Gramáticas livre de contexto

Na aula anterior, vimos um novo formalismo para computar um conjunto de linguagens mais complexo que as linguagens regulares. Vimos que a inclusão de uma memória adicional conferia aos autômatos de pilha seu maior poder computacional, uma vez que, agora, dados adicionais poderiam ser armazenados e consultados nessa estrutura.

Hoje veremos um outro formalismo para expressar essas linguagens. Esse formalismo, chamado de gramática formal, permite especificar regras que mostram como as palavras da linguagem devem ser <u>geradas</u>, tal como as gramáticas das linguagens naturais especificam como as sentenças nessas linguagens são construídas.

Exemplo: A gramática abaixo especifica um subconjunto de regras para construção de algoritmos (ou de alguns comandos em pseudo-código).

$$
\begin{array}{rcl}
\langle \text{programa} \rangle & ::= & \langle \text{declarações} \rangle \ ; \ \langle \text{lista-de-cmds} \rangle \ . \\
\vdots & & \\
\langle \text{lista-de-cmds} \rangle & ::= & \langle \text{comando} \rangle \ ; \ \langle \text{lista-de-cmds} \rangle \mid \\
 & & \lambda \\
\langle \text{comando} \rangle & ::= & \langle \text{cmd-enquanto} \rangle \mid \\
 & & \langle \text{cmd-se} \rangle \mid \\
 & & \langle \text{cmd-atribuição} \rangle \mid \\
 & & \ldots \\
\langle \text{cmd-enquanto} \rangle & ::= & \textbf{enquanto} \ \langle \text{exp-lógica} \rangle \ \textbf{faça} \\
 & & \langle \text{lista-de-cmds} \rangle \ \textbf{fimenquanto} \\
\langle \text{cmd-se} \rangle & ::= & \textbf{se} \ \langle \text{exp-lógica} \rangle \ \textbf{então} \\
 & & \langle \text{lista-de-cmds} \rangle \ \langle \text{senaoses} \rangle \ \langle \text{senao} \rangle \ \textbf{fimse} \\
\langle \text{senaoses} \rangle & ::= & \textbf{senãose} \ \langle \text{exp-lógica} \rangle \ \textbf{então} \\
 & & \langle \text{lista-de-cmds} \rangle \ \langle \text{senaoses} \rangle \mid \\
 & & \lambda \\
\langle \text{senao} \rangle & ::= & \textbf{senão} \ \langle \text{lista-de-cmds} \rangle \mid \\
 & & \lambda \\
\langle \text{cmd-atribuição} \rangle & ::= & \langle \text{variável} \rangle \leftarrow \langle \text{expressão} \rangle
\end{array}
$$

Nessa gramática, cada regra é denotada por uma variável à esquerda, o símbolo ::=, e um conjunto de símbolos e variáveis à direita. As variáveis, denotadas entre < e >, podem ser substituídas pelo lado direito das regras em que aparecem para produzir uma sentença. Partindo-se de uma variável inicial, essas podem ser substituídas sucessivamente até que a sentença tenha somente símbolos de um alfabeto (terminais). Nesse momento, teremos gerado uma palavra a partir da gramática.

Formalmente, uma gramática livre de contexto (GLC) é uma quadrupla $G = (V, \Sigma, R, P)$ em que:

- V é um conjunto variáveis (símbolos auxiliares que não pertencem à linguagem);
- Σ é um alfabeto (da linguagem);
- R é o conjunto de regras da gramática;
- $P \in V$ é a variável inicial (de partida).

As regras nas GLCs possuem o seguinte formato:

- $X \to w$ sendo $X \in V$ e $w \in (V \cup \Sigma)^*$

Ou seja, o lado esquerdo sempre é composto por uma variável, enquanto o lado direito pode ter símbolos do alfabeto, que são chamados de terminais, e variáveis, ou ainda ser a palavra vazia.

Como dito anteriormente, partindo-se da variável P, as regras podem ser aplicadas para substituir variáveis na sentença pelo correspondente da regra. Essas substituições são chamadas de **derivações**. Ao fazer uma derivação, dizemos que uma variável foi **expandida**.

Exemplo: Segue um exemplo de uma gramática para reconhecer a linguagem $L = \{0^n 1^n \mid n \in \mathbb{N}\}$

$G = (\{P\}, \{0,1\}, R, P)$, sendo o conjunto de regras R o seguinte:

$P \to 0P1 \mid \lambda$

Nesse exemplo, duas regras foram escritas na mesma linha, sendo separadas pelo símbolo '|'. Ou seja, as regras $P \to 0P1$ e $P \to \lambda$ foram colocadas lado-a-lado para simplificar a notação.

Veja que, partindo da variável P, podemos gerar a seguinte palavra:

- $P \Rightarrow 0P1 \Rightarrow 00P11 \Rightarrow 0011$

Seja $G = (V, \Sigma, R, P)$, a relação de derivação ($\Rightarrow$) é formalmente definida por:

- $uAv \Rightarrow uwv$ para $u, v, w \in (V \cup \Sigma)^*$ e $A \in V$ se a regra $(A \to w) \in R$

O fecho transitivo e reflexivo da relação de derivação é denotado por $\Rightarrow^*$. Assim, lê-se $u \Rightarrow^* v$ como u produz v em zero ou mais derivações.

Logo, a linguagem gerada por uma gramática $G = (V, \Sigma, R, P)$ é:

- $L(G) = \{w \in \Sigma^* \mid P \Rightarrow^* w\}$

Exemplo: Crie uma GLC que gere a linguagem $L = \{a^m b^n \mid m < n\}$.

$P \to EB$

$E \to aEb \mid \lambda$

$B \to bB \mid b$

- Exemplo: Crie uma GLC que gere $L = \{w \in \{0,1\}^* \mid \eta_0(w) = \eta_1(w)\}$.

$P \to 0P1P \mid 1P0P \mid \lambda$