Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

IIC2223 — Teoría de Autómatas y Lenguajes Formales — 2' 2018

PAUTA TAREA 5

Pregunta 1

Pregunta 1.1

Dada una gramática libre de contexto $G = (V, \Sigma, P, S)$ el siguiente algoritmo verifica si el lenguaje $\mathcal{L}(G)$ es infinito

function IsInfinite(G)
  $G' \leftarrow$ remover producciones unitarias y en vacío de G
  $G'' \leftarrow$ Generadoras(G')
  $G''' \leftarrow$ Alcanzables(G'')
  for $A \in V$ **do**
    $\mathcal{S} = \varnothing$; $\mathcal{S}' = \{X \mid \exists A \to \gamma \in P.\ X \in \text{var}(\gamma)\}$
    while $\mathcal{S}' \neq \varnothing$ **do**
      let $X \in \mathcal{S}$
      $\mathcal{S} = \mathcal{S} \cup \{X\}$; $\mathcal{S}' = \mathcal{S}' - \{X\}$
      for $X \to \gamma \in P$ **do**
        for $Y \in \text{var}(\gamma)$ **do**
          if $Y \notin \mathcal{S}$ **then** $\mathcal{S}' = \mathcal{S}' \cup \{Y\}$
        end for
      end for
    end while
    if $A \in \mathcal{S}$ **then return** TRUE
  end for
  return FALSE
end function

Donde, dado $\gamma = \alpha_1\alpha_2\cdots\alpha_n$ con $\alpha_i = (V \cup \Sigma)$, $\text{var}(\gamma) = \{\alpha_i \mid \alpha_i \in V\}$. Básicamente primero se deben quitar las producciones unitarias y en vacío, luego las variables inútiles (primero las variables no generadoras y luego las no alcanzables), y finalmente se realiza una búsqueda a través de las producciones de G con tal de buscar una variable $A \in V$ tal que $A \overset{*}{\Rightarrow} \alpha_1 A \alpha_2$ (un ciclo).

Dado lo anterior, el puntaje asignado es el siguiente:

- **(4 puntos)** Dar el algoritmo (en pseudo-código) correcto y la explicación.
- **(3 puntos)** Por no quitar las variables inútiles y producciones unitarias y en vacío, o tener errores menores.
- **(0 puntos)** En otro caso

Pregunta 1.2

Se debe demostrar que el algoritmo responde TRUE ssi $\mathcal{L}$ es infinito.

($\Rightarrow$) Como se menciono anteriormente, si el algoritmo responde TRUE entonces existe $A \in V$ tal que $A \overset{*}{\Rightarrow} \alpha_1 \cdot A \cdot \alpha_2$. Como A es alcanzable, entonces $S \overset{*}{\Rightarrow} \beta_1 \cdot A \cdot \beta_2$. Ahora como toda variable es generadora, se cumple que $\alpha_1 \overset{*}{\Rightarrow} x$, $\alpha_2 \overset{*}{\Rightarrow} y$, $\beta_1 \overset{*}{\Rightarrow} u$, y $\beta_2 \overset{*}{\Rightarrow} v$. Reemplazando tenemos que $S \overset{*}{\Rightarrow} x \cdot A \cdot y$ y $S \overset{*}{\Rightarrow} u \cdot A \cdot v$. Mas aún, como G no tiene producciones unitarias ni en vacío, sabemos que $u \neq \epsilon$ ó $v \neq \epsilon$. Por último, como A es variable generadora, se tiene que $A \overset{*}{\Rightarrow} w$. Componiendo todo lo anterior se tiene que $xu^i wv^i y \in \mathcal{L}(G)$ para todo $i \geq 1$. Dado que $u \neq \epsilon$ ó $v \neq \epsilon$, concluimos que $|\mathcal{L}(G)| = \infty$.

($\Leftarrow$) Sin pérdida de generalidad, asumimos que G está en forma normal de Chomsky (la demostración en el caso general es similar). Si $|L(G)| = \infty$ entonces existe $w \in \mathcal{L}(G)$ tal que $|w| > 2^{|V|}$, por lo tanto, el árbol de derivación $\mathcal{T}$ de G para w debe tener altura mayor a $|V|$. Dicho esto, por principio del palomar, debe existir $A \in V$ tal que $A \overset{*}{\Rightarrow} \alpha_1 \cdot A \cdot \alpha_2$, por lo que el algoritmo responderá TRUE. □

Dado lo anterior, el puntaje asignado es el siguiente:

- **(4 puntos)** Ambos lados de la demostración están correctos.
- **(3 puntos)** Por errores menores (por ejemplo, no considerar las restricciones impuestas en el algortimo de quitar variables inútiles o producciones unitarias y en vacío).
- **(0 puntos)** En otro caso

Pregunta 2

Pregunta 2.1

El contrapositivo del Lema de Ogden es:

Si $\forall N \geq 0$
$\exists z \in L$ con N o más posiciones marcadas como distinguidas
y $\forall$ descomposición $z = uvwxy$ tal que
v y x tienen en conjunto por lo menos una posicion distinguida
y vwx contienen a lo más N posiciones distinguidas
$\exists i \geq 0 \; s.t. \; uv^i wx^i y \notin L$
entonces L no es un lenguaje libre de contexto.

Vamos a utilizar el contrapositivo del Lema de Ogden para demostrar que L no es libre de contexto.

Para todo $N \geq 0$, sea $z = a^{2N!} b^{N!} c^{3N!}$ donde todas las b's han sido marcadas como distinguidas. Para toda descomposición $z = uvwxy$ caemos en uno de los siguientes casos:

- x contiene más de un caracter distinto (es de la forma a^+b^+ o b^+c^+). En este caso es fácil ver que si bombeamos 2 o más veces, entonces la palabra $uv^i wx^i y$ deja de ser de la forma $a^*b^*c^*$, por lo que deja de estar en L. Pasa lo mismo si v contiene más de un caracter distinto.
- $v = a^p, x = b^q$: En este caso tenemos que $|q| \leq N$ puesto que vwx pueden contener a lo más N posiciones distinguidas. Por lo tanto, q divide a $N!$. Luego si bombeamos $\frac{2N!}{q} + 1$ veces, tenemos que

$$z' = uv^i wx^i y = a^{2N! - p + (\frac{N!}{q}+1)p} b^{N! - q + (\frac{2N!}{q}+1)q} c^{3N!} = a^s b^{3N!} c^{3N!} \notin L$$

- $v = b^q, x = c^p$: Por el mismo argumento anterior q divide a N. Si bombeamos $\frac{N!}{q} + 1$ veces tenemos que
$$z' = uv^i wx^i y = a^{2N!} b^{N!-q+(\frac{N!}{q}+1)q} c^{3N!-p+(\frac{N!}{q}+1)p} = a^{2N!} b^{2N!} c^s \notin L$$

- $v = b^q, x = b^p$: Tenemos que $k = p + q \leq N$ puesto que vwx no puede contener más de N posiciones distinguidas. Por lo tanto, k divide a $N!$. Si bombeamos $\frac{N!}{k} + 1$ veces tenemos que
$$z' = uv^i wx^i y = a^{2N!} b^{N!-k+(\frac{N!}{k}+1)k} c^{3N!} = a^{2N!} b^{2N!} c^{3N!} \notin L$$

- $v = a^q, x = c^p$: en este caso se tiene que vwx contiene $N!$ posiciones distinguidas, por lo que no es una descomposición válida.

Así, para toda descomposición válida logramos encontrar un i tal que $uv^i wx^i y$ no pertenece a L. Por lo tanto, L no es un lenguaje libre de contexto.

Dado lo anterior, el puntaje asignado es el siguiente:

- **(4 puntos)** Por realizar la demostración de manera correcta.
- **(3 puntos)** Por errores menores o realizar la demostración de manera incompleta.
- **(0 puntos)** En otro caso

Pregunta 2.2

L^C sí es un lenguaje libre de contexto. Notemos que $L^C = L_1 \cup L_2$, donde:

$$L_1 = \{a^i b^j c^k \mid i = j \vee i = k \vee j = k\}$$

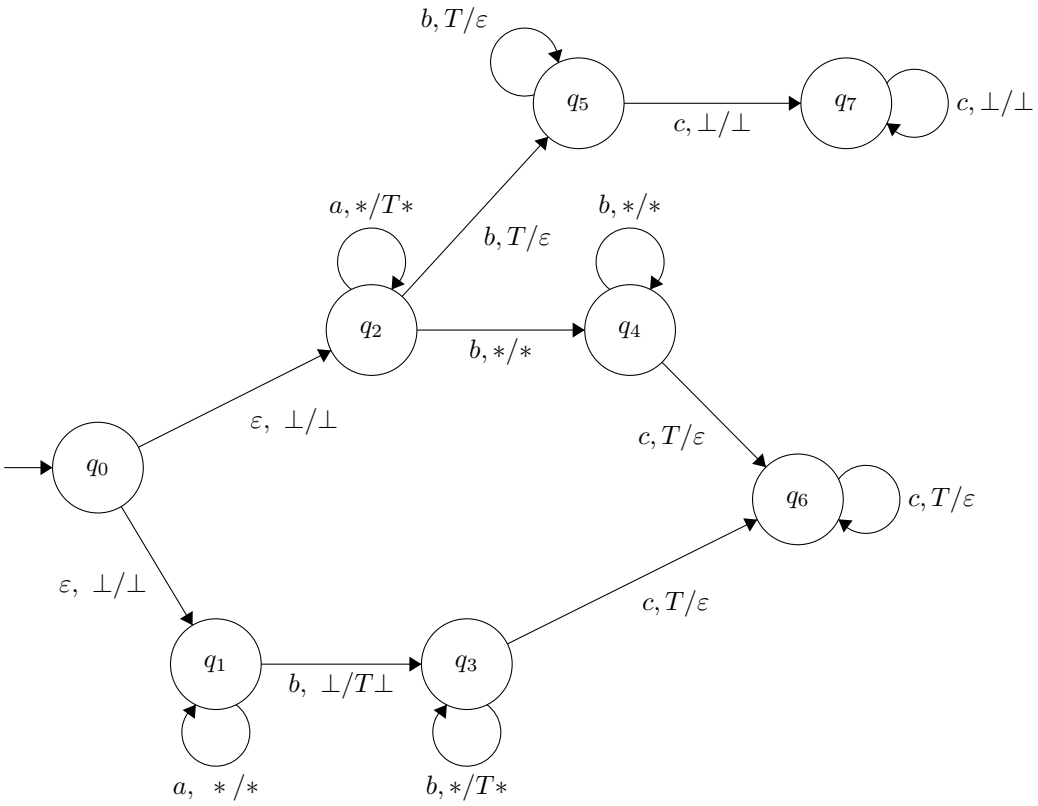
$$L_2 = \Sigma \setminus \mathcal{L}(a^* b^* c^*)$$

En otras palabras, L es la unión del lenguaje de las palabras de la forma $a^i b^j c^k$ donde al menos dos exponentes son iguales, junto con el lenguaje de las palabras que no son de la forma $a^* b^* c^*$.

Si L_1 y L_2 son lenguajes son libres de contexto, entonces L^C también lo es puesto que los lenguajes libre de contexto son cerrados bajo la unión.

Sea $L_3 = \mathcal{L}(a^* b^* c^*)$. Tenemos que L_3 es regular pues corresponde al lenguaje de una expresión regular. Los lenguajes regulares son cerrados bajo complemento, por lo que $L_2 = \Sigma \setminus \mathcal{L}(a^* b^* c^*) = L_3^C$ también es regular. Dado que los lenguajes regulares tambien son libres de contexto, tenemos que L_2 es libre de contexto.

Para demostrar que L_1 es libre de contexto podemos construir un autómata apilador que lo acepte. Un autómata es



El autómata decide de manera no determinista si va a contar la cantidad de a's o de b's. Si cuenta la cantidad de a's, luego de manera no determinista decide si lo va a comparar con la cantidad de b's o de c's. Si cuenta la cantidad de b's, entonces lo compara con la cantidad de c's. Una vez definido el autómata hay que demostrar que el lenguaje que acepta por stack vacío es igual a L_1. Hecho esto se demuestra que L_1 es libre de contexto.

Puesto que L_1 y L_2 son libres de contexto, entonces L^C también es libre de contexto.

Dado lo anterior, el puntaje asignado es el siguiente:

- **(4 puntos)** Por realizar la demostración de manera correcta.
- **(3 puntos)** Por errores menores o realizar la demostración de manera incompleta.
- **(0 puntos)** En otro caso