Gramática

- Já vimos:
  - 2 métodos equivalentes para descrever linguagens:
    - expressão regular e autômatos finitos
  - as linguagens de interesse incluem restrições aos strings:
    - sintaxe – formas aceitáveis
- Gramática
  - é um sistema formal usado para descrever linguagens
  - especificação e implementação de linguagens de programação

Gramática livre de Contexto

- É um formalismo para se definir linguagens mais complexas que as linguagens regulares

- Formalmente é uma quádrupla G = (V, Σ, P, S):

  - V = conjunto de símbolos não-terminais (variáveis),
  - ∀ Σ = conjunto de símbolos terminais (tokens), $\Sigma \cap V = \varnothing$,
  - P = conjunto de regras ou produções, $P \subseteq V \times (V \cup \Sigma)^*$,
  - S = símbolo não-terminal inicial da gramática, $S \in V$.

Exemplo

- G_1: $S \rightarrow A\,B$

  $A \rightarrow a\,B$

  $A \rightarrow a$

  $B \rightarrow b\,A$

  $B \rightarrow b$

- G_2: $S \rightarrow A\,B$

  $A \rightarrow a\,B \mid a$

  $B \rightarrow b\,A \mid b$

Derivação

- Uma gramática (G) deriva um string w a partir de seu símbolo inicial S, trocando sucessivamente a variável pelo lado direito da produção.

  - $S \Rightarrow AB \Rightarrow aB \Rightarrow ab$
  - $S \rightarrow AB \qquad A \rightarrow a \qquad B \rightarrow b$

Derivação

- Derivação mais à direita: durante o processo de derivação o terminal expandido de cada forma sentencial é sempre o mais à direita.

- Derivação mais à esquerda: durante o processo de derivação o terminal expandido de cada forma sentencial é o mais à esquerda.

Mostre uma derivação mais à esquerda e uma mais à direita para o string ababaa em G:

$$S \rightarrow A\,A$$
$$A \rightarrow A\,A\,A \mid b\,A \mid A\,b \mid a$$

Árvore de Parse/Derivação

- É uma representação gráfica de uma derivação na qual a ordem das expansões não é considerada.

- Observações:
  - a raiz da árvore corresponde ao símbolo inicial da gramática
  - as folhas correspondem a símbolos terminais ou λ
  - os nodos internos correspondem a símbolos não-terminais

Árvore de Parse

- Forneça a árvore de parse para o string ababa dado G:

$$S \rightarrow A\,B$$

$$A \rightarrow a\,B \mid a$$

$$B \rightarrow b\,A \mid b$$

Ambigüidade

- Uma gramática que produz mais de uma árvore de derivação para um mesmo string é denominada ambígua.

- Exemplo:

  $S \rightarrow S\,a\,S \mid S\,b\,S \mid c \qquad w = cacbc$

- Gramáticas ambíguas não são adequadas para a construção de compiladores
  - uma linguagem livre de contexto (L) é inerentemente ambígua se toda gramática que gerar L for ambígua

Ambigüidade

- Exemplo: $S \rightarrow b\ S \mid S\ b \mid a$
  - que linguagem é gerada por esta gramática?
  - existe uma gramática sem ambigüidade para esta linguagem?

Notas

- $S \Rightarrow w$ lê-se "S deriva *w* em um passo"
- $S \overset{*}{\Rightarrow} w$ lê-se "S deriva *w* em zero ou mais passos"
- se $S \overset{*}{\Rightarrow} w \in (V \cup \Sigma)^*$, então w é uma forma sentencial de G
- se $S \overset{*}{\Rightarrow} w \in \Sigma^*$, então *w* é uma sentença de G

Notas

- A linguagem de GLC G: $L(G) = \{ w \in \Sigma^* \mid S \overset{*}{\Rightarrow} w \}$
- Uma linguagem definida por uma GLC é uma Linguagem Livre de Contexto (LLC)
    - uma linguagem L é livre de contexto se existe uma GLC G tal que L = L(G)

Propriedades de Fecho

- O conjunto das linguagens livres de contexto é fechado com relação às operações de união, concatenação e estrela de Kleene.

- Prova
  - considere $G_1 = (\Sigma, V_1, S_1, P_1)$ e $G_2 = (\Sigma, V_2, S_2, P_2)$, $V_1 \cap V_2 = \emptyset$, $L_1 = L(G_1)$ e $L_2 = L(G_2)$.

    1. $L_1 \cup L_2 = L(G)$, tal que $G = (\Sigma, V, S, P)$ onde:
       - $V = V_1 \cup V_2$
       - $P = \{ S \rightarrow S_1 \mid S_2 \} \cup P_1 \cup P_2$

Propriedades de Fecho

2. $L_1L_2 = L(G)$, tal que $G = (\Sigma, V, S, P)$ onde:
   - $V = V_1 \cup V_2$
   - $P = \{ S \rightarrow S_1S_2 \} \cup P_1 \cup P_2$

3. $L_1^* = L(G)$, tal que $G = (\Sigma, V, S, P)$ onde:
   - $V = V_1$
   - $P = \{ S \rightarrow S_1S \mid \lambda \} \cup P_1$

Exemplos de gramáticas - linguagens

- Seja G a gramática dada por:

$$S \rightarrow aSa \quad | \quad aBa$$
$$B \rightarrow bB \quad | \quad b$$

então: $L(G) = \{ a^n b^m a^n \mid n > 0, m > 0 \}$

- Sejam G_1, G_2 as gramáticas:

G_1: $S \rightarrow AB$
$A \rightarrow aA \mid a$
$B \rightarrow bB \mid \lambda$

G_2: $S \rightarrow aS \mid aB$
$B \rightarrow bB \mid \lambda$

então: $L(G_1) = L(G_2) = a^+ b^*$

Exemplos de gramáticas - linguagens

- Seja G a gramática dada por:

  $S \rightarrow abScB \mid \lambda$

  $B \rightarrow bB \mid b$

  então: $L(G) = \{ (ab)^n \ (cb^{m_n})^n \mid n \geq 0, m > 0 \}$

- Sejam G_1, G_2 as gramáticas:

  G_1: $S \rightarrow AbAbA$
  $A \rightarrow aA \mid \lambda$

  G_2: $S \rightarrow aS \mid bA$
  $A \rightarrow aA \mid bC$
  $C \rightarrow aC \mid \lambda$

  então: $L(G_1) = L(G_2) = a^*ba^*ba^*$

Gramáticas Regulares

- Uma gramática regular é uma gramática livre de contexto onde cada produção tem a seguinte forma:

  1. $A \rightarrow a$
  2. $A \rightarrow aB$
  3. $A \rightarrow \lambda$ $\quad$ $\mathbf{A, B \in V}$, $\mathbf{a \in \Sigma}$.

- Observação:
  - uma linguagem é dita regular se pode ser gerada por alguma gramática regular

Exemplo

- Vamos construir uma gramática regular que gere a linguagem da seguinte gramática:

$$G:\ S \rightarrow abSA \mid \lambda$$
$$A \rightarrow Aa \mid \lambda$$

$$L(G) = (ab)^+ a^* \cup \lambda$$

Solução:

$$S \rightarrow aB \mid \lambda$$
$$B \rightarrow bS \mid bA$$
$$A \rightarrow aA \mid \lambda$$

Exemplo

- Vamos construir uma gramática regular que gere a linguagem da seguinte gramática:

$$G: S \rightarrow abS \mid A \mid \lambda$$
$$A \rightarrow Aa \mid \lambda$$

$$L(G) = (ab)^* a^*$$

Solução:

$$S \rightarrow aB \mid \lambda \mid aA$$
$$B \rightarrow bS$$
$$A \rightarrow aA \mid \lambda$$