Teoría de Lenguajes
Teoría de la Programación I
(Soluciones)

Ejercicio 1
Indique si las siguientes afirmaciones son verdaderas o falsas. Justifique adecuadamente cada respuesta.

a) Sean Σ, Δ alfabetos, $L_a \subseteq \Sigma^*$ y $h:\Sigma \rightarrow \Delta$ un homomorfismo.
Se cumple que si $h(L_a)$ es regular, entonces L_a es regular.

Falso: Sean $\Sigma = \{a,b\}$, $\Delta = \{c\}$ y $L_a = \{a^n b^n \,/\, n \geq 0\}$ Se toma h tal que:
$h(a) = c$
$h(b) = c$
Finalmente: $h(L_a) = \{c^{2n} \,/\, n \geq 0\}$ es regular (tiras con cantidad par de c's) siendo L_a no regular.

b) Si L_{b1} y L_{b2} son lenguajes libres de contexto pero no regulares, entonces $L_{b1} \cup L_{b2}$ también es libre de contexto no regular.

Falso:
Sean: $L_{b1} = \{a^n b^n \,/\, n \geq 0\} \cup \{b^p a^q \,/\, p,q \geq 0\}$ es libre de contexto no regular
$L_{b2} = \{b^n a^n \,/\, n \geq 0\} \cup \{a^p b^q \,/\, p,q \geq 0\}$ es libre de contexto no regular
y
$L_{b1} \cup L_{b2} = \{a^p b^q \,/\, p,q \geq 0\} \cup \{b^p a^q \,/\, p,q \geq 0\} = L(a^*b^*|b^*a^*)$ es regular

c) Si L_{c1} no es regular y $L_{c1} \cap L_{c2}$ es regular y no vacío, entonces L_{c2} es regular.

Falso:
Sean $L_{c1} = \{a^n b^n \,/\, n \geq 1\} \cup \{c\}$ es libre de contexto no regular
$L_{c2} = \{b^n a^n \,/\, n \geq 1\} \cup \{c\}$ es libre de contexto no regular
y
$L_{c1} \cap L_{c2} = \{c\}$ es regular y no vacío

Ejercicio 2

$L_{21} = \{ a^k b^r a^t b^k \,/\, r > k > 0, r \geq 2t > 0 \}$

El lenguaje es RE y NO Libre de Contexto (no se demuestra porque solo se pide gramática)

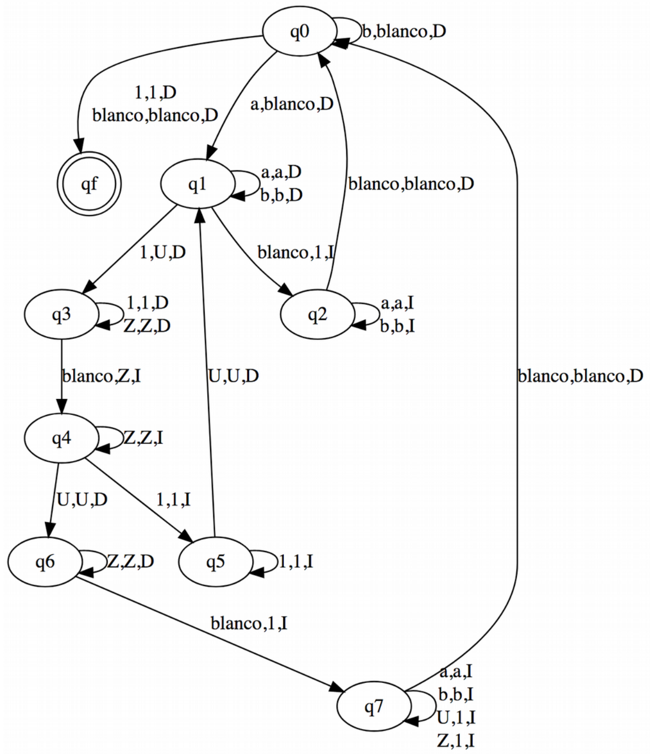
$S \rightarrow$ aa BB S A bb | a BB S A b | BB S A | a B S b | B S
$S \rightarrow$ a M bb a M b
B a $\rightarrow$ a B
BM $\rightarrow$ M b
bA $\rightarrow$ A b
MA $\rightarrow$ a M
M $\rightarrow$ ε

$L_{22} = \{ a^k b^r (ab)^t \,/\, r \geq 0; k > 0, t \text{ MOD } 2 = 0 \}$

Este lenguaje es Regular. Se construye una Gramática Lineal (en este caso una GLD).

S → aS | aA | a
A → bA | ababB | abab | b
B → ababB | abab

Ejercicio 3



Ejercicio 4

Sea $L_4 = \{ b^t a^{k+r} b^{2(k-r)} \ / \ k > r \geq 0, \ t > 0 \}$

a) Clasifique L_4 según la Jerarquía de Chomsky.

Primero vamos a redefinir el lenguaje de la siguiente manera: $k - r = p$
de donde L_4 lo podemos ver como:
$L_4 = \{ b^t a^{r+p+r} b^{2p} \ / \ p > 0, \ r \geq 0, \ t > 0 \} = \{ b^t a^{p+2r} b^{2p} \ / \ p > 0, \ r \geq 0, \ t > 0 \}$

Sea N la constante del PL, y sea $z= ba^N b^{2N}$; (con p=N, r=0, t=1), $|z| = 3N+1 \geq N$

Consideramos todas las descomposiciones posibles para $z = uvw$ que cumplan:
$|uv| \leq N$ y $|v| \geq 1$

Analizamos las siguientes familias de descomposiciones:

i) $u=\varepsilon$ $\quad j+1 \leq N$
$v=ba^j$ $\quad j \geq 0$
$w= a^{N-j} b^{2N}$
$z_i= (ba^j)^i a^{N-j}b^{2N}$ y tomando i=0 queda
$z_0 = a^{N-j}b^{2N}$ que NO pertenece a L_4 porque NO comienza en **b**

ii) $u=ba^j$ $1+j+q \leq N$
$v=a^q$ $q \geq 1$
$w= a^{N-j-q} b^{2N}$

$z_i= ba^j (a^q)^i a^{N-j-q} b^{2N}$ y tomando i=0 queda
$z_0 = ba^{N-q} b^{2N}$ que NO pertenece a L_4 porque la cantidad de **b**'s luego de las **a**'s es **mayor** que el doble de la cantidad de **a**'s, ya que $q \geq 1$

Estas son todas las familias que cumplen $|uv| \leq N$ y $|v|\geq 1$ de donde L_4 NO es un lenguaje regular.

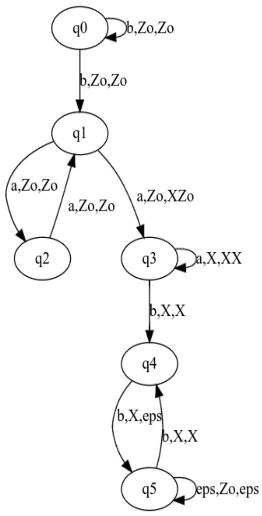
b) Construya una gramática G_4 / $L_4 = L(G_4)$. ¿Se encuentra G_4 simplificada?

G_4 / $L_4 = L(G_4)$ sería:

S → b S | b T
T → a a T | R
R → a R b b | a b b

G_4 no esta simplificada, dado que tiene una producción unitaria: T → R

c) Construya un autómata M_4 / $L_4 = L(M_4)$. ¿Es M_4 determinista?



Reconociendo por stack vacío.

M_4 no es determinista por:
- las dos transiciones que salen de q_0, para el símbolo **b** y tope del stack Zo
- las dos transiciones que salen de q_1, para el símbolo **a** y tope del stack Zo

d) Defina la relación R_L para lenguajes vista en el curso. ¿Cuántas clases de equivalencia se definen para el lenguaje L_4? Justifique.

Sea Σ un cierto alfabeto y $L \subseteq \Sigma^*$ Sean x, y dos tiras $\in \Sigma^*$
$x R_L y \Leftrightarrow \forall w \in \Sigma^* \; xw \in \Sigma^* \wedge yw \in \Sigma^*$ ó $xw \notin \Sigma^* \wedge yw \notin \Sigma^*$

La cantidad de clases de equivalencia definidas por R_{L4} es *infinita*, ya que si fuera finita, L_4 sería Regular.

Nota: Las gramáticas y los autómatas deben corresponderse con el tipo del lenguaje considerado en cada caso, según la Jerarquía de Chomsky. Se valora positivamente la simplicidad de las soluciones propuestas así como una breve explicación de éstas. Todas las respuestas deben estar debidamente justificadas.

Ejercicio 5 [Teoría de Lenguajes]

a) Sea un Autómata Finito Determinista M: $(Q, \Sigma, \delta, q_0, F)$
Dados $x,y \in \Sigma^*$ se puede afirmar que $x\ R_M\ y \Leftrightarrow \delta(q_0,x) = \delta(q_0,y)$

b)
i)
Pasaje AFND-ε → AFND

ε-clausura(q0) = {q0,q1,q3}
ε-clausura(q1) = {q1,q3}
ε-clausura(q2) = {q2}
ε-clausura(q3) = {q3}
ε-clausura(q4) = {q4,q2}

δ'	a	b
q0	{q2}	{q0,q1,q3}
q1	{}	{q0,q1,q3}
q2	{q0,q1,q3}	{q1,q2,q3,q4}
q3	{}	{q0,q1,q3}
q4	{q0,q1,q3}	{q1,q2,q3,q4}

Pasaje AFND → AFD

	δ''	a	b
p0	[q0]	[q2]	[q0,q1,q3]
p1	[q2]	[q0,q1,q3]	[q1,q2,q3,q4]
p2	[q0,q1,q3]	[q2]	[q0,q1,q3]
p3	[q1,q2,q3,4]	[q0,q1,q3]	[q0,q1,q2,q3,q4]
p4	[q0,q1,q2,q3,q4]	[q0,q1,q2,q3]	[q0,q1,q2,q3,q4]
p5	[q0,q1,q2,q3]	[q0,q1,q2,q3]	[q0,q1,q2,q3,q4]

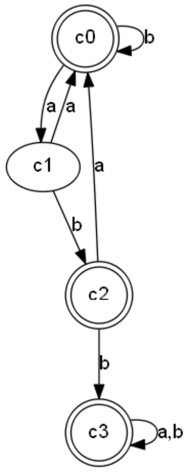
Quedando como estados finales: {p0,p2,p4,p5}

Pasaje de AFD -> AFDM

$\prod_0$: [p1] [p0p2p3p4p5]
$\prod_1$: [p1] [p0p2] [p3p4p5]
$\prod_2$: [p1] [p0p2] [p3] [p4p5]
$\prod_3 = \prod_2$

Nota: Las gramáticas y los autómatas deben corresponderse con el tipo del lenguaje considerado en cada caso, según la Jerarquía de Chomsky. Se valora positivamente la simplicidad de las soluciones propuestas así como una breve explicación de éstas. Todas las respuestas deben estar debidamente justificadas.

Luego tomamos:
C1 = [p1]
C0 = [p0p2]
C2 = [p3]
C3 = [p4p5]



ii)
Se calculan las expresiones regulares de las clases de equivalencia de R_{M5} para el autómata mínimo M_5 obtenido en la parte anterior.
Como el AFD es mínimo, por Myhill-Nerode las clases de R_{M5} coinciden con las de R_{L5} y por lo tanto las expresiones regulares que se obtienen son las solicitadas.

Para cada estado C_i se tiene una expresión regular X_i correspondiente a las tiras de Σ^* que partiendo del estado inicial C_0 terminan en dicho estado C_i.
Se utiliza el Lema de Arden: la única solución a la ecuación $X = Xr \mid s$ es: $X = sr^*$ si $\varepsilon \in L(r)$.

$X_0 = X_0b \mid X_1a \mid X_2a \mid \varepsilon$
$X_1 = X_0a$
$X_2 = X_1b$
$X_3 = X_2b \mid X_3 (a \mid b)$

Sustituyendo y aplicando Arden resulta

$X_0 = X_0(b \mid aa \mid aba) \mid \varepsilon = (b \mid aa \mid aba)^*$
$X_1 = (b \mid aa \mid aba)^*a$
$X_2 = (b \mid aa \mid aba)^*ab$
$X_3 = (b \mid aa \mid aba)^*abb \mid X_3(a \mid b) = (b \mid aa \mid aba)^*abb(a \mid b)^*$

iii)
$r_{L5} = X_0 \mid X_2 \mid X_3 = (b \mid aa \mid aba)^* (\varepsilon \mid ab \mid abb(a \mid b)^*)$, es la expresión regular que denota a la unión de las clases de equivalencia asociadas a los estados finales del AFD mínimo, y por lo tanto $L_5 = L(r_{L5})$.

Ejercicio 5 [Teoría de la Programación I]

a)

i. Computable

Este es un programa en P que la computa:

```
PROGRAM(<i,j,k>)
        X1:=1; -- flag de fin y resultado
        X2:=EVAL_PROG(i,k);
        X3:=EVAL_PROG(j,k);
RESULT(X1)
```

ii. No Computable

Definimos g'(i) = g(Id,i,i), siendo Id un índice de la identidad (converge para todo valor de su argumento).

g'(i) = g(Id,i,i) = 1 si <Id,i>↓ ^ <Ix(i),i>↑
                    indef en caso contrario

g'(i) = 1 si <Ix(i),i>↑
        indef en caso contrario

Supongamos g es computable; entonces también lo es g'.
Pero g' es la función característica parcial de K(C). Absurdo.

iii. Computable

Este es un programa en P que la computa:

```
        PROGRAM(<i,j,k>)
                X0:=0;
                X1:=EVAL_PROG_STEP(i,k,j);
                X2:=EVAL_PROG_STEP(j,k,i);
                IF (FST(X1)=1 AND FST(X2)=1) THEN
                X0:=1
        RESULT(X0) .
```

b) No es decidible
Supongamos que lo fuera; sea MD una macro que computa su función característica. Sea
el siguiente programa de índice k:

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

Como MD siempre para, Ix(k) también lo hará.

Además:
<Ix(k),i> devuelve 1 ⇔ MD(i,i,i) devuelve 1
⇔ f(i,i,i)=1 ⇔ <Ix(i),i>↓ ^ <Ix(i),i>↓
⇔ <Ix(i),i>↓ ⇔ θ(i)=1
Pero entonces, el programa de índice k computa θ. Absurdo

Nota: Las gramáticas y los autómatas deben corresponderse con el tipo del lenguaje considerado en cada caso, según la Jerarquía de Chomsky. Se valora positivamente la simplicidad de las soluciones propuestas así como una breve explicación de éstas. Todas las respuestas deben estar debidamente justificadas.