Teoría de Lenguajes
Teoría de la Programación I
Soluciones

Consideraciones generales
i) Escriba nombre y C.I. en todas las hojas.
ii) Numere todas las hojas.
iii) En la primera hoja indique el total de hojas.
iv) Comience cada ejercicio en una hoja nueva.
v) Utilice las hojas de un solo lado.
vi) Entregue los ejercicios en orden.

Ejercicio 1

Sea $L_1 = \{ w \in \{a, b, c\}^*, |w|_a = 2|w|_b - |w|_c \}$

a) Clasifique a L_1 según la Jerarquía de Chomsky.

L1 es Libre de Contexto, NO Regular. Se demuestra por el contrarecíproco del Pumping Lema para lenguajes regulares. Por otro lado, se justifica que es Libre de Contexto construyendo una gramática Libre de Contexto en la parte b), o por el APD de la parte c).

Dada N cte. del PL se elige $z=a^N b^N c^N$ perteneciente a L. $|z|=3N$. La única descomposición de z en uvw, con $|uv|\leq N$ y $|v|\geq 1$, es:

$u=a^p$ $p+q<=N$
$v=a^q$ $q\geq 1$
$w=a^{N-p-q}b^N c^N$
$zi=aN+q(i-1)bNc^N$

Para i=2 $z_2=a^{N+q}b^N c^N$ la cual no pertence a L_1, porque para que sea una tira del lenguaje se debiera cumplir que la cantidad de a's es igual al doble de b's menos c's; o sea N+q= 2N-N=N; pero sólo se cumple si q=0 --> Absurdo porque $q\geq 1$

Entonces L_1 no es regular.

b) Construya una gramática G_1 / $L_1 = L(G_1)$. ¿Está simplificada? Justifique.

Construimos una gramática con las siguientes reglas de producción:

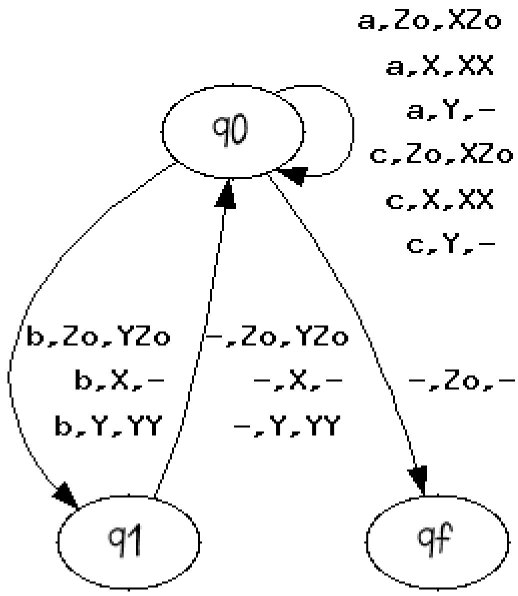
```
S → S P S P S b S
  | S P S b S P S
  | S b S P S P S
  | ε

P → a | c
```

Formalmente, esta gramática NO está simplificada al tener la regla $S \rightarrow \varepsilon$
De todas formas, como ε pertenece a L_1, es imposible quitar esa producción; con lo cual se admite como válida la respuesta que "a menos de la regla $S \rightarrow \varepsilon$, la gramática está simplificada, ya que todas las variables {S,P} son útiles. Además no hay producciones unitarias.

c) Construya un autómata M_1 / $L_1 = L(M_1)$. ¿Es determinista? Justifique.

Solución:



Nota: donde están los "-" significa "epsilon"

No es determinista porque existen transiciones comunes y transiciones épsilon que parten del mismo estado (q_0) y con el mismo símbolo en el tope de la pila (Z_o).

d) Sea $L_2 = L_1 \cap L(abca^*b^*c^*)$. Aplicando propiedades, ¿puede demostrar que L_2 es Libre de Contexto? Justifique.

L_2 es Libre de contexto, porque la intersección de un libre de contexto (L_1) y uno regular (dado por la ER) es siempre libre de contexto.

Propiedad: Si L' es un LLC y L'' es un LR, entonces $L = L' \cap L''$ es un LLC

Esbozo de la demostración:
Si L' es un LLC entonces existe M' (Q',δ',Γ',q0',Zo',F') APD / L' = L(M')
Si L'' es un LR entonces existe M'' AFD (Q'',δ'',χ'',$q_{0''}$,F'') / L'' = L(M'')
La idea es mostrar que se puede construir un APD M / L = L(M) y de ahí concluir que L es un LLC.

Los estados son pares de estados de M' y M'', para simular que nos movemos en ambos autómata a la vez, de ahí que:

- $Q \subseteq \{<q_i,q_j> / q_i \in Q' \wedge q_j \in Q''\}$
- el estado inicial es $<q_{0'},q_{0''}>$
- $\Gamma = \Gamma'$
- Zo = Zo'
- $F = \{<q_{f'},q_{f''}> / q_{f'} \in F' \wedge q_{f''} \in F''\}$
- $\delta(<q_i,q_j>,a,X) = \{(<p_i,p_j>,\beta) \ / \ (p_i,\beta) \in \delta'(q_i,a,X) \text{ y } \delta''(q_j,a) = p_j \}$

Ejercicio 2

Sea L_2 = { w / w es de la forma xy#z, donde x,y,z $\in$ {0, 1}*, |x|=|y| > 0 y z es el OR bit a bit de x con y }

a) Construya una gramática G_2 / L_2 = L(G_2).

S -> IG%#F

/* Genero ternas de ORs formadas por:
0s y 1s de la tira a
As (equivalen a 0s) y Bs (equivalen a 1s) de la tira b
Cs (equivalen a 0s) y Ds (equivalen a 1s) de la tira c
*/

T -> 0AC | 0BD | 1AD | 1BD
G -> TG | T

/* Muevo Cs hacia la derecha */

CA -> AC
CB -> BC
C0 -> 0C
C1 -> 1C
C% -> %C

/* Transformo Cs en 0s de la tira c. */

C# -> #0

/* Muevo Ds hacia la derecha */

DA -> AD
DB -> BD
D0 -> 0D
D1 -> 1D
D% -> %D

/* Transformo Ds en 1s de la tira c */

D# -> #1

/* Muevo As hacia la derecha */

A0 -> 0A
A1 -> 1A

/* Transformo As en 0s de la tira b */

A% -> %0

/* Muevo Bs hacia la derecha */

B0 -> 0B
B1 -> 1B

/* Transformo Bs en 1s de la tira b */

B% -> %1

/* Muevo I hacia la derecha */

I0 -> 0I
I1 -> 1I
I# -> #I

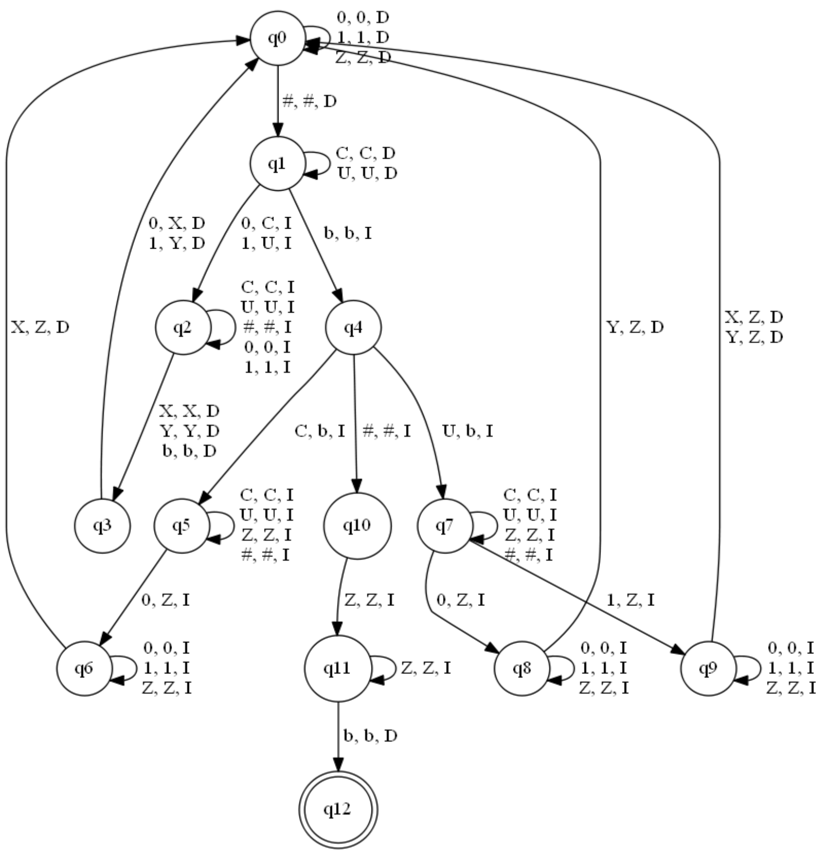
/* Elimino el % */

I% -> I

/* Elimino I y F */

IF -> eps

b) Construya un autómata M_2 / $L_2 = L(M_2)$. **[Teoría de Lenguajes]**



Breve explicación

Estados: q0, q1, q2, q3
Identificación de "x"
1. Voy hasta el último símbolo de "z" que no haya procesado.
2. Marco con C los 0 (ceros) y con U los 1 (unos) y voy al principio de la tira "x".
3. En "x", marco con X los 0 (ceros) y con Y los 1 (unos). Vuelvo a 1. hasta que se marquen todos los símbolos de "z" (esto se hace para marcar a "x", y ver dónde se "divide").
Estados: q0, q1, q4, q5, q6, q8, q9
Verificación de OR bit a bit
4. Se discriminan 3 casos: el resultado es 0, el resultado es 1, no hay más para procesar.
4a. En caso de que el resultado sea 0, se verifica que los bits de "x" y "y" sean 0.
4b. En caso de que el resultado sea 1, se verifica que por lo menos uno de los bits (de "x" o "y") es 1.
Estados: q10, q11, q12
Verificación de |x| = |y| y fin
4c. Si no hay más para procesar, verifico que no haya bits "extra" de “x” o “y”, y termino.

Ejercicio 3
Indique si las siguientes afirmaciones son verdaderas o falsas, justificando en cada caso.

a) Se define la operación DEL que, si L es un lenguaje cualquiera y **a** un símbolo del alfabeto de L, DEL(L,a) es el lenguaje resultante de eliminar **a** de todas las tiras de L.
Los lenguajes libres de contexto son cerrados bajo la operación DEL.
b) Si L_b no es regular, DEL(L_b,a) no es regular.
c) Sea L_c un lenguaje regular. Sea L_s el lenguaje resultante de eliminar el primer y último símbolo de todas las tiras del lenguaje L_c. Entonces L_s es regular.
d) Sea L_d un lenguaje para el cual existe una gramática libre de contexto G_d / L_d=L(G_d). Entonces existe una máquina de Turing M_d / L_d=L(M_d).

Solución

a) **Verdadero**. Si L es libre de contexto, entonces existe (por definición) una GLC que lo genera. Si eliminamos de todas las producciones de la gramática el símbolo **a** (sustituyendo, además, por A -> Ɛ las producciones de la forma A -> **a**), la gramática genera el lenguaje DEL(L,a), y por lo tanto es libre de contexto.

b) **Falso**. El lenguaje resultante podría ser regular. Si L={tiras de la forma $a^n b^n$, n≥0}, DEL(L,a) es L(b*), que es regular porque existe una expresión regular que lo define.

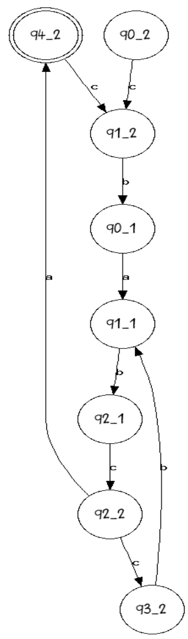
c) **Verdadero**. Sea M un AFD (Q,Σ,δ,F) tal que L_c=L(M). Construimos un nuevo autómata M'=(Q',Σ,δ',F'), tal que Q'=Q∪{q'_0,q'_F}, F'={q'_F}, y δ' surge de agregar a δ transiciones δ'(q'_0, Ɛ) = r, sii δ(q_0, a)=r, y δ'(r, Ɛ) = q'_F, sii δ'(r, a) = q_F y q_F es un estado final de M.

El autómata M' reconoce a L_s, y por lo tanto el lenguaje es regular.

d) **Verdadero**. Si hay una GLC que lo genera, es libre de contexto. Por Jerarquía de Chomsky, es también recursivamente enumerable, y por lo tanto debe haber una máquina de Turing que lo reconozca.

Ejercicio 4

Construya un Autómata Finito Determinista de dos cintas que acepte la relación:
$\{<(a(bc)^k)^p, ((cb)^k a)^p> \ / \ p, k>0 \ \}$



Ejercicio 5 [Teoría de la Programación I]

Sean los siguientes conjuntos:

A = {<i,m> / ∃n, Ix(i,n)↓ en a los sumo m pasos}
B = {<i,m> / ∀n, Ix(i,n)↓ en al menos m pasos}

Indique si las siguientes afirmaciones son verdaderas o falsas. Justifique.

i. A no es r.e.
ii. B no es r.e.
iii. B No es decidible

i. **Falso**

Es r.e. porque el siguiente programa computa la función característica parcial de A:

```
PROGRAM(X0)                       -- X0=<i,m>
    X1:=FST(X0);                  -- índice del programa
    X2:=SND(X0);                  -- cantidad de pasos a probar
    X3:=0;                        -- entrada a probar
    X4:=1;                        -- flag de fin
    WHILE (X4≠0) DO
        X5:=EVAL_PROG_STEP(X1, X3, X2);
        IF FST(X5) ≠0 THEN
          X4:=0
        ELSE
             X3:=SUC(X3)
    FI
    END;
    X4=SUC(0);
RESULT(X4)
```

ii. **Verdadero.**

Supongamos que B es r.e. ⇒ la función característica parcial de B es computable. Sea MB una macro que la computa.

Construyo el siguiente programa:

```
PROGRAM(X0)
     X1:=MB(<X0,0>);
RESULT(X1)
```

Si un programa para con toda entrada, para con toda entrada en al menos 0 pasos. El programa anterior verifica esta condición.
Pero entonces, este programa computa la función característica parcial de **TOT**, y esto es absurdo.

iii. **Verdadero.**

Por la parte (ii) B no es r.e. ⇒ B no es decidible.