:

- *Expresiones regulares básicas:* a representa una correspondencia con un carácter simple del alfabeto

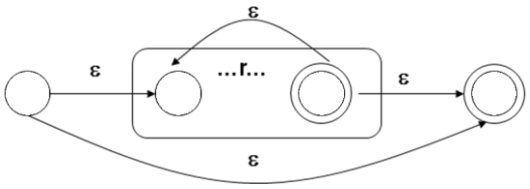
- *Concatenación:* AFN para la expresión regular rs , donde r y s son expresiones regulares.



- *Selección de alternativas*: AFN para la expresión regular *r*|*s*, donde r y s son expresiones regulares.



- *Repetición o cerradura de Kleene:* AFN para la expresión regular *r**, donde r es una expresión regular.



Construcción del subconjunto:

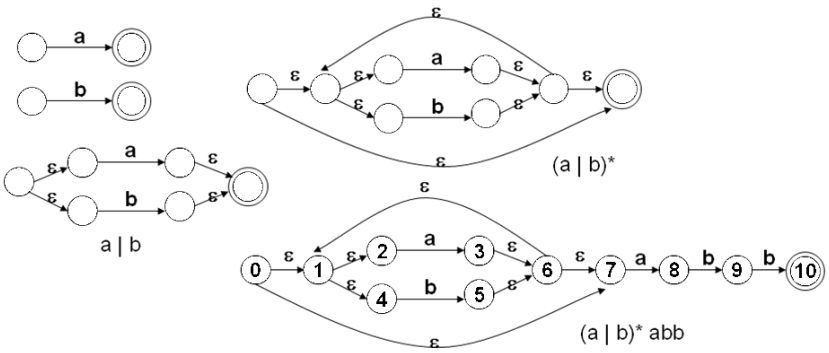
Para la construcción de un AFD a partir de un AFN M dado, M

- Calcularnos la cerradura ε del estado de inicio de M, esto se convierte en el estado de inicio de M.
- Para este conjunto, y para cada conjunto subsiguiente, calculamos las transiciones en los caracteres.

Ejercicio 1:

Cree, mediante "Construcción de Thompson", los siguientes autómatas finitos no deterministas:

Ejemplo: dada la expresión regular (a | b)* abb.

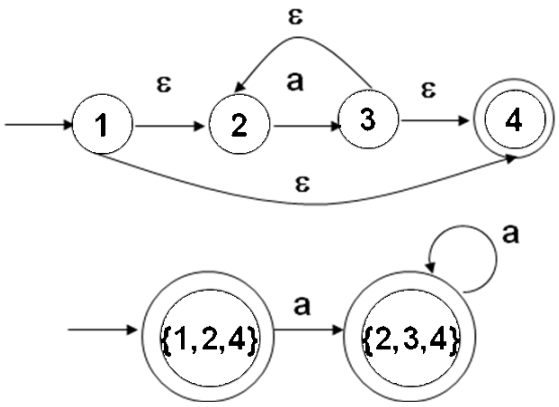


a) Que reconozca la expresión regular (b | (b* a)*)a.
b) Que reconozca la expresión regular x (x | y) *.

Ejercicio 2:

Convierta, utilizando construcción de subconjuntos, los autómatas finitos no deterministas realizados en el ejercicio anterior, en autómatas finitos deterministas.

Ejemplo: NFA que corresponde a la expresión regular a* bajo la construcción de Thompson



- El estado de inicio del DFA correspondiente es
  $\overline{1} = \{1,2,4\}$
- Existe una transición desde el estado 2 hasta el estado 3 en a, y no hay transiciones desde los estados 1 o 4 en a
  $\{1,2,4\}_a = \{3\} = \{2,3,4\}$
- Existe una transición desde 2 a 3 en a y ninguna transición a desde 3 o 4, de modo que hay una transición desde {2, 3, 4} hasta
  $\overline{\{2,3,4\}_a} = \{3\} = \{2,3,4\}$
- Existe una transición a desde {2, 3, 4} hacia sí misma.

PARTE "B": MAQUINAS DE TURING

Definiciones

Máquinas de Turing pueden reconocer lenguajes regulares, lenguajes independientes de contexto y otros tipos de lenguajes. Utilizan una cinta infinita.
Cinta infinita es una colección de celdas de almacenamiento que se extiende infinitamente en ambas direcciones, cada celda almacena un único símbolo, permite acceder a los contenidos de las celdas en cualquier orden.
Definición de una máquina de Turing es una 7-tupla M = (Q, Σ, Γ, s, b, F, δ) donde:

- Q es el conjunto finito de estados
- Σ es un alfabeto de entrada
- Γ es el alfabeto de la cinta
- s ϵ Q es el estado inicial
- B ϵ Γ es el símbolo blanco (no está en Σ)
- F es el conjunto de estados de aceptación
- δ: Q x Γ -> Q x Γ x {L,R} es una función parcial que se llama función de transición

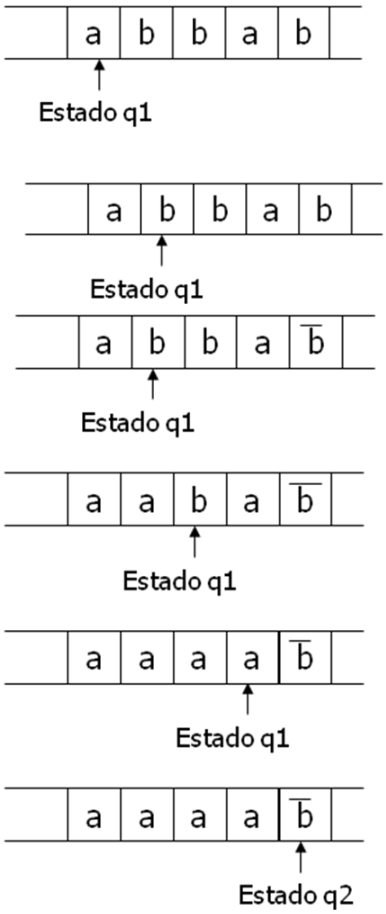
Ejercicio 1:

Especifique una cadena de entrada y represente las configuraciones de la máquina de Turing definida mediante:

Ejemplo: dada la siguiente máquina de Turing definida por:

- Q = {q1, q2}
- Σ = {a, b}
- Γ = {a, b, $\overline{b}$}
- s = q1
- F = {q2}
- δ dado por:
  - δ(q1,a)=(q1,a,R)
  - δ(q1,b)=(q1,a,R)
  - δ(q1,b)=(q2,b,L)

Para la cadena abbbb:



a) Q = {q1, q2}

Σ = {a, b}

Γ = {a, b, ~~b~~}

s = q1

F = {q2}

δ dado por: δ(q1,a)=(q1,a,R)

δ(q1,b)=(q1,a,R)

δ(q1,~~b~~)=(q2,~~b~~,L)

b) Q = {q0, q1}

Σ = {0, 1}

Γ = {0, 1, B}

s = q0

F = {q1}

δ dado por: δ(q0,0)=(q1,B,R)

δ(q0,1)=(q0,B,R)

δ(q1,0)= (q0,B,R)

δ(q1,1)= (q1,B,R)

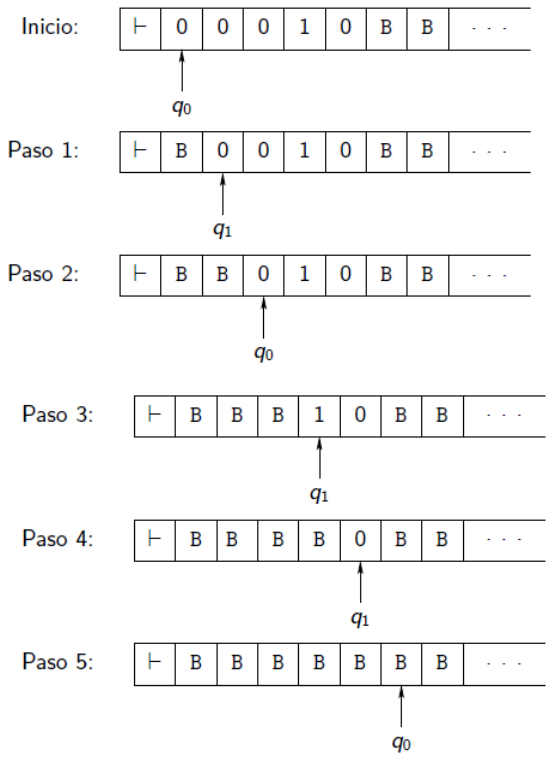
Ejercicio 2:

Construya una máquina de Turing para cada uno de los AFD obtenidos en el ejercicio 2 de la parte A. Especifique una cadena de entrada y represente las configuraciones de la máquina de Turing definidas.

Ejemplo: Construir una máquina de Turing que verifique si el número de 0s en una palabra es par: M=(Q, Σ, Γ, s, F)

- Q = {q_0, q_1}
- Σ = {0, 1}
- Γ = {0, 1, ~~b~~,B}
- s = q_0
- F = {q_0}
- δ dado por:
  - δ(q_0,0)=(q_1,B,R)
  - δ(q_0,1)=(q_0,B,R)
  - δ(q_1,0)=(q_0,B,R)
  - δ(q_1,1)=(q_1,B,R)

Supongamos que w = 00010:



La máquina acepta w = 00010.

Ejercicio 3:

Programe en Python los autómatas obtenidos en el ejercicio 2 de la parte A.