Febrero 2017

viernes, 6 de enero de 2023 20:32

EJERCICIO 1 (1 punto)

Enuncie y demuestre el Lema de Bombeo para Autómatas Finitos.

El lema de bombeo para autómatas finitos dice que

Sea L un lenguaje regular sobre un alfabeto Σ reconocido por un AFD con m estados. Si $w \in L$ y $|w| \geq m$, entonces existen s, t y u con $|t| \geq 1$ y $|st| \leq m$ tal que $w = stu$.

y para todo $n \geq 0$ $rs^n t \in L$.

El lema se demuestra utilizando el principio del palomar que dice que como existen m estados diferentes, si la cadena tiene una longitud mayor o igual que m, entonces debe existir, al menos un estado repetido.

EJERCICIO 2 (2 puntos)

Considere la siguiente gramática libre de contexto, expresada en Forma Normal de Chomsky.

E → E A	F → M H	H → E N
E → T B	F → **id**	M → **lparen**
E → M C	A → O T	N → **rparen**
E → **id**	B → P F	O → **plus**
T → T D	C → E N	P → **prod**
T → M G	D → P F	
T → **id**	G → E N	

Verifique que la cadena "**id prod lparen id plus id rparen**" pertenece al lenguaje definido por la gramática por medio del algoritmo de Cocke-Younger-Kasami.

id	prod	(	id	plus	id	)
T,F,E	—	—	—	—	—	E*,T
	P	—	—	—	—	B,D
		M	—	—	—	E,T,F
			T,F,E	—	E	C,G,H
				O	A	—
					T,F,E	C,G,H
						N

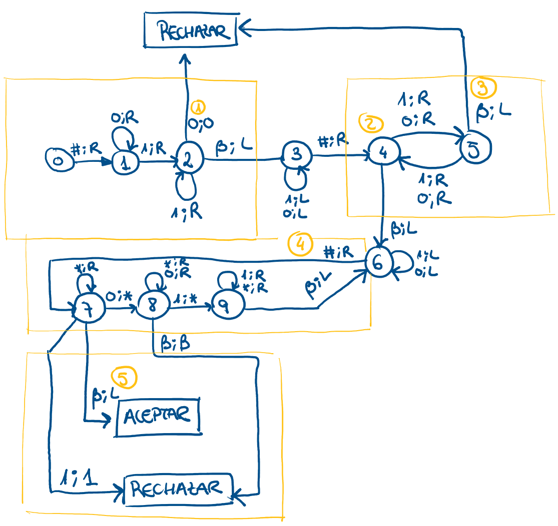
Como podemos generar la cadena a partir del símbolo inicial de la gramática, la cadena pertenece al lenguaje.

EJERCICIO 3 (2 puntos)

El siguiente algoritmo permite reconocer el lenguaje L={ $0^k 1^k$ | k ≥ 0}

1.- Recorrer la cinta. Si se encuentra un 0 a la derecha de un 1, **rechazar**.
2.- Repetir mientras queden 0s y 1s:
3.- Recorrer la lista verificando si el número de 0s y 1s es par o impar. Si es impar, ***rechazar***.
4.- Recorrer la lista marcando (quitando) la mitad de 0s y la mitad de 1s.
5.- Si no quedan 0s ni 1s, ***aceptar***. En otro caso, ***rechazar***.

Desarrolle una Máquina de Turing que implemente este algoritmo.



Los estados 0,1,2 se encargan de recorrer la cadena. En el caso de que haya un cero después de leer el primer 1, lo rechaza.

El estado 3 vuelve al comienzo de la cadena.

Los estados 4 y 5 se encargan de ver si el tamaño de la cadena es par o impar. Si es impar lo rechaza. Si es par pasa al estado 6.

El estado 6 vuelve al inicio de la cadena.

Los estados 7, 8 y 9 se encargan de tachar la mitad de 0 y la mitad de 1 en cada iteración.

Si se encuentra un 1 en lugar de un 0 significaría que hay más 1 que 0 y lo rechazaría. Si se encuentra un β en lugar de un 1 significaría que hay más cantidad de ceros que de 1 y lo rechazaría.

En el caso de que, al volver al inicio lea un β significa que ha marcado todos los caracteres y, por lo tanto, aceptaría la cadena.

EJERCICIO 4 (1.5 puntos)

Sea $HALT_{TM}$ el lenguaje formado por las cadenas <*M*,*w*> tales que *M* es la codificación de una máquina de Turing y *w* es una cadena que hace que dicha máquina termine (ya sea aceptando o rechazando). Demuestre que el lenguaje $HALT_{TM}$ es indecidible.

Supongamos $HALT_{TM}$ decidible, entonces existirá una máquina R que lo decida.

$$R(\langle M, w\rangle) = \begin{cases} \text{acepta si } M \text{ para ante } w \\ \text{rechaza si } M \text{ no para ante } w. \end{cases}$$

Podemos construir una máquina S utilizando R tal que $S(\langle M, w\rangle)$ haga:

1º Ejecute R (<M, w>)

2º Si R rechaza → rechaza

3º Si R acepta → ejecutar M(w)

4º Si M acepta → aceptar

5º Si M rechaza → rechazar

La máquina S resuelve el problema A_{TM} que es indecidible por tanto no puede ser construida, como utiliza la máquina R, esta tampoco puede ser construida y, en consecuencia, el lenguaje $HALT_{TM}$ no es decidible.

EJERCICIO 5 (2 puntos)

Considere el modelo de computación de las funciones recursivas. Asuma que las siguientes funciones ya han demostrado ser recursivas primitivas: Suma(x,y), Producto(x,y), Potencia(x,y), Decremento(x), RestaAcotada(x,y), Signo(x), SignoNegado(x), Factorial(x), Min(x,y), Max(x,y), And(x,y), Or(x,y), Not(x), Igual(x,y), Mayor(x,y), Menor(x,y), MayorOIgual(x,y), MenorOIgual(x,y).

Demuestre que la función *Mod3*(x) es primitiva recursiva.

$$Mod3(x) = Resto(x, 3) = x \% 3$$

X	Mod3(x)
0	0
1	1
2	2
3	0
4	1
5	2
6	0
7	1
8	2
9	0

CASO BASE

$Mod3(0) = Z_0$

CASO GENERAL

$Mod3(S(x)) = g(Mod3(x), x) =$

$= g(U_1^2, U_2^2)$

Vemos que los valores de la función siguen un patrón de comportamiento que puede describirse mediante la siguiente función:

$$Mod3(S(x)) = \begin{cases} Z_2 & \text{si } S(Mod3(x)) == 3 \\ S(Mod3(x)) & \text{en otro caso.} \end{cases}$$

Por lo tanto, podemos definir la función mediante funciones recursivas primitivas de la forma:

$$Mod3(x) = R(Z_0, C(If, C(Eq, C(S, U_1^2), \overbrace{S(S(S(Z_2)))}^{3}), Z_2, C(S, U_1^2)))$$

EJERCICIO 6 (1.5 puntos)

Defina los siguientes conceptos:

(a) ¿Qué es un problema de clase P?

(b) ¿Qué es un problema de clase NP?

(c) ¿Qué es un problema NP-completo?

a) Los problemas de clase P son problemas decidibles en un tiempo polinomial sobre una máquina de Turing determinista de una cinta.

Es invariante para todos los modelos de computación que pueden simularse en un tiempo polinomial.

b) Los problemas de clase NP son los problemas o lenguajes decidibles mediante una Máquina de Turing no determinista en tiempo polinomial.

c) Un problema NP-completo es un problema que pertenece a la clase NP y, si tuvieran una solución en tiempo polinomial, permitirían demostrar que todos los problemas NP tienen solución en tiempo polinomial.