Gramáticas livres de contexto

Linguagens Formais e Autômatos

Prof. Daniel Saad Nogueira Nunes

IFB – Instituto Federal de Brasília, Campus Taguatinga

Sumário

Introdução

- Vimos que algumas linguagens **NÃO** são regulares, por exemplo:

$$L = \{0^n\#1^n \mid n \geq 0\}$$

- Precisamos de formalismos mais poderosos para trabalhar com as linguagens não regulares.
- As **gramáticas livres de contexto**, em inglês *context-free grammars (CFG)* são formalismos capazes de reconhecer uma classe de linguagens conhecida como **linguagens livres de contexto**, que por sua vez, contém a classe das linguagens regulares.

Introdução

- As CFGs inicialmente foram utilizadas para no estudo da linguagem natural, para entender o relacionamento sintático de estruturas textuais como substantivos, verbos, preposições, entre outros.
- Como lidam bem com características recursivas, as CFGs são interessantes para essa finalidade.
- Aplicações envolvendo projeto de linguagens de programação e tradutores utilizam esse formalismos para criar o analisador sinstático (*parser*) da linguagem.
- Algumas ferramentas inclusive conseguem gerar o código inteiro do analisador sintático partindo apenas da descrição da gramática.

Sumário

CFGs

Um exemplo de CFG

$$A \rightarrow 0A1$$
$$A \rightarrow B$$
$$B \rightarrow \#$$

Nela temos a presença de:

- **Variáveis**, ou símbolos **não-terminais**: A e B.

CFGs

Um exemplo de CFG

$$A \rightarrow 0A1$$
$$A \rightarrow B$$
$$B \rightarrow \#$$

Nela temos a presença de:

- **Símbolos terminais**: 0, $\#$ e 1.

CFGs

Um exemplo de CFG

$$A \rightarrow 0A1$$
$$A \rightarrow B$$
$$B \rightarrow \#$$

Nela temos a presença de:

- **Regras de produção**: uma regra de produção tem uma variável do lado esquerdo e uma sequência de variáveis ou terminais do lado direito. Essas regras especificam que o lado esquerdo pode ser **substituído** pelo que está do lado direito.

CFGs

Um exemplo de CFG

$$A \rightarrow 0A1$$
$$A \rightarrow B$$
$$B \rightarrow \#$$

Partindo do símbolo A, conseguimos gerar a palavra $000\#111$ com a seguinte sequência de substituições:

$$\mathrm{A} \Rightarrow \mathrm{0A1} \Rightarrow \mathrm{00A11} \Rightarrow \mathrm{000A111} \Rightarrow \mathrm{000B111} \Rightarrow 000\#111$$

CFGs

A árvore de derivações (ou de *parse*) que representa a derivação anterior corresponde à seguinte figura:

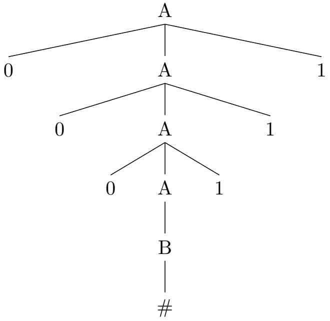


Figura: Árvore de derivações.

Sumário

Definição formal

Definição (CFG)

Uma CFG é uma 4-tupla $G = (V, \Sigma, R, S)$ em que:

- V é o conjunto finito de **variáveis**, ou **não-terminais**.
- Σ é o conjunto finito de **terminais**. $V \cap \Sigma = \emptyset$.
- $R = V \times (V \cup \Sigma)^*$ é o conjunto de **regras de produção**.
- $S \in V$ é a **variável inicial**.

Definição formal

Definição (Derivações)

Sejam u, v, w strings sobre $(V \cup \Sigma)^*$ e $A \to w$ uma regra da gramática.

Dizemos que $u\mathrm{A}v$ **produz** uwv, também escrito como $u\mathrm{A}v \Rightarrow uwv$.

Dizemos que u **deriva** v, escrito como $u \Rightarrow^* v$ se:

- $u = v$, ou;
- Existe uma sequência $u_1, u_2, \ldots, u_k$, tal que:

$$u \Rightarrow u_1 \Rightarrow u_2 \ldots \Rightarrow u_k \Rightarrow v$$

Definição formal

Definição (Geração de palavras)

Uma palavra de terminais $w \in \Sigma^*$ é **gerada** pela gramática $G = (V, \Sigma, R, S)$ quando, partindo do símbolo inicial S, é possível derivar w. Em outras palavras, G gera w se:

$$\mathrm{S} \Rightarrow^* w$$

Definição formal

Definição

Linguagem da gramática Seja $G = (V, \Sigma, R, S)$ uma gramática. $L(G)$ corresponde à linguagem da gramática e contém todas as palavras, formadas apenas por símbolos terminais, geradas por G. Em outras palavras:

$$L(G) = \{w \in \Sigma^* | S \Rightarrow^* w\}$$

Definição formal

Exemplo

Seja $G = (V, \Sigma, R, S)$ com $V = \{A, B\}$, $\Sigma = \{0, 1, \#\}$, $S = A$, e R sendo:

$$A \rightarrow 0A1$$
$$A \rightarrow B$$
$$B \rightarrow \#$$

Temos que $\mathrm{S} \Rightarrow^* 000\#111$, isto é, S gera $000\#111$ e que $L(G) = \{0^n\#1^n | n \geq 0\}$.

Definição formal

Notação

Caso tenhamos várias regras com a mesma variável do lado esquerdo, como em:

$$A \to 0A1$$
$$A \to B$$
$$B \to \#$$

Podemos escrevê-las em uma única linha, separando os lados direitos pelo símbolo de barra vertical, *e.g.*:

$$A \to 0A1 \mid B$$
$$B \to \#$$

CFGs

Exemplo

Tome a gramática $G = (V, \Sigma, R, S)$ com $V = \{S\}$, $\Sigma = \{(,)\}$, e as seguintes regras:

$$S \to (S) \mid SS \mid \epsilon$$

Essa gramática gera todas as palavras que consistem de parênteses balanceados como: $()$ $((()))$, $(()())$.
Observe que o lado direito de uma regra pode ser a palavra vazia.

CFGs

Exemplo

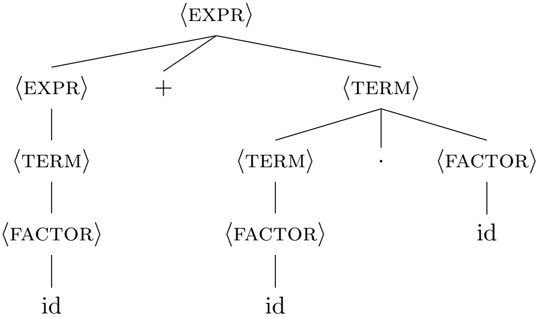
Considere a seguinte gramática $G = (V, \Sigma, R, \langle \text{EXPR} \rangle)$, com $\Sigma = \{\text{id}, +, \cdot, (,)\}$, $V = \{\langle \text{EXPR} \rangle, \langle \text{TERM} \rangle, \langle \text{FACTOR} \rangle\}$ e as seguintes regras:

$$\langle \text{EXPR} \rangle \rightarrow \langle \text{EXPR} \rangle + \langle \text{TERM} \rangle \mid \langle \text{TERM} \rangle$$
$$\langle \text{TERM} \rangle \rightarrow \langle \text{TERM} \rangle \cdot \langle \text{FACTOR} \rangle \mid \langle \text{FACTOR} \rangle$$
$$\langle \text{FACTOR} \rangle \rightarrow (\langle \text{EXPR} \rangle) \mid id$$

CFGs

Exemplo

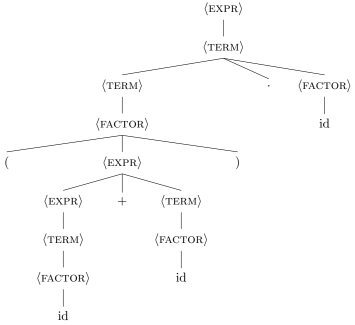
A palavra $\mathrm{id} + \mathrm{id} \cdot \mathrm{id}$ pode ser gerada de acordo com a seguinte árvore de derivações.



CFGs

Exemplo

A palavra $(\mathrm{id} + \mathrm{id}) \cdot \mathrm{id}$ pode ser gerada de acordo com a seguinte árvore de derivações.



CFGs

Definição (Ambiguidade)

Dependendo de como for projetada, uma gramática pode gerar uma mesma palavra de várias formas diferentes. Contudo isso não é desejado em algumas aplicações, como por exemplo compiladores, haja visto que um programa deve ter uma única interpretação pelo *parser*.

Uma gramática é dita **ambígua** quando ela consegue gerar uma palavra de formas diferentes.

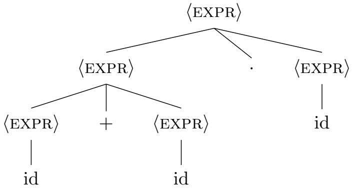
CFGs

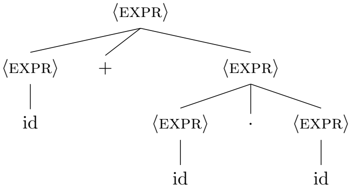
Exemplo

Considere a gramática $G = (V, \Sigma, R, \langle\text{EXPR}\rangle)$ com $\Sigma = \{+, \cdot, \text{id}\}$, $V = \{\langle\text{EXPR}\rangle\}$ e as seguintes regras de produção:

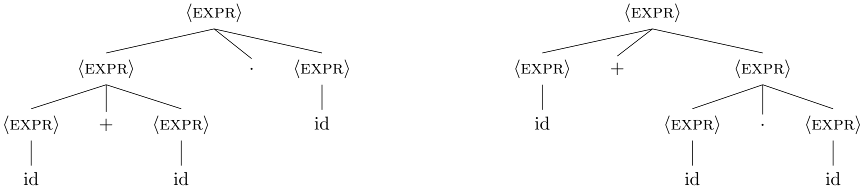
$$\langle\text{EXPR}\rangle \rightarrow \langle\text{EXPR}\rangle + \langle\text{EXPR}\rangle \mid \langle\text{EXPR}\rangle \cdot \langle\text{EXPR}\rangle \mid \langle\text{EXPR}\rangle \mid \text{id}$$

Ela consegue gerar a palavra $\text{id} + \text{id} \cdot \text{id}$ de duas formas diferentes, gerando duas avaliações da expressão matemática.





CFGs


⟨EXPR⟩
⟨EXPR⟩
·
⟨EXPR⟩
⟨EXPR⟩
+
⟨EXPR⟩
id
id
id
⟨EXPR⟩
⟨EXPR⟩
+
⟨EXPR⟩
id
⟨EXPR⟩
·
⟨EXPR⟩
id
id

CFGs

Definição

Derivação mais à esquerda Uma derivação de uma palavra w a partir de uma gramática G é uma **derivação mais à esquerda** se, a cada passo da derivação, a variável mais à esquerda é substituída.

Uma gramática é dita **ambígua** se gera uma mesma palavra a partir de duas ou mais derivações à esquerda distintas.

Sumário

CNF

- Para algumas aplicações, é importante que a gramática esteja na **forma normal de Chomsky** (CNF).
- Isso possibilita a aplicação de algoritmos que operam sobre essas gramáticas específicas.
- Felizmente, qualquer CFG pode ser convertida em uma gramática com essa propriedade.

CNF

Definição (Forma normal de Chomsky)

Uma CFG está na forma normal de Chomsky se toda regra de produção é da forma:

$$A \rightarrow BC$$
$$A \rightarrow a$$

Isto é, cada variável pode ser substituída por um terminal ou por outras duas variáveis. É importante que B e C não sejam o símbolo inicial da gramática.

Também é permitido que a existência da regra $S \rightarrow \epsilon$, sendo S o símbolo inicial.

CNF

Algoritmo para conversão na CNF

- Uma nova variável S_0 é criada e ela passa a ser o símbolo inicial da gramática. Além disso, a regra $S_0 \rightarrow S$ é adicionada à gramática. Essa mudança garantirá que o símbolo inicial nunca aparecerá do lado direito das regras.

CNF

Algoritmo para conversão na CNF

- Removemos toda regra do tipo $A \to \epsilon$, em que A não é a variável inicial. Para remover regras desse tipo sem prejuízo temos que criar algumas regras adicionais. Sempre que tivermos uma regra do tipo $R \to uAv$, adicionamos a regra $R \to uv$. Caso tenhamos uma regra do tipo $R \to a$, adicionamos a regra $R \to \epsilon$, **a não ser que** $R \to \epsilon$ já tenha sido eliminada anteriormente. Esse processo continua até que todas as regras do tipo $A \to \epsilon$ sejam eliminadas, com A não sendo o símbolo inicial.

CNF

Algoritmo para conversão na CNF

- Todas as regras unitárias, do tipo $A \to B$ são removidas. Para fazer isso, é preciso que, sempre que tivermos uma regra $B \to u$, adicionamos a regra $A \to u$, sendo $u \in (V \cup \Sigma)^*$. O processo é repetido até que não se tenha mais regras unitárias.

CNF

Algoritmo para conversão na CNF

- Finalmente, todas as regras que tenham do lado direito uma palavra com comprimento maior ou igual a 3 é substituída por regras adicionais. Isto é, caso tenhamos uma regra $A \to u_1 u_2 \ldots u_k$, substituímos elas pelas regras: $A \to u_1 A_1$, $A_1 \to u_2 A_2$,...,$A_{k-2} \to u_{k-1} u_k$ em que $A_1, \ldots, A_{k-2}$ são novas variáveis. Além disso, todo terminal u_i tem que ser substituído por uma variável nova $\mathrm{U_i}$ e a regra $U_i \to u_i$ deve ser criada.

CNF

Exercício

Converta a seguinte gramática $G = (V, \Sigma, R, S)$ para a CNF, em que $\Sigma = \{a, b\}$, $V = \{A, B, S\}$ e R:

$$S \rightarrow ASA \mid aB$$
$$A \rightarrow B \mid S$$
$$B \rightarrow b \mid \epsilon$$

CNF

Exercício

$$S \to ASA \mid aB$$
$$A \to B \mid S$$
$$B \to b \mid \epsilon$$

CNF

Exercício

1. Criação do novo símbolo inicial.

$$
\begin{aligned}
&S_0 \to S \\
&S \to ASA \mid aB \\
&A \to B \mid S \\
&B \to b \mid \epsilon
\end{aligned}
$$

CNF

Exercício

2.1 Eliminação da regra $B \to \epsilon$

$$
\begin{aligned}
S_0 &\to S \\
S &\to ASA \mid aB \\
A &\to B \mid S \\
B &\to b \mid \epsilon
\end{aligned}
$$

CNF

Exercício

2.1 Eliminação da regra $B \to \epsilon$

$$
\begin{aligned}
&S_0 \to S \\
&S \to ASA \mid aB \mid a \\
&A \to B \mid S \mid \epsilon \\
&B \to b
\end{aligned}
$$

CNF

Exercício

2.2 Eliminação da regra $A \rightarrow \epsilon$

$$
\begin{aligned}
&S_0 \rightarrow S \\
&S \rightarrow ASA \mid aB \mid a \\
&A \rightarrow B \mid S \mid \epsilon \\
&B \rightarrow b
\end{aligned}
$$

CNF

Exercício

2.2 Eliminação da regra $A \rightarrow \epsilon$

$$
\begin{aligned}
&S_0 \rightarrow S \\
&S \rightarrow ASA \mid aB \mid a \mid SA \mid AS \mid S \\
&A \rightarrow B \mid S \\
&B \rightarrow b
\end{aligned}
$$

CNF

Exercício

3.1 Eliminação da regra unitária $S \rightarrow S$

$$
\begin{aligned}
&S_0 \rightarrow S \\
&S \rightarrow ASA \mid aB \mid a \mid SA \mid AS \mid S \\
&A \rightarrow B \mid S \mid \\
&B \rightarrow b
\end{aligned}
$$

CNF

Exercício

3.1 Eliminação da regra unitária $S \rightarrow S$

$$S_0 \rightarrow S$$
$$S \rightarrow ASA \mid aB \mid a \mid SA \mid AS$$
$$A \rightarrow B \mid S$$
$$B \rightarrow b$$

CNF

Exercício

3.1 Eliminação da regra unitária $S_0 \rightarrow S$

$$
\begin{aligned}
S_0 &\rightarrow S \\
S &\rightarrow ASA \mid aB \mid a \mid SA \mid AS \\
A &\rightarrow B \mid S \\
B &\rightarrow b
\end{aligned}
$$

CNF

Exercício

3.1 Eliminação da regra unitária $S_0 \rightarrow S$

$$
\begin{aligned}
S_0 &\rightarrow ASA \mid aB \mid a \mid SA \mid AS \\
S &\rightarrow ASA \mid aB \mid a \mid SA \mid AS \\
A &\rightarrow B \mid S \\
B &\rightarrow b
\end{aligned}
$$

CNF

Exercício

3.2 Eliminação da regra unitária $A \to B$

$$
\begin{aligned}
S_0 &\to ASA \mid aB \mid a \mid SA \mid AS \\
S &\to ASA \mid aB \mid a \mid SA \mid AS \\
A &\to B \mid S \\
B &\to b
\end{aligned}
$$

CNF

Exercício

3.2 Eliminação da regra unitária $A \rightarrow B$

$$
\begin{aligned}
&S_0 \rightarrow ASA \mid aB \mid a \mid SA \mid AS \\
&S \rightarrow ASA \mid aB \mid a \mid SA \mid AS \\
&A \rightarrow S \mid {\color{red} b} \\
&B \rightarrow b
\end{aligned}
$$

CNF

Exercício

3.3 Eliminação da regra unitária $A \rightarrow S$

$$
\begin{aligned}
S_0 &\rightarrow ASA \mid aB \mid a \mid SA \mid AS \\
S &\rightarrow ASA \mid aB \mid a \mid SA \mid AS \\
A &\rightarrow S \mid b \\
B &\rightarrow b
\end{aligned}
$$

CNF

Exercício

3.3 Eliminação da regra unitária $A \rightarrow S$

$$S_0 \rightarrow ASA \mid aB \mid a \mid SA \mid AS$$
$$S \rightarrow ASA \mid aB \mid a \mid SA \mid AS$$
$$A \rightarrow b \mid ASA \mid aB \mid a \mid SA \mid AS$$
$$B \rightarrow b$$

CNF

Exercício

4. Conversão das regras com lado direito com comprimento maior ou igual a 3

$$
\begin{aligned}
S_0 &\rightarrow AA_1 \mid UB \mid a \mid SA \mid AS \\
S &\rightarrow AA_1 \mid UB \mid a \mid SA \mid AS \\
A &\rightarrow b \mid AA_1 \mid UB \mid a \mid SA \mid AS \\
A_1 &\rightarrow SA \\
U &\rightarrow a \\
B &\rightarrow b
\end{aligned}
$$