

Linguagens Formais e Autômatos

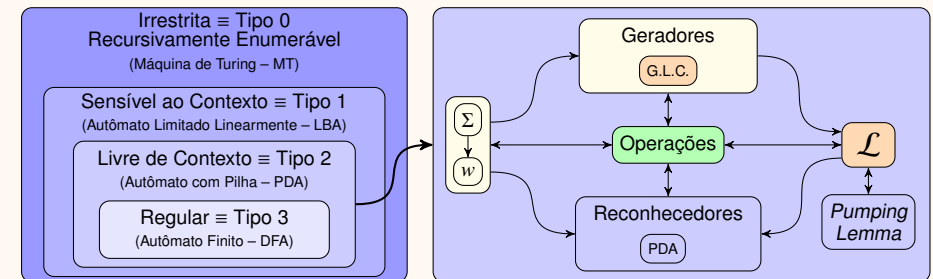
Humberto Longo

Instituto de Informática
Universidade Federal de Goiás

Bacharelado em Ciência da Computação, 2021/1



Roteiro



Derivação à esquerda

Teorema 1.29

- ▶ Seja $G = (V, \Sigma, P, S)$ uma GLC e $w \in \Sigma^*$. $w \in \mathcal{L}(G)$ se e somente se existe, a partir de S , uma derivação mais à esquerda de w .

Definição 1.30

- ▶ Derivação mais à esquerda: a variável mais à esquerda é a única a ser expandida a cada passo.
- ▶ Derivação mais à esquerda: a variável mais à esquerda na forma sentencial corrente é a única a ser expandida a cada passo.



Derivação à esquerda

Teorema 1.29

- ▶ Seja $G = (V, \Sigma, P, S)$ uma GLC e $w \in \Sigma^*$. $w \in \mathcal{L}(G)$ se e somente se existe, a partir de S , uma derivação mais à esquerda de w .

Demonstração.

- ⇐ Claramente, se existe, a partir de S , uma derivação de w mais à esquerda, então $w \in \mathcal{L}(G)$.
- ⇒ $w \in \mathcal{L}(G) \Rightarrow S \xRightarrow{*} w$ (derivação mais à esquerda).

□



Derivação à esquerda

Teorema 1.29

- ▶ Seja $G = (V, \Sigma, P, S)$ uma GLC e $w \in \Sigma^*$. $w \in \mathcal{L}(G)$ se e somente se existe, a partir de S , uma derivação mais à esquerda de w .

Demonstração.

- ▶ Se $w \in \mathcal{L}(G)$, então existe pelo menos uma derivação de w (não necessariamente mais à esquerda):

$$S \Rightarrow w_1 \Rightarrow w_2 \Rightarrow w_3 \Rightarrow \dots \Rightarrow w_n = w.$$

- ▶ Seja $w_k = u_1 A u_2 B u_3$ a primeira forma sentencial na derivação na qual não é feita a derivação mais à esquerda ($u_1 \in \Sigma^*$ e $u_2, u_3 \in (\Sigma \cup V)^*$).

□



Derivação à esquerda

Teorema 1.29

- ▶ Seja $G = (V, \Sigma, P, S)$ uma GLC e $w \in \Sigma^*$. $w \in \mathcal{L}(G)$ se e somente se existe, a partir de S , uma derivação mais à esquerda de w .

Demonstração.

- ▶ A aplicação, no passo $k + 1$, da regra de derivação $B \rightarrow v$ ($v \in (\Sigma \cup V)^*$), gera:

$$w_k = u_1 A u_2 B u_3 \Rightarrow u_1 A u_2 v u_3 = w_{k+1}.$$

- ▶ Suponha que a substituição de A , na derivação original, ocorre no passo $j + 1$.
- ▶ Logo, uso da regra de derivação $A \rightarrow p$ ($p \in (\Sigma \cup V)^*$) gera:

$$w_j = u_1 A q \Rightarrow u_1 p q = w_{j+1}.$$



Derivação à esquerda

Teorema 1.29

- ▶ Seja $G = (V, \Sigma, P, S)$ uma GLC e $w \in \Sigma^*$. $w \in \mathcal{L}(G)$ se e somente se existe, a partir de S , uma derivação mais à esquerda de w .

Demonstração.

- ▶ As regras de derivação aplicadas nos passos $k + 2$ até j transformam $u_2 v u_3$ em q .
- ▶ A derivação é completada com a subderivação:

$$w_{j+1} \xRightarrow{*} w_n = w.$$

- ▶ Uma derivação de w , que é mais à esquerda para as $k + 1$ primeiras aplicações de regras de derivação, pode ser agora obtida (os k primeiros passos já eram mais à esquerda).

□



Derivação à esquerda

Teorema 1.29

- ▶ Seja $G = (V, \Sigma, P, S)$ uma GLC e $w \in \Sigma^*$. $w \in \mathcal{L}(G)$ se e somente se existe, a partir de S , uma derivação mais à esquerda de w .

Derivação original.

$$\begin{array}{ll} S \xRightarrow{k} w_k = u_1 A u_2 B u_3 & (u_1 \in \Sigma^* \text{ e } u_2, u_3 \in (\Sigma \cup V)^*) \\ \Rightarrow w_{k+1} = u_1 A u_2 v u_3 & (\text{Regra } B \rightarrow v, v \in (\Sigma \cup V)^*) \\ \xRightarrow{j-(k+1)} w_j = u_1 A q & (\text{Derivação } u_2 v u_3 \xRightarrow{*} q, q \in (\Sigma \cup V)^*) \\ \Rightarrow w_{j+1} = u_1 p q & (\text{Regra } A \rightarrow p, p \in (\Sigma \cup V)^*) \\ \xRightarrow{n-(j+1)} w_n & (\text{Derivação } w_{j+1} \xRightarrow{*} w_n, w_n \in \Sigma^*) \end{array}$$

□



Derivação à esquerda

Teorema 1.29

- Seja $G = (V, \Sigma, P, S)$ uma GLC e $w \in \Sigma^*$. $w \in \mathcal{L}(G)$ se e somente se existe, a partir de S , uma derivação mais à esquerda de w .

Derivação mais à esquerda.

$$\begin{aligned}
 S &\xRightarrow{k} w_k = u_1 A u_2 B u_3 && (u_1 \in \Sigma^* \text{ e } u_2, u_3 \in (\Sigma \cup V)^*) \\
 &\Rightarrow w_{k+1} = u_1 p u_2 B u_3 && (\text{Regra } A \rightarrow p, p \in (\Sigma \cup V)^*) \\
 &\Rightarrow w_{k+2} = u_1 p u_2 v u_3 && (\text{Regra } B \rightarrow v, v \in (\Sigma \cup V)^*) \\
 &\xRightarrow{j+1-(k+2)} w_{j+1} = u_1 p q && (\text{Derivação } u_2 v u_3 \xRightarrow{*} q, q \in (\Sigma \cup V)^*) \\
 &\xRightarrow{n-(j+1)} w_n && (\text{Derivação } w_{j+1} \xRightarrow{*} w_n, w_n \in \Sigma^*)
 \end{aligned}$$

□



Derivação à esquerda

Teorema 1.29

- Seja $G = (V, \Sigma, P, S)$ uma GLC e $w \in \Sigma^*$. $w \in \mathcal{L}(G)$ se e somente se existe, a partir de S , uma derivação mais à esquerda de w .

Demonstração.

- Se o comprimento de uma derivação é n , então com no máximo n utilizações do procedimento descrito qualquer derivação torna-se mais à esquerda.

□



Derivação à esquerda

- O Teorema 1.29 não garante que qualquer forma sentencial pode ser gerada por uma derivação mais à esquerda.
- A garantia é válida apenas para cadeias de símbolos terminais.

Exemplo 1.31

$$\begin{aligned}
 S &\rightarrow AB \\
 A &\rightarrow aA \mid \varepsilon \\
 B &\rightarrow bB \mid \varepsilon
 \end{aligned}$$

- Forma sentencial A é gerada por: $S \Rightarrow AB \Rightarrow A$.
- Não existe derivação mais à esquerda para A .



Derivação à esquerda

- Uso apenas da derivação mais à esquerda elimina várias possibilidades de derivação de uma cadeia.
 - Há a garantia de que, dada uma cadeia da linguagem de uma gramática, existe apenas uma derivação mais à esquerda?
 - Não!

Exemplo 1.32

$$\begin{array}{ll}
 S \Rightarrow AA & S \Rightarrow AA \\
 \Rightarrow aA & \Rightarrow AAAA \\
 \Rightarrow aAAA & \Rightarrow aAAA \\
 \Rightarrow abAAA & \Rightarrow abAAA \\
 \Rightarrow abaAA & \Rightarrow abaAA \\
 \Rightarrow ababAA & \Rightarrow ababAA \\
 \Rightarrow ababaA & \Rightarrow ababaA \\
 \Rightarrow ababaa & \text{ou} \Rightarrow ababaa
 \end{array}$$

$$P = \left\{ S \rightarrow AA, \right. \\
 \left. A \rightarrow AAA \mid bA \mid Ab \mid a \right\}$$



Ambiguidade

Definição 1.33

Uma gramática livre de contexto G é ambígua se existe uma cadeia $w \in \mathcal{L}(G)$ para a qual existem duas derivações mais à esquerda distintas.

Exemplo 1.34

- ▶ $G = (\{S\}, \{a\}, P, S)$, onde: $P = \{S \rightarrow aS \mid Sa \mid a\}$.
- ▶ G é ambígua, pois aa pode ser derivado à esquerda de duas formas:

$$\begin{array}{ccc} S \Rightarrow aS & & S \Rightarrow Sa \\ \Rightarrow aa & \text{ou} & \Rightarrow aa \end{array}$$

- ▶ Gramática não ambígua equivalente a $G : \{S \rightarrow aS \mid a\}$.



Ambiguidade

- ▶ Ambiguidade é uma propriedade de gramáticas e não de linguagens.
- ▶ Quando uma gramática é ambígua, “**geralmente**”, é possível construir outra gramática não ambígua equivalente.
- ▶ Existem linguagens livres de contexto que só podem ser geradas por gramáticas ambíguas.
 - ▶ LLC inerentemente ambígua.
 - ▶ Exemplo: $\mathcal{L} = \{a^n b^n c^m d^m \mid n \geq 1, m \geq 1\} \cup \{a^n b^m c^m d^n \mid n \geq 1, m \geq 1\}$,



Ambiguidade

Exemplo 1.35

- ▶ $G = (\{S\}, \{a, b\}, P, S)$, onde $P = \{S \rightarrow bS \mid Sb \mid a\}$.

- ▶ G é ambígua, pois:

$$\begin{array}{ccc} S \Rightarrow bS & & S \Rightarrow Sb \\ \Rightarrow bSb & & \Rightarrow bSb \\ \Rightarrow bab & & \Rightarrow bab \end{array}$$

- ▶ Gramáticas não ambíguas equivalentes a G :

$$G_1 : \left\{ \begin{array}{l} S \rightarrow bS \mid aA, \\ A \rightarrow bA \mid \varepsilon \end{array} \right\} \quad G_2 : \left\{ \begin{array}{l} S \rightarrow bS \mid A, \\ A \rightarrow Ab \mid a \end{array} \right\}$$

- ▶ Derivação mais à esquerda, em G_1 , da cadeia $b^n ab^m$:

$$\underbrace{S \xRightarrow{n} b^n S}_{S \rightarrow bS} \xRightarrow{S \rightarrow aA} \underbrace{b^n aA}_{S \rightarrow aA} \xRightarrow{A \rightarrow bA} \underbrace{b^n ab^m A}_{A \rightarrow bA} \xRightarrow{A \rightarrow \varepsilon} \underbrace{b^n ab^m}_{A \rightarrow \varepsilon}.$$



Ambiguidade

Exemplo 1.36

- ▶ $G = (\{S\}, \{a, b\}, P, S)$, onde $P = \{S \rightarrow aSb \mid aSbb \mid \varepsilon\}$.

- ▶ $\mathcal{L}(G) = \{a^n b^m \mid 0 \leq n \leq m \leq 2n\}$.

- ▶ G é ambígua, pois:

$$\begin{array}{ccc} S \Rightarrow aSb & & S \Rightarrow aSbb \\ \Rightarrow aaSbbb & & \Rightarrow aaSbbb \\ \Rightarrow aabbbb & \text{ou} & \Rightarrow aabbbb \end{array}$$

- ▶ Gramáticas não ambígua equivalente a G :

$$G_1 : \left\{ \begin{array}{l} S \rightarrow aSb \mid A \mid \varepsilon, \\ A \rightarrow aAbb \mid abb \end{array} \right\}$$



Árvore de derivação

Definição 1.37

- Seja $G = (V, \Sigma, P, S)$ uma GLC e $S \xRightarrow{*} w$ uma derivação em G . A Árvore de Derivação \mathcal{A}_d de $S \xRightarrow{*} w$ pode ser construída como segue:
1. Inicie \mathcal{A}_d com a raiz S .
 2. Se $A \rightarrow x_1 x_2 \dots x_n$, com $x_i \in (V \cup \Sigma)$, é a regra de derivação aplicada à forma sentencial uAv , então acrescente x_1, x_2, \dots, x_n como filhos de A em \mathcal{A}_d .
 3. Se $A \rightarrow \varepsilon$ é a regra de derivação aplicada à forma sentencial uAv , então acrescente ε como único filho de A em \mathcal{A}_d .
 4. Se $A \rightarrow a$, $a \in \Sigma$, é a regra de derivação aplicada à forma sentencial uAv , então acrescente a como único filho de A em \mathcal{A}_d .



Derivação como uma árvore

Exemplo 1.38 (Derivação)

$S \Rightarrow AA$
 $\Rightarrow aA$
 $\Rightarrow aAAA$
 $\Rightarrow abAAA$
 $\Rightarrow abaAA$
 $\Rightarrow ababAA$
 $\Rightarrow ababaA$
 $\Rightarrow ababaa$

Exemplo 1.38 (Árvore de derivação)

S

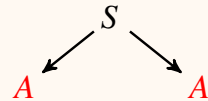


Derivação como uma árvore

Exemplo 1.38 (Derivação)

$S \Rightarrow AA$
 $\Rightarrow aA$
 $\Rightarrow aAAA$
 $\Rightarrow abAAA$
 $\Rightarrow abaAA$
 $\Rightarrow ababAA$
 $\Rightarrow ababaA$
 $\Rightarrow ababaa$

Exemplo 1.38 (Árvore de derivação)

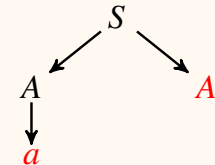


Derivação como uma árvore

Exemplo 1.38 (Derivação)

$S \Rightarrow AA$
 $\Rightarrow aA$
 $\Rightarrow aAAA$
 $\Rightarrow abAAA$
 $\Rightarrow abaAA$
 $\Rightarrow ababAA$
 $\Rightarrow ababaA$
 $\Rightarrow ababaa$

Exemplo 1.38 (Árvore de derivação)

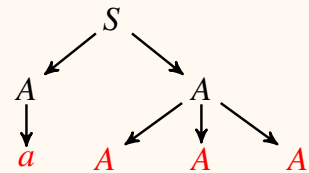


Derivação como uma árvore

Exemplo 1.38 (Derivação)

$S \Rightarrow AA$
 $\Rightarrow aA$
 $\Rightarrow aAAA$
 $\Rightarrow abAAA$
 $\Rightarrow abaAA$
 $\Rightarrow ababAA$
 $\Rightarrow ababaA$
 $\Rightarrow ababaa$

Exemplo 1.38 (Árvore de derivação)

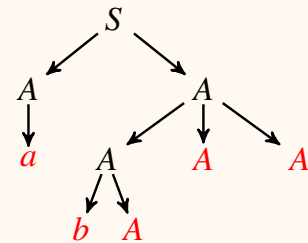


Derivação como uma árvore

Exemplo 1.38 (Derivação)

$S \Rightarrow AA$
 $\Rightarrow aA$
 $\Rightarrow aAAA$
 $\Rightarrow abAAA$
 $\Rightarrow abaAA$
 $\Rightarrow ababAA$
 $\Rightarrow ababaA$
 $\Rightarrow ababaa$

Exemplo 1.38 (Árvore de derivação)

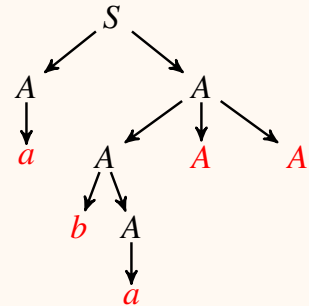


Derivação como uma árvore

Exemplo 1.38 (Derivação)

$S \Rightarrow AA$
 $\Rightarrow aA$
 $\Rightarrow aAAA$
 $\Rightarrow abAAA$
 $\Rightarrow abaAA$
 $\Rightarrow ababAA$
 $\Rightarrow ababaA$
 $\Rightarrow ababaa$

Exemplo 1.38 (Árvore de derivação)

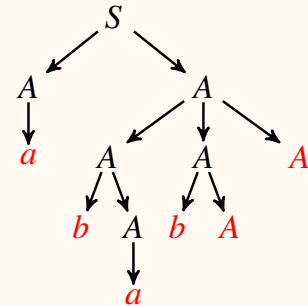


Derivação como uma árvore

Exemplo 1.38 (Derivação)

$S \Rightarrow AA$
 $\Rightarrow aA$
 $\Rightarrow aAAA$
 $\Rightarrow abAAA$
 $\Rightarrow abaAA$
 $\Rightarrow ababAA$
 $\Rightarrow ababaA$
 $\Rightarrow ababaa$

Exemplo 1.38 (Árvore de derivação)

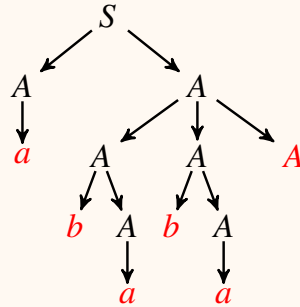


Derivação como uma árvore

Exemplo 1.38 (Derivação)

$S \Rightarrow AA$
 $\Rightarrow aA$
 $\Rightarrow aAAA$
 $\Rightarrow abAAA$
 $\Rightarrow abaAA$
 $\Rightarrow ababAA$
 $\Rightarrow ababaA$
 $\Rightarrow ababaa$

Exemplo 1.38 (Árvore de derivação)

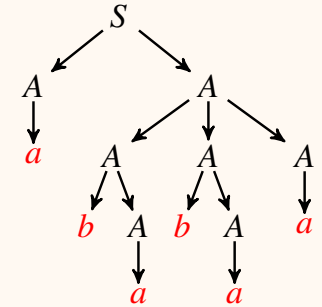


Derivação como uma árvore

Exemplo 1.38 (Derivação)

$S \Rightarrow AA$
 $\Rightarrow aA$
 $\Rightarrow aAAA$
 $\Rightarrow abAAA$
 $\Rightarrow abaAA$
 $\Rightarrow ababAA$
 $\Rightarrow ababaA$
 $\Rightarrow ababaa$

Exemplo 1.38 (Árvore de derivação)



Árvore de derivação

- ▶ A ordem das folhas independe da derivação a partir da qual a árvore foi gerada.
- ▶ Derivações distintas podem gerar a mesma árvore de derivação.
- ▶ Uma árvore de derivação pode ser usada para gerar diversas derivações que geram a mesma cadeia.



Ambiguidade

- ▶ Há uma correspondência natural (um para um) entre derivações mais a esquerda e árvores de derivação.
 - ▶ Definição recursiva de árvore de derivação especifica a construção de uma árvore a partir de uma derivação mais esquerda.
 - ▶ Apenas uma derivação mais à esquerda de uma cadeia w pode ser extraída de uma árvore de derivação de w .

Definição 1.39

Uma gramática G é ambígua se existe uma cadeia w em $\mathcal{L}(G)$ cujos símbolos são as folhas de duas árvores de derivação distintas.

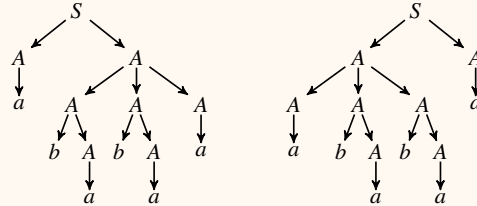


Árvores de derivação distintas

Exemplo 1.40

- $G = (V = \{S, A\}, \Sigma = \{a, b\}, P, S)$, com $P = \left\{ \begin{array}{l} S \rightarrow AA, \\ A \rightarrow AAA \mid bA \mid Ab \mid a \end{array} \right\}$, e $w = ababaa$:

$S \Rightarrow A$	$S \Rightarrow AA$
$\Rightarrow aA$	$\Rightarrow AAAA$
$\Rightarrow aAAA$	$\Rightarrow aAAA$
$\Rightarrow abAAA$	$\Rightarrow abAAA$
$\Rightarrow abaAA$	$\Rightarrow abaAA$
$\Rightarrow ababAA$	$\Rightarrow ababAA$
$\Rightarrow ababaA$	$\Rightarrow ababaA$
$\Rightarrow ababaa$	$\Rightarrow ababaa$



Uma gramática regular pode ser ambígua?

(Marcos V. R. Oliveira – LFA-2018/1)

Exemplo 1.41

- Uma gramática regular ambígua é $G = (V = \{S, A, B\}, \Sigma = \{a\}, P, S)$, com

$$P = \left\{ \begin{array}{l} S \rightarrow aA \mid aB \\ A \rightarrow \varepsilon \\ B \rightarrow \varepsilon \end{array} \right\}.$$

- Toda gramática regular $G = (V, \Sigma, P, S)$ que contém uma regra de derivação da forma $A \rightarrow aB$ (com $A, B \in V$ e $a \in \Sigma$) que seja alcançável a partir do símbolo inicial S , possui uma gramática regular ambígua equivalente.
- Para construir uma tal gramática basta inserir um novo símbolo não terminal X e adicionar a regra $A \rightarrow aX$ e replicar as regras do B , com a substituição do B pelo X .



Uma gramática regular pode ser ambígua?

(Marcos V. R. Oliveira – LFA-2018/1)

Exemplo 1.42

- $G = (V = \{S, A, B\}, \Sigma = \{a, b\}, P, S)$, com $P = \left\{ \begin{array}{l} S \rightarrow aS \mid bA \\ A \rightarrow bA \mid aB \mid \varepsilon \\ B \rightarrow aB \mid \varepsilon \end{array} \right\}$.

- Gramática ambígua equivalente é $G_1 = (V_1 = \{S, A, B, X\}, \Sigma = \{a, b\}, P_1, S)$, com

$$P = \left\{ \begin{array}{l} S \rightarrow aS \mid bA \\ A \rightarrow bA \mid aB \mid aX \mid \varepsilon \\ B \rightarrow aB \mid \varepsilon \\ X \rightarrow aX \mid \varepsilon \end{array} \right\}.$$



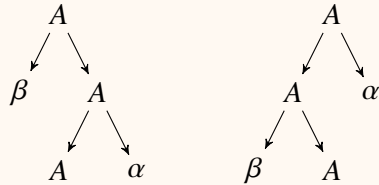
Ambiguidade – Resumo

- Derivação ambígua: uma mesma cadeia tem mais de uma árvore de derivação.
- Gramática ambígua: gera uma cadeia por derivação ambígua.
- A cadeia tem duas árvores de derivação.
 - Duas derivações podem diferir na ordem em que variáveis são substituídas.
- Uma cadeia w é derivada ambigualmente se possui duas ou mais derivações à esquerda.
- Uma gramática é ambígua se gera alguma cadeia ambigualmente.
- LLC inerentemente ambígua: só pode ser gerada por uma gramática ambígua.
- Ex.: $L = \{0^i 1^j 2^k \mid i = j \text{ ou } j = k\}$.



Detecção de ambiguidade

- ▶ Não há algoritmo para decidir se uma GLC $G = (V, \Sigma, P, S)$ arbitrária é ambígua!
- ▶ Contudo, se um símbolo $A \in V$ é recursivo à esquerda ($A \xRightarrow{+} A\alpha, |\alpha| \geq 0$) e recursivo à direita ($A \xRightarrow{+} \beta A, |\beta| \geq 0$), então G é ambígua, desde que G seja “reduzida”, ou seja, não possua símbolos redundantes.
- ▶ Exemplo: $A \xRightarrow{*} \beta A \alpha$.



Detecção de ambiguidade

Gramática de operadores

- ▶ Falta de precedência entre os operadores (por exemplo, entre $+$ e $*$).
- ▶ Falta de agrupamento em sequências de operadores (por exemplo, $E + E + E$ pode significar $E + (E + E)$ ou $(E + E) + E$).
- ▶ Solução: introduzir mais variáveis, cada uma representando um tipo de expressão.



Remoção de ambiguidade

Exemplo 1.43

- ▶ $G = (\{E\}, \{+, *, (,), id\}, \{E \rightarrow E + E \mid E * E \mid (E) \mid id\}, E)$.

1. Forçar maior precedência para o símbolo $*$:








$$\begin{aligned} E &\rightarrow E + E \mid T \\ T &\rightarrow T * T \mid (E) \mid id \end{aligned}$$

2. Eliminar recursões à direita:

$$\begin{aligned} E &\rightarrow E + T \mid T \\ T &\rightarrow T * id \mid T * (E) \mid (E) \mid id \end{aligned}$$



Livros texto

-  R. P. Grimaldi
Discrete and Combinatorial Mathematics – An Applied Introduction.
Addison Wesley, 1994.
-  D. J. Velleman
How To Prove It – A Structured Approach.
Cambridge University Press, 1996.
-  J. E. Hopcroft; J. Ullman.
Introdução À Teoria de Autômatos, Linguagens e Computação.
Ed. Campus.
-  T. A. Sudkamp.
Languages and Machines – An Introduction to the Theory of Computer Science.
Addison Wesley Longman, Inc. 1998.
-  J. Carroll; D. Long.
Theory of Finite Automata – With an Introduction to Formal Languages.
Prentice-Hall, 1989.
-  M. Sipser.
Introduction to the Theory of Computation.
PWS Publishing Company, 1997.
-  H. R. Lewis; C. H. Papadimitriou
Elementos de Teoria da Computação.
Bookman, 2000.

