Auxiliar 4 - Gramáticas Libres de Contexto y Autómatas de Pila

Curso: Teoría de la Computación
Profesor: Alejandro Hevia
Auxiliar: Javiera Born, Nicolás Lehmann
October 4, 2013

1. Entregue una GLC para cada uno de los siguientes lenguajes:

(a) $L = \{w \in \{0,1\} \mid w \text{ no es palíndrome}\}$
Sol:

`S` $\longrightarrow$ `aSa`	| `bSb`	| `A`
`A` $\longrightarrow$ `aBb`	| `bBa`	
`B` $\longrightarrow$ `bB`	| `aB`	| ϵ

(b) $L = \{a^i b^j c^k \mid i,j,k \in \mathbb{N},\ i \leq j + k \leq 2 \cdot i\}$
Sol:
La idea es por cada a agregar una o dos b's o c's de tal forma que se preserve el orden. El lenguaje se comporta bien salvo el caso de los string de la forma $a^i b^i c^i$ con i impar. En este caso es necesario generar una b y una c en la misma regla es por esto que se agrega $A \rightarrow aAbc$

`S` $\longrightarrow$ `aSc`	| `aScc`	| `A`
`A` $\longrightarrow$ `aAbc`	| `B`	
`B` $\longrightarrow$ `aBb`	| `aBbb`	| ϵ

(c) $L = \{a^i b^j c^k \mid i,j,k \in \mathbb{N},\ i \neq j + k\}$
Sol:

`S` $\longrightarrow$ `AV`	| `VC`	| `T`
`V` $\longrightarrow$ `aVc`	| `W`	
`W` $\longrightarrow$ `aWb`	| ϵ	
`T` $\longrightarrow$ `aTc`	| `WB`	
`A` $\longrightarrow$ `aA`	| `a`	
`B` $\longrightarrow$ `bB`	| `b`	
`C` $\longrightarrow$ `cC`	| `c`	

2. Considere el alfabeto $\Sigma = \{0, \ldots, 9, (,), +, -, *, /, \hat{}\}$. Construya una GLC sobre Σ que genere expresiones aritméticas de enteros utilizando las operaciones $+, -, *, /, \hat{}$

¿Qué puede decir de su ambigüedad?

Sol:

`S` $\longrightarrow$ `F`	| `S + F`	| `S - F`
`F` $\longrightarrow$ `E`	| `F * E`	| `F / E`
`E` $\longrightarrow$ `B`	| `B ^ E`	
`B` $\longrightarrow$ `N`	| `( S )`	
`N` $\longrightarrow$ `D`	| `DN`	

`D` $\longrightarrow$ `1` | `2` | `3` $\cdots$

3. Considere la gramática

$$G : \mathtt{S} \longrightarrow \mathtt{aS} \mid \mathtt{aSbS} \mid \epsilon$$

y el lenguaje

$$E = \{w \in \{a,b\}^* \mid \text{ todo prefijo de } w \text{ tiene al menos tantas } a\text{'s como } b\text{'s}\}$$

Demuestre que $L(G) = E$

Sol:

- $L(G) \subseteq E$
  Sea w una palabra derivada en k pasos desde G. Demostraremos por inducción en el largo de la derivación que $w \in L$
  Caso base $k = 1$
  $w = \epsilon$, que pertenece claramente a E
  Hipótesis Inductiva
  Toda palabra derivada en $n \geq 1$ o menos pasos desde G pertenece a E
  Tésis Inductiva
  Suponga que w es una palabra derivada en $n+1$ pasos. Dependiendo de la primera regla usada para su derivación podemos distinguir tres casos
  - $S \to aS$
    Entonces $w = ax$ donde x es una palabra derivada en n pasos. Por HI $x \in E$ y por lo tanto x también
  - $S \to aSbS$
    Entonces $w = axby$ donde x, y son palabras derivadas en $k < n$ pasos. Por HI inductiva $x, y \in E$ concluyendo que w también.
  - $S \to \epsilon$ Cubierto en el caso base.
- $E \subseteq L(G)$
  Sea w una palabra en E de largo k. Demostraremos por inducción en k que $w \in L(G)$
  Caso base $k = 0$
  $w = \epsilon$ que puede derivarse directamente con la regla $S \to \epsilon$
  Hipótesis Inductiva
  Toda palabra en E de largo menor a n pertenece a $L(G)$.
  Tésis Inductiva
  Sea w una palabra en E de largo n. Sabemos que $w = ax$. Podemos distinguir dos casos
  - $w \in E$
    En este caso podemos usar la regla $S \to aS$ y luego usar la derivación que por HI existe para w.
  - $w \notin E$
    Entonces existe un prefijo que tiene más b's que a's. Sea u el más corto de ellos. En primer lugar u termina en b, pues en caso contrario $u = va$ y v sería un prefijo más corto con más b's que a's. Se cumple entonces que $w = avbz$, para algún $v, z \in \{a,b\}^*$. Notar que cada prefijo de v debe tener tantas a's como b's, pues si tuviera más a's, entonces $\#_a(u) \not\leq \#_b(u)$ y si tuviera más b's, u no sería el prefijo más corto con más b's que a's. Luego avb tiene el mismo número de a's que de b's. Con esto se concluye que cada prefijo de z cumple la propiedad de E, pues es necesario para que $w \in E$
    Finalmente podemos usar la regla $S \to aSbS$ y luego derivar v y z para obtener w

4. Construya un autómata de pila para cada uno de los siguientes lenguajes:

(a) $L = \{w \in \{a, b\} \mid$ el número de a's es igual que el de b's$\}$

Sol: La idea es mantener la diferencia entre a's y b's en la pila. Si la diferencia es positiva se mantendra el número con la misma cantidad de A's, en caso contrarios se guardaran B's. Definimos entonces el autómata $A = (Q, \Sigma, \Gamma, \delta, q_0, F\}$. Por simplicidad utilizaremos transicione que pueden meter más de un string a la pila (¿Cómo puedo simular estás transiciones?)

- $Q = \{q_0, q_1, q_f\}$
- $\Gamma = \{Z, A, B\}$
- $F = \{q_f\}$
- $\delta : Q \times \Sigma \times \Gamma(\mathcal{R}Q \times \Gamma^*$

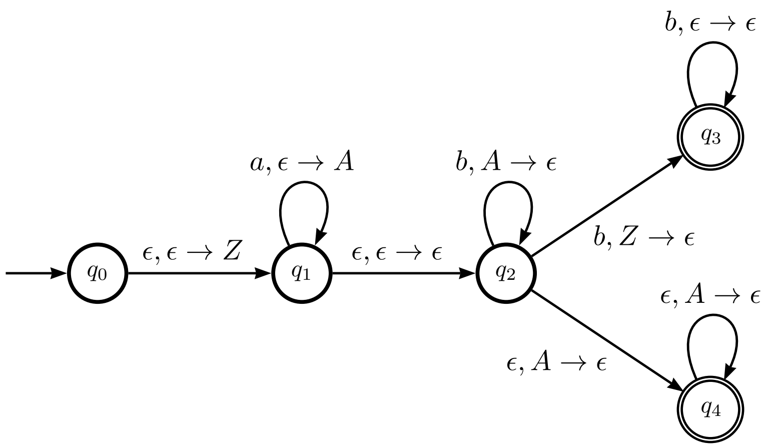
$$\delta(q_0, \epsilon, \epsilon) = (q_1, Z), \delta(q_1, a, Z) = (q_1, AZ)$$

$$\delta(q_1, a, A) = (q_1, AA), \delta(q_1, a, B) = (q_1, a, \epsilon)$$

$$\delta(q_1, b, Z) = (q_1, BZ), \delta(q_1, b, A) = (q_1, a, \epsilon)$$

$$\delta(q_1, b, B) = (q_1, BB), \delta(q_1, \epsilon, Z) = (q_f, \epsilon, \epsilon)$$

(b) $L = \{a^i b^j \mid i \neq j\}$



La idea es ingresar a la pila la cantidad de a's en el string. Luego al leer las vez tengo dos posibilidades, que se acaben primero las b's o que se acaben las a's