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

IIC2223 — Teoría de Autómatas y Lenguajes Formales — 2' 2021

PAUTA TAREA 4

Pregunta 1

Pregunta 1.1

Una posible gramática G que cumple con el enunciado es:

$$\begin{aligned} S &\to aSa \mid bSb \mid X \mid X' \\ X &\to aY \mid bY \mid Ya \mid Yb \\ X' &\to aYb \mid bYa \\ Y &\to aYa \mid bYb \mid a \mid b \mid \epsilon \end{aligned}$$

Dado lo anterior, la distribución de puntaje es la siguiente:

- **(1 punto)** Por encontrar la derivación de la variable S formando palíndromos (en este caso la primera línea)
- **(1 punto)** Por las reglas que permitan la eliminación de una letra (en este caso la segunda línea)
- **(1 punto)** Por las reglas que permitan el cambio de una letra (en este caso la tercera línea)
- **(1 punto)** Por las reglas que permitan formar un palíndromo interior luego de la eliminación o cambio de letras (en este caso la cuarta línea)

Pregunta 1.2

Primero vemos que de la gramática anterior se deduce que:

(*) $S \Rightarrow^* uSv$ si, y solo si, $|u| = |v|$ y uv es un palíndromo.

(**) $Y \Rightarrow^* w$ si, y solo si, w es un palíndromo.

Estas dos afirmaciones la usaremos en la siguiente demostración.

Por demostrar: w es un casi-palíndromo si, y solo si, w está en $\mathcal{L}(G)$.

($\Rightarrow$) Suponemos w es un casi-palíndromo. Tenemos 2 casos posibles para una palabra casi-palíndromo:

1. $w = ucv$ con $c \in \{a, b\}$ y uv es un palíndromo
2. $w = wc_1xc_2v$ con $c_1 \neq c_2$, $|u| = |v|$ y uc_1xc_1v es palíndromo

Para el caso 1, si $w = ucv$, sin pérdida de generalidad asumimos que $|u| \leq |v|$. Descomponemos $v = v_1 v_2$ tal que $|u| = |v_2|$, con uv_2 palíndromo. De esto, se desprende que v_1 es un palíndromo. Finalmente, usando (*) y (**) del inicio existe una derivación:

$$S \Rightarrow^* uSv_2 \Rightarrow uXv_2 \Rightarrow ucYv_2 \Rightarrow^* ucv_1v_2$$

Para el caso 2, si $w = uc_1xc_2v$. Sabemos que uv es palíndromo y x es palíndromo. Sin pérdida de generalidad, asumimos que $c_1 = a$ y $c_2 = b$. Finalmente, usando (*) y (**) existe una derivación:

$$S \Rightarrow^* uSv \Rightarrow uX'v \Rightarrow uaYbv \Rightarrow^* uaxbv$$

($\Leftarrow$) Suponemos que $w \in \mathcal{L}(G)$. Por construcción tenemos dos posibilidades:

1. $S \Rightarrow^* uSv \Rightarrow uXv \Rightarrow ucYv \Rightarrow^* ucxv$, con uxv un palíndromo, donde el caso en que c esta a la derecha es similar.
2. Sin pérdida de generalidad: $S \Rightarrow^* uSv \Rightarrow uX'v \Rightarrow uaYbv \Rightarrow^* uaxbv$, con $uaxav$ un palíndromo y $|u| = |v|$, donde el caso en que $X' \to bYa$ es similar.

Dado lo anterior, la distribución de puntaje es la siguiente:

- **(1 punto)** Por demostrar que cualquier palabra casi-palíndromo por perder una letra está en $\mathcal{L}(G)$
- **(1 punto)** Por demostrar que cualquier palabra casi-palíndromo por cambiar una letra está en $\mathcal{L}(G)$
- **(1 punto)** Por demostrar que cualquier palabra en $\mathcal{L}(G)$, derivada utilizando la regla $S \to X$ es un casi-palíndromo
- **(1 punto)** Por demostrar que cualquier palabra en $\mathcal{L}(G)$, derivada utilizando la regla $S \to X'$ es un casi-palíndromo

Pregunta 2

Pregunta 2.1

El algoritmo propuesto es una leve modificación del algoritmo de búsqueda `alcanzables` visto en clases:

Function `Búsqueda`(V, E, S, X)
- $A = S$
- $B = \emptyset$
- **while** $A \neq \emptyset$ **do**
  - **take** $x \in A$
  - $A = A \setminus \{x\}$
  - $B = B \cup \{x\}$
  - **foreach** $(x, y) \in E \wedge y \notin B$ **do**
    - $A = A \cup \{y\}$
- **return** $X \in B$

Luego la demostración de correctitud sería de la siguiente forma.

($\Rightarrow$) Suponga que $S \overset{*}{\Rightarrow} \alpha X \beta$ y sea:

$$S \Rightarrow \alpha_1 Y_1 \beta_1 \Rightarrow \alpha_1 \alpha_2 Y_2 \beta_2 \beta_1 \Rightarrow \alpha_1 \ldots \alpha_k Y_k \beta_k \ldots \beta_1 \Rightarrow \alpha X \beta$$

La derivación más corta para llegar de S a X. Luego, como $S \in B$ luego de la primera iteración, tenemos a $Y_1 \in B$ por la definición de E y finalmente $X \in B$ por lo que el algoritmo responde `TRUE`.

($\Leftarrow$) Suponga $X \in B$ por lo que la respuesta es `TRUE`. Luego debe existir $Y_1 \in B$ tal que $(S, Y_1) \in E$ y por lo tanto existen, para algún k, $\{Y_i \mid Y_i \in V \wedge 1 \leq i \leq k\}$ tal que $(Y_i, Y_{i+1}) \in E$ y además $(Y_k, X) \in E$. Por ende existe una derivación de la forma:

$$S \Rightarrow \alpha_1 Y_1 \beta_1 \Rightarrow \alpha_1 \alpha_2 Y_2 \beta_2 \beta_1 \Rightarrow \alpha_1 \dots \alpha_k Y_k \beta_k \dots \beta_1 \Rightarrow \alpha X \beta$$

Lo que significa que $S \overset{*}{\Rightarrow} \alpha X \beta$, es decir, X es alcanzable.

Dado lo anterior, la distribución de puntaje es la siguiente:

- **(3 puntos)** Por construir el algoritmo.
- **(1 puntos)** Por demostrar su correctitud.

Pregunta 2.2

Si suponemos que existe un algoritmo que cumple con lo pedido nos basta con encontrar un contraejemplo de input para el cual el algoritmo deba devolver dos resultados distintos. Para esto, definiremos dos gramáticas con distintas variables generadoras, pero que definen el mismo input.

La primera gramática es:

$$\begin{aligned} S &\to XZ \\ X &\to a \\ Z &\to Z \end{aligned}$$

Por otro lado definimos:

$$\begin{aligned} S &\to X \mid Z \\ X &\to a \\ Z &\to Z \end{aligned}$$

Podemos notar que ambas gramáticas definen el input $T = \{X\}$ y el grafo (V, E) con $V = \{S, X, Z\}$ y $E = \{(S, X), (S, Z), (Z, Z)\}$) y por ende, nuestro algoritmo entregará el mismo resultado. Sin embargo, podemos notar que S es generadora en la segunda gramática, pero no en la primera por lo que nuestro algoritmo debería responder `TRUE` y `FALSE` respectivamente, lo cual es una contradicción y por ende este algoritmo, que considera solo T y (V, E) no puede existir.

Dado lo anterior, la distribución de puntaje es la siguiente:

- **(3 puntos)** Por dar contraejemplos correctos.
- **(1 puntos)** Por explicar por que estos ejemplos implican que el algoritmo no existe.