ACH2043
INTRODUÇÃO À TEORIA DA COMPUTAÇÃO

Aula 10

Cap. 2.1 – Gramáticas Livres de Contexto

Profa. Ariane Machado Lima
ariane.machado@usp.br

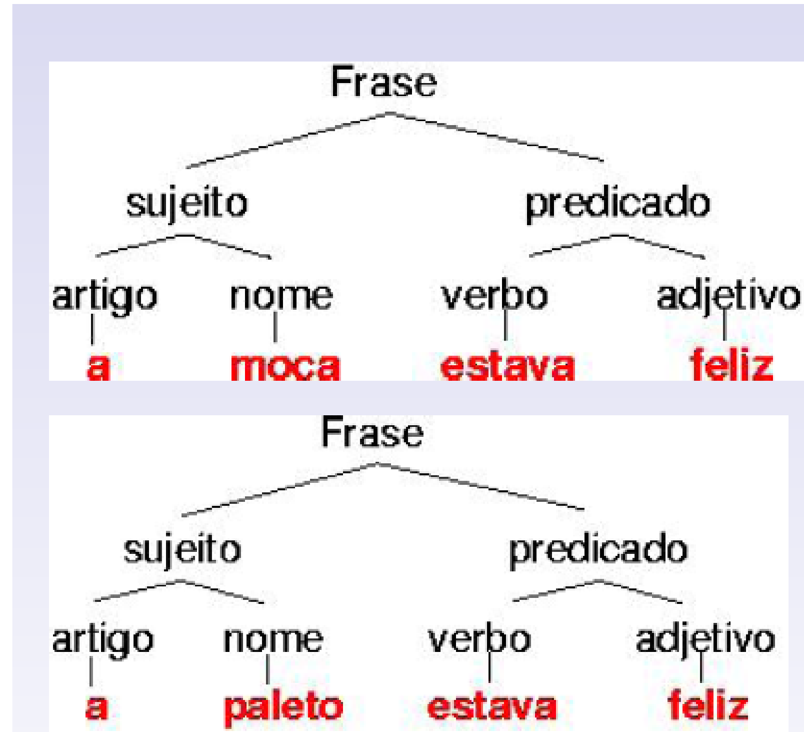
Aula anterior

Gramáticas

conjunto de produções

símbolo inicial

Frase	→	sujeito	predicado
sujeito	→	artigo	nome
artigo	→	a	
artigo	→	o	
nome	→	paletó	
nome	→	moça	
nome	→	dia	
predicado	→	verbo	adjetivo
verbo	→	é	
verbo	→	estava	
adjectivo	→	feliz	
adjectivo	→	azul	



símbolos não-terminais

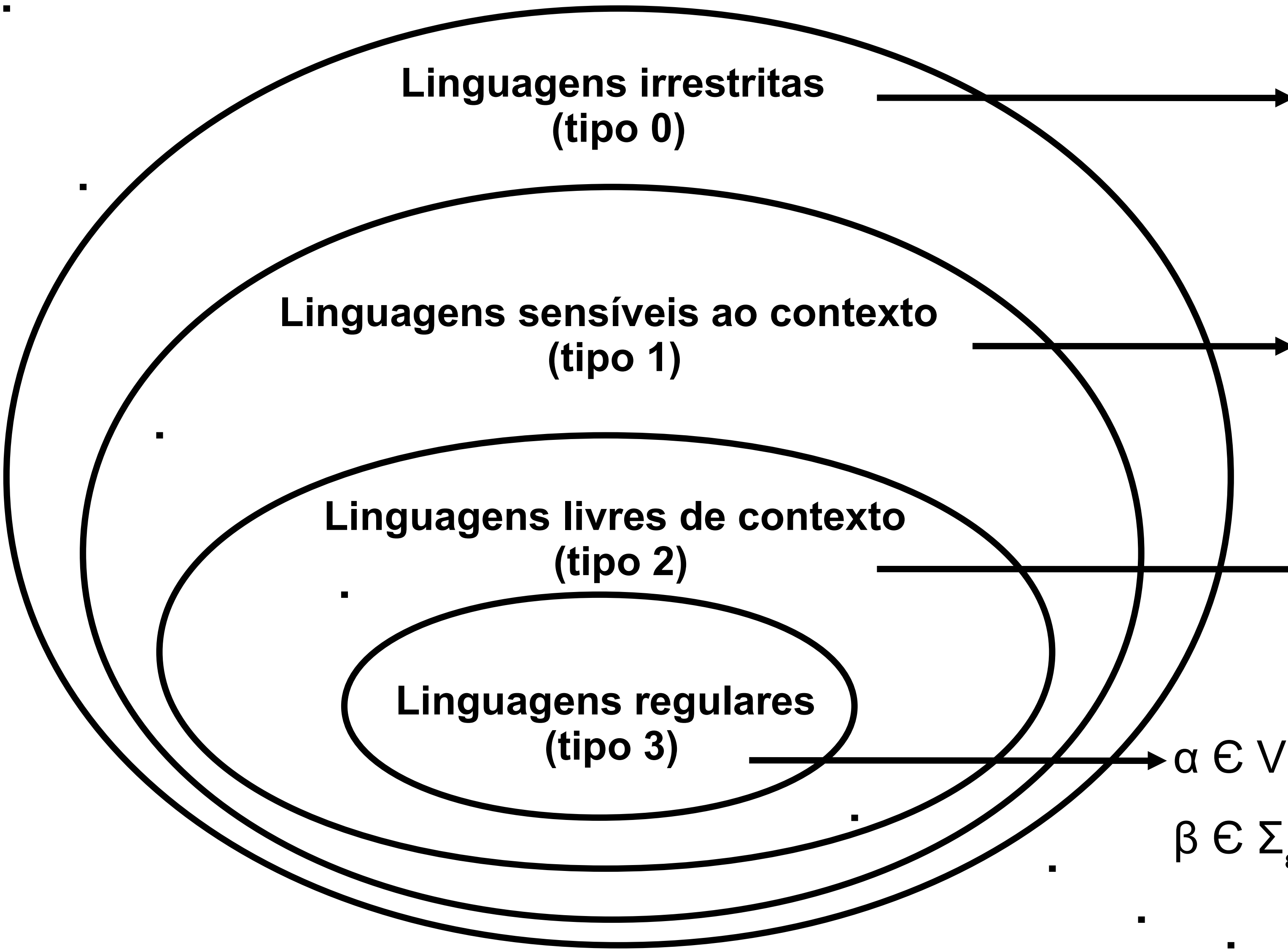
símbolos terminais

Gramáticas

- Definição: uma gramática G é uma quádrupla (V, Σ, S, P), onde
  - V é o conjunto de símbolos não-terminais (variáveis)
  - Σ é o conjunto de símbolos terminais
  - S é o símbolo inicial
  - P é o conjunto de produções da forma

    $(\Sigma \cup V)^* \, V \, (\Sigma \cup V)^* \rightarrow (\Sigma \cup V)^*$

Hierarquia de Chomsky $\alpha \rightarrow \beta$



$\alpha \in (V \cup \Sigma)^* V (V \cup \Sigma)^*$
$\beta \in (V \cup \Sigma)^*$

$\alpha \in (V \cup \Sigma)^* V (V \cup \Sigma)^*$
$\beta \in (V \cup \Sigma)^*$
$|\alpha| <= |\beta|$

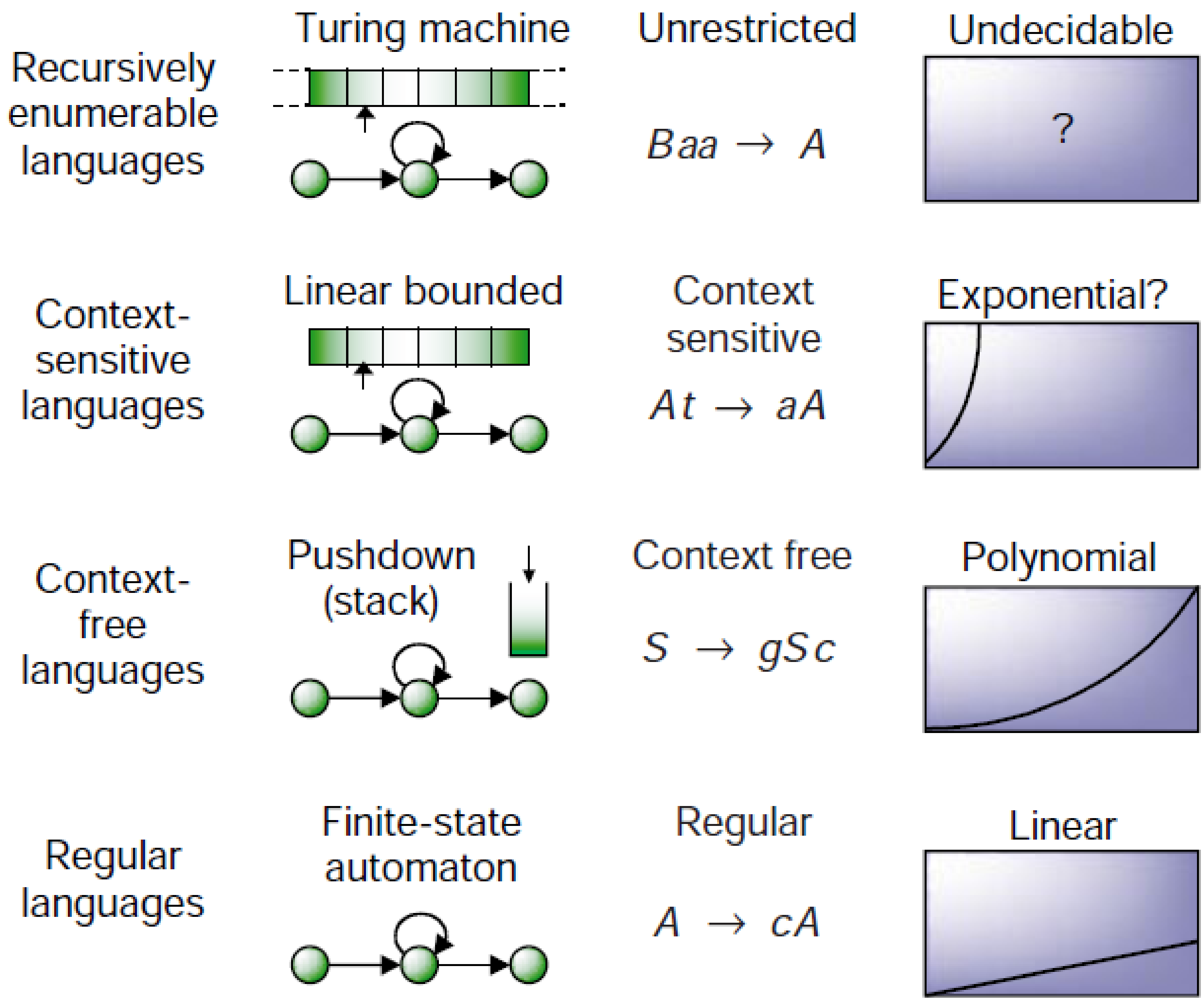
$\alpha \in V$

$\beta \in (V \cup \Sigma)^*$

$\alpha \in V$

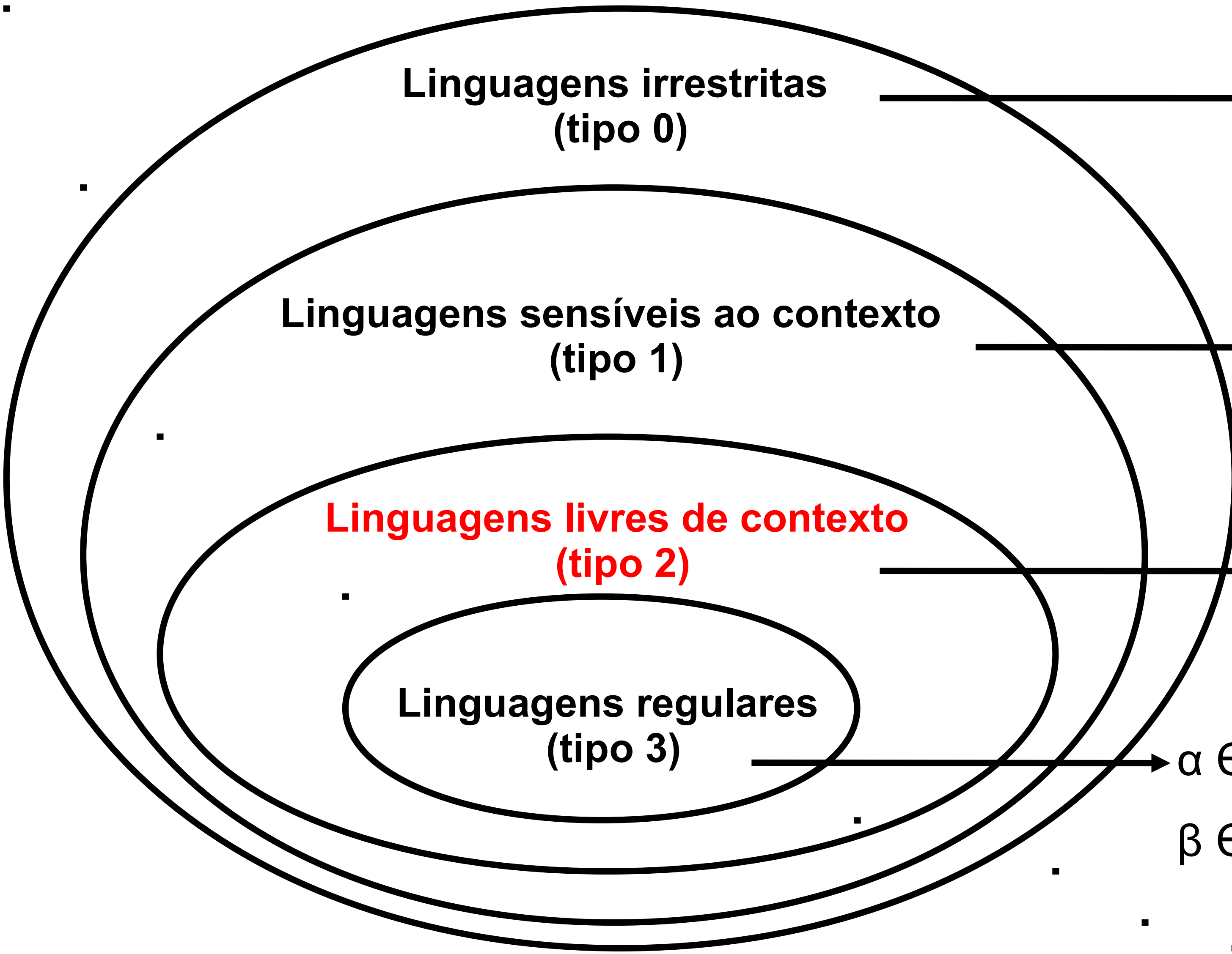
$\beta \in \Sigma_\varepsilon$, $\beta \in V$, $\beta \in$ ($V\Sigma$ou ΣV)

Linguagens, dispositivos, gramáticas e complexidades



Hierarquia de Chomsky

$\alpha \rightarrow \beta$



$\alpha \in (V \cup \Sigma)^* V (V \cup \Sigma)^*$
$\beta \in (V \cup \Sigma)^*$

$\alpha \in (V \cup \Sigma)^* V (V \cup \Sigma)^*$
$\beta \in (V \cup \Sigma)^*$
$|\alpha| <= |\beta|$

$\alpha \in V$

$\beta \in (V \cup \Sigma)^*$

$\alpha \in V$

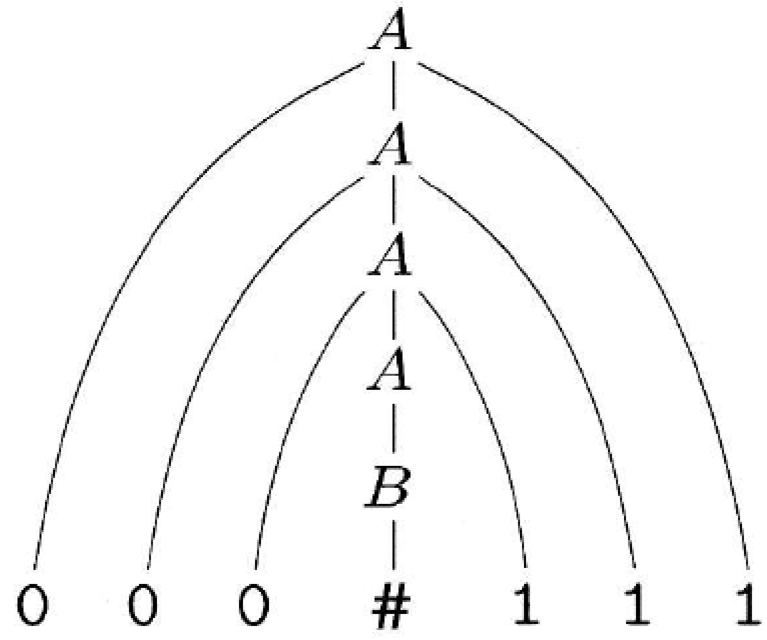
$\beta \in \Sigma_\varepsilon$, $\beta \in V$, $\beta \in$ ($V\Sigma$ ou ΣV)

Gramáticas Livres de Contexto

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

Por exemplo, a gramática G_1 gera a cadeia 000#111.

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



Árvore sintática
ou
Árvore de derivação

Dicas para projetar gramáticas livres de contexto

- É a união de linguagens mais simples?

Por exemplo, para obter uma gramática para a linguagem $\{0^n1^n \mid n \geq 0\} \cup \{1^n0^n \mid n \geq 0\}$, primeiro construa a gramática

$$S_1 \rightarrow 0S_11 \mid \varepsilon$$

para a linguagem $\{0^n1^n \mid n \geq 0\}$ e a gramática

$$S_2 \rightarrow 1S_20 \mid \varepsilon$$

para a linguagem $\{1^n0^n \mid n \geq 0\}$ e então adicione a regra $S \rightarrow S_1 \mid S_2$ para dar a gramática

$$\begin{aligned} S &\rightarrow S_1 \mid S_2 \\ S_1 &\rightarrow 0S_11 \mid \varepsilon \\ S_2 &\rightarrow 1S_20 \mid \varepsilon\ . \end{aligned}$$

Dicas para projetar gramáticas livres de contexto

- Definições recursivas

EXEMPLO 2.3

Considere a gramática $G_3 = (\{S\}, \{\mathtt{a}, \mathtt{b}\}, R, S)$. O conjunto de regras, R, é

$$S \to \mathtt{a}S\mathtt{b} \mid SS \mid \varepsilon.$$

Aula de hoje…

mais sobre Gramáticas Livres de Contexto

Derivações

- É possível que uma mesma cadeia possua mais de uma derivação
- **Derivação mais à esquerda** (sempre o primeiro símbolo não terminal da forma sentencial é substituído primeiro)
- **Derivação mais à direita** (sempre o último símbolo não terminal da forma sentencial é substituído primeiro)

Derivações

Gramática:

1 $S \rightarrow S ; S$	4 $E \rightarrow \text{id}$	8 $L \rightarrow E$
2 $S \rightarrow \text{id} := E$	5 $E \rightarrow \text{num}$	9 $L \rightarrow L , E$
3 $S \rightarrow \text{print} (L)$	6 $E \rightarrow E + E$	
	7 $E \rightarrow (S , E)$	

Cadeia: `id := num; id := id + (id := num + num, id)`

$\underline{S}$

$S ; \underline{S}$

$\underline{S} ; \text{id} := E$

$\text{id} := \underline{E} ; \text{id} := E$

$\text{id} := \text{num} ; \text{id} := \underline{E}$

$\text{id} := \text{num} ; \text{id} := E + \underline{E}$

$\text{id} := \text{num} ; \text{id} := \underline{E} + (S , E)$

$\text{id} := \text{num} ; \text{id} := \text{id} + (\underline{S} , E)$

$\text{id} := \text{num} ; \text{id} := \text{id} + (\text{id} := \underline{E} , E)$

$\text{id} := \text{num} ; \text{id} := \text{id} + (\text{id} := E + E , \underline{E})$

$\text{id} := \text{num} ; \text{id} := \text{id} + (\text{id} := \underline{E} + E , \text{id})$

$\text{id} := \text{num} ; \text{id} := \text{id} + (\text{id} := \text{num} + \underline{E} , \text{id})$

$\text{id} := \text{num} ; \text{id} := \text{id} + (\text{id} := \text{num} + \text{num} , \text{id})$

Uma derivação possível

Mais à esquerda ou mais à direira?

Derivações

Gramática:

1 $S \rightarrow S \; ; \; S$	4 $E \rightarrow \mathrm{id}$	8 $L \rightarrow E$
2 $S \rightarrow \mathrm{id} \; := E$	5 $E \rightarrow \mathrm{num}$	9 $L \rightarrow L \; , \; E$
3 $S \rightarrow \mathrm{print} \; (\; L \;)$	6 $E \rightarrow E + E$	
	7 $E \rightarrow (S \; , \; E \;)$	

Cadeia: `id := num; id := id + (id := num + num, id)`

$$
\begin{array}{l}
\underline{S} \\
S \; ; \; \underline{S} \\
\underline{S} \; ; \; \mathrm{id} \; := E \\
\mathrm{id} \; := \underline{E} \; ; \; \mathrm{id} \; := E \\
\mathrm{id} \; := \mathrm{num} \; ; \; \mathrm{id} \; := \underline{E} \\
\mathrm{id} \; := \mathrm{num} \; ; \; \mathrm{id} \; := E + \underline{E} \\
\mathrm{id} \; := \mathrm{num} \; ; \; \mathrm{id} \; := \underline{E} + (S \; , \; E \;) \\
\mathrm{id} \; := \mathrm{num} \; ; \; \mathrm{id} \; := \mathrm{id} + (\underline{S} \; , \; E \;) \\
\mathrm{id} \; := \mathrm{num} \; ; \; \mathrm{id} \; := \mathrm{id} + (\mathrm{id} \; := \underline{E} \; , \; E \;) \\
\mathrm{id} \; := \mathrm{num} \; ; \; \mathrm{id} \; := \mathrm{id} + (\mathrm{id} \; := E + E \; , \; \underline{E} \;) \\
\mathrm{id} \; := \mathrm{num} \; ; \; \mathrm{id} \; := \mathrm{id} + (\mathrm{id} \; := \underline{E} + E \; , \; \mathrm{id} \;) \\
\mathrm{id} \; := \mathrm{num} \; ; \; \mathrm{id} \; := \mathrm{id} + (\mathrm{id} \; := \mathrm{num} + \underline{E} \; , \; \mathrm{id} \;) \\
\mathrm{id} \; := \mathrm{num} \; ; \; \mathrm{id} \; := \mathrm{id} + (\mathrm{id} \; := \mathrm{num} + \mathrm{num} \; , \; \mathrm{id} \;)
\end{array}
$$

Uma derivação possível

Mais à esquerda ou mais à direira? Nenhuma das 2

Derivações

Gramática:

1 $S \rightarrow S\,;\,S$	4 $E \rightarrow \text{id}$	8 $L \rightarrow E$
2 $S \rightarrow \text{id} := E$	5 $E \rightarrow \text{num}$	9 $L \rightarrow L\,,\,E$
3 $S \rightarrow \text{print}\,(\,L\,)$	6 $E \rightarrow E + E$	
	7 $E \rightarrow (\,S\,,\,E\,)$	

Cadeia: `id := num; id := id + (id := num + num, id)`

$$
\begin{array}{l}
\underline{S} \\
S\,;\,\underline{S} \\
\underline{S}\,;\ \text{id} := E \\
\text{id} := \underline{E}\,;\ \text{id} := E \\
\text{id} := \text{num}\,;\ \text{id} := \underline{E} \\
\text{id} := \text{num}\,;\ \text{id} := E + \underline{E} \\
\text{id} := \text{num}\,;\ \text{id} := \underline{E} + (S\,,\,E) \\
\text{id} := \text{num}\,;\ \text{id} := \text{id} + (\underline{S}\,,\,E) \\
\text{id} := \text{num}\,;\ \text{id} := \text{id} + (\text{id} := \underline{E}\,,\,E) \\
\text{id} := \text{num}\,;\ \text{id} := \text{id} + (\text{id} := E + E\,,\,\underline{E}) \\
\text{id} := \text{num}\,;\ \text{id} := \text{id} + (\text{id} := \underline{E} + E\,,\,\text{id}) \\
\text{id} := \text{num}\,;\ \text{id} := \text{id} + (\text{id} := \text{num} + \underline{E}\,,\,\text{id}) \\
\text{id} := \text{num}\,;\ \text{id} := \text{id} + (\text{id} := \text{num} + \text{num}\,,\,\text{id})
\end{array}
$$

$$
\begin{array}{l}
\underline{S} \\
\underline{S}\,;\,S \\
\text{id} := \underline{E}\,;\ S \\
\text{id} := \text{num}\,;\ \underline{S} \\
\text{id} := \text{num}\,;\ \text{id} := \underline{E} \\
\text{id} := \text{num}\,;\ \text{id} := \underline{E} + E \\
\quad\vdots
\end{array}
$$

Uma derivação possível

Mais à esquerda ou mais à direira? Nenhuma das 2

Derivações

Gramática:

1 $S \rightarrow S\ ;\ S$	4 $E \rightarrow \mathrm{id}$	8 $L \rightarrow E$
2 $S \rightarrow \mathrm{id} := E$	5 $E \rightarrow \mathrm{num}$	9 $L \rightarrow L\ ,\ E$
3 $S \rightarrow \mathrm{print}\ (\ L\ )$	6 $E \rightarrow E + E$	
	7 $E \rightarrow (S\ ,\ E\ )$	

Cadeia: `id := num; id := id + (id := num + num, id)`

$$
\begin{array}{l}
\underline{S} \\
S\ ;\ \underline{S} \\
\underline{S}\ ;\ \mathrm{id} := E \\
\mathrm{id} := \underline{E}\ ;\ \mathrm{id} := E \\
\mathrm{id} := \mathrm{num}\ ;\ \mathrm{id} := \underline{E} \\
\mathrm{id} := \mathrm{num}\ ;\ \mathrm{id} := E + \underline{E} \\
\mathrm{id} := \mathrm{num}\ ;\ \mathrm{id} := \underline{E} + (S\ ,\ E\ ) \\
\mathrm{id} := \mathrm{num}\ ;\ \mathrm{id} := \mathrm{id} + (\underline{S}\ ,\ E\ ) \\
\mathrm{id} := \mathrm{num}\ ;\ \mathrm{id} := \mathrm{id} + (\mathrm{id} := \underline{E}\ ,\ E\ ) \\
\mathrm{id} := \mathrm{num}\ ;\ \mathrm{id} := \mathrm{id} + (\mathrm{id} := E + E\ ,\ \underline{E}\ ) \\
\mathrm{id} := \mathrm{num}\ ;\ \mathrm{id} := \mathrm{id} + (\mathrm{id} := \underline{E} + E\ ,\ \mathrm{id}\ ) \\
\mathrm{id} := \mathrm{num}\ ;\ \mathrm{id} := \mathrm{id} + (\mathrm{id} := \mathrm{num} + \underline{E}\ ,\ \mathrm{id}\ ) \\
\mathrm{id} := \mathrm{num}\ ;\ \mathrm{id} := \mathrm{id} + (\mathrm{id} := \mathrm{num} + \mathrm{num}\ ,\ \mathrm{id}\ )
\end{array}
$$

Uma derivação possível

Mais à esquerda ou mais à direira? Nenhuma das 2

$$
\begin{array}{l}
\underline{S} \\
\underline{S}\ ;\ S \\
\mathrm{id} := \underline{E}\ ;\ S \\
\mathrm{id} := \mathrm{num}\ ;\ \underline{S} \\
\mathrm{id} := \mathrm{num}\ ;\ \mathrm{id} := \underline{E} \\
\mathrm{id} := \mathrm{num}\ ;\ \mathrm{id} := \underline{E} + E \\
\quad \vdots
\end{array}
$$

Derivação mais à esquerda

Análise sintática

Um algoritmo de análise sintática (ou analisador sintático) é tal que, dada uma cadeia e uma gramática, encontra uma derivação (ou alternativamente uma árvore sintática) para a cadeia segundo aquela gramática

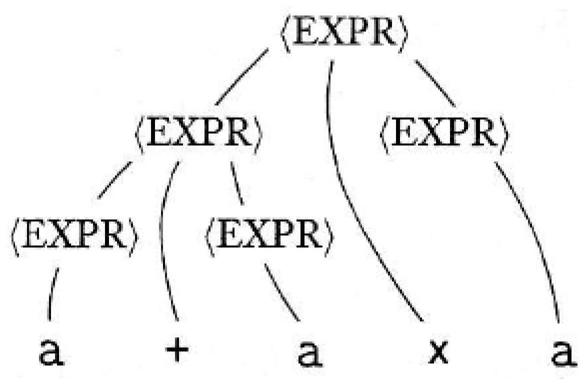
Obs: passo necessário para a compilação de um programa em uma dada linguagem de programação

Mais exemplos de árvores sintáticas

$$\langle \text{EXPR} \rangle \rightarrow \langle \text{EXPR} \rangle \texttt{+} \langle \text{EXPR} \rangle \mid \langle \text{EXPR} \rangle \texttt{x} \langle \text{EXPR} \rangle \mid \texttt{(} \langle \text{EXPR} \rangle \texttt{)} \mid \texttt{a}$$

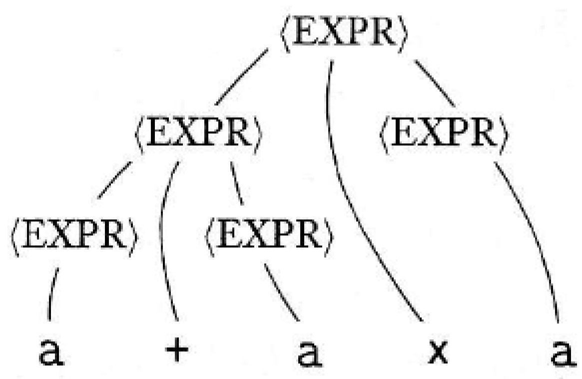
Mais exemplos de árvores sintáticas

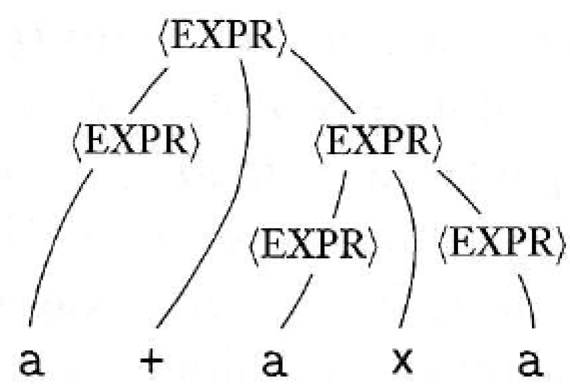
$$\langle \mathrm{EXPR} \rangle \rightarrow \langle \mathrm{EXPR} \rangle \texttt{+} \langle \mathrm{EXPR} \rangle \mid \langle \mathrm{EXPR} \rangle \texttt{x} \langle \mathrm{EXPR} \rangle \mid \texttt{(} \langle \mathrm{EXPR} \rangle \texttt{)} \mid \texttt{a}$$



Mais exemplos de árvores sintáticas

$$\langle \text{EXPR} \rangle \rightarrow \langle \text{EXPR} \rangle \texttt{+} \langle \text{EXPR} \rangle \mid \langle \text{EXPR} \rangle \texttt{x} \langle \text{EXPR} \rangle \mid \texttt{(} \langle \text{EXPR} \rangle \texttt{)} \mid \texttt{a}$$





Mais exemplos de árvores sintáticas

$$\langle \text{EXPR} \rangle \rightarrow \langle \text{EXPR} \rangle + \langle \text{EXPR} \rangle \mid \langle \text{EXPR} \rangle \times \langle \text{EXPR} \rangle \mid (\langle \text{EXPR} \rangle) \mid \mathtt{a}$$

Duas árvores sintáticas distintas para a mesma cadeia!!!!

Ambiguidade

DEFINIÇÃO 2.7

Uma cadeia w é derivada ***ambiguamente*** na gramática livre-do-contexto G se ela tem duas ou mais derivações mais à esquerda diferentes. A gramática G é ***ambígua*** se ela gera alguma cadeia ambiguamente.

Ambiguidade

DEFINIÇÃO 2.7

Uma cadeia w é derivada ***ambiguamente*** na gramática livre-do-contexto G se ela tem duas ou mais derivações mais à esquerda diferentes. A gramática G é ***ambígua*** se ela gera alguma cadeia ambiguamente.

- Ambiguidade é às vezes indesejável, por exemplo em linguagens de progamação
- Algumas gramáticas ambíguas podem ser convertidas em não-ambíguas
- Algumas linguagens são inerentemente ambíguas (só podem ser descritas por gramáticas ambíguas)
  - Eu vi o menino com uma luneta

Expressões aritméticas sem ambiguidade

EXEMPLO 2.4

Considere a gramática $G_4 = (V, \Sigma, R, \langle \text{EXPR} \rangle)$.

V é $\{\langle \text{EXPR} \rangle, \langle \text{TERM} \rangle, \langle \text{FACTOR} \rangle\}$ e Σ é $\{\texttt{a}, +, \times, (,)\}$. As regras são

$$\begin{aligned}
\langle \text{EXPR} \rangle &\to \langle \text{EXPR} \rangle + \langle \text{TERM} \rangle \mid \langle \text{TERM} \rangle \\
\langle \text{TERM} \rangle &\to \langle \text{TERM} \rangle \times \langle \text{FACTOR} \rangle \mid \langle \text{FACTOR} \rangle \\
\langle \text{FACTOR} \rangle &\to (\langle \text{EXPR} \rangle) \mid \texttt{a}
\end{aligned}$$

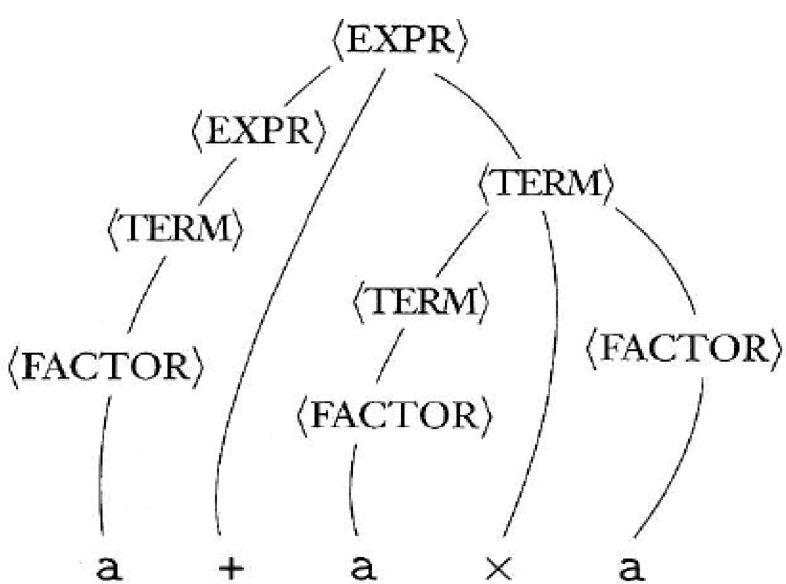
Expressões aritméticas sem ambiguidade

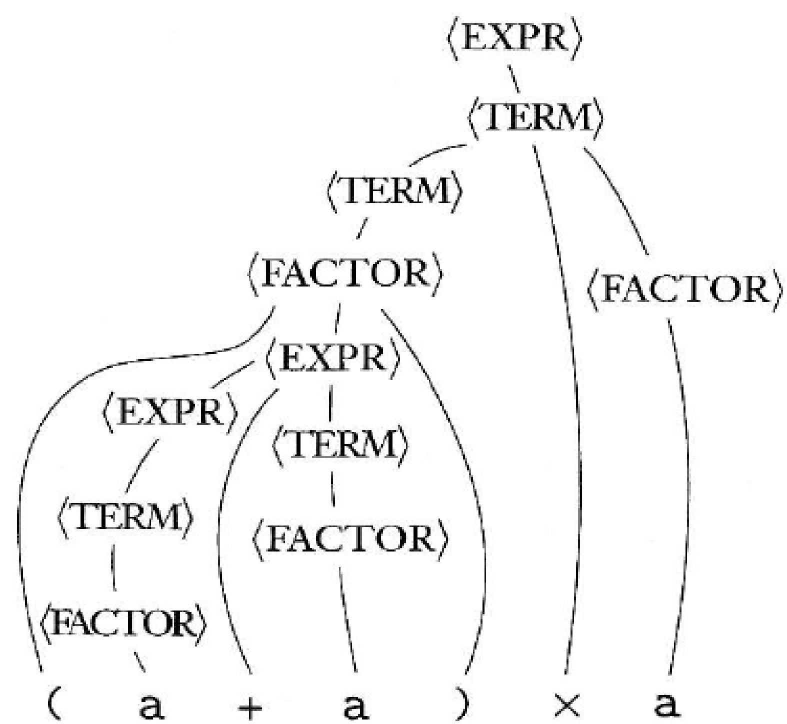
EXEMPLO 2.4

Considere a gramática $G_4 = (V, \Sigma, R, \langle \text{EXPR} \rangle)$.

V é $\{\langle \text{EXPR} \rangle, \langle \text{TERM} \rangle, \langle \text{FACTOR} \rangle\}$ e Σ é $\{\mathtt{a}, +, \mathtt{x}, (,)\}$. As regras são

$$\begin{aligned}
\langle \text{EXPR} \rangle &\to \langle \text{EXPR} \rangle + \langle \text{TERM} \rangle \mid \langle \text{TERM} \rangle \\
\langle \text{TERM} \rangle &\to \langle \text{TERM} \rangle \mathtt{x} \langle \text{FACTOR} \rangle \mid \langle \text{FACTOR} \rangle \\
\langle \text{FACTOR} \rangle &\to (\langle \text{EXPR} \rangle) \mid \mathtt{a}
\end{aligned}$$





O caso if then else

<prog> → … <com>...

<com> → ...

<com> → <cond>

<cond> → if <exp> then <com>

<cond> → if <exp> then <com> else <com>

<exp> → …

if <exp> then if <com> then <com> else <com>

O caso if then else

<prog> → … <com>...

<com> → ...

<com> → <cond>

<cond> → if <exp> then <com>

<cond> → if <exp> then <com> else <com>

<exp> → …

if <exp> then if <com> then <com> else <com>

Com que o *else* faz "par"?

O caso if then else

<prog> → … <com>...

<com> → ...

<com> → <cond>

<cond> → if <exp> then <com>

<cond> → if <exp> then <com> else <com>

<exp> → …

if <exp> then if <com> then <com> else <com>

AMBIGUIDADE!!!

O caso if then else

Solução 1: “manter a ambiguidade” sintática, e resolvê-la por meio de uma convenção: o *else* deve “fazer par” com o último *if* (solução adotada por muitas linguagens de programação)

Solução 2: resolver a ambiguidade sintaticamente, tornando a gramática não ambígua

O caso if then eles – Solução 2

<prog> → … <com>...

<com> → ...

<com> → <cond>

<cond> → if <exp> then <com> endif

<cond> → if <exp> then <com> else <com> endif

<exp> → …

if <exp> then if <com> then <com> endif else <com> endif

ou

if <exp> then if <com> then <com> else <com> endif endif

SEM AMBIGUIDADE!!!

Resolução de ambiguidade

- Opção 1: tirar a ambiguidade da gramática
  – Como feito nas expressões aritméticas
  – Como feito no caso if/then/else com inclusão da palavra-chave endif
- Opção 2: Convencionar a forma de desambiguar, e “programar o analisador sintático” para seguir esse convenção
  – Como feito na maioria das linguagens de programação para tratar o caso if/then/else (que casa o else com o if imediatamente anterior)
  – Obs: isso não tira a ambiguidade da gramática (simplesmente o analisador sintático se satisfaz com uma árvore ao invés de calcular todas)

Forma Normal de Chomsky

DEFINIÇÃO 2.8

Uma gramática livre-do-contexto está na ***forma normal de Chomsky*** se toda regra é da forma

$$A \to BC$$
$$A \to a$$

onde a é qualquer terminal e A, B e C são quaisquer variáveis — exceto que B e C não podem ser a variável inicial. Adicionalmente, permitimos a regra $S \to \varepsilon$, onde S é a variável inicial.

Forma Normal de Chomsky

DEFINIÇÃO 2.8

Uma gramática livre-do-contexto está na ***forma normal de Chomsky*** se toda regra é da forma

$$A \to BC$$
$$A \to a$$

onde a é qualquer terminal e A, B e C são quaisquer variáveis — exceto que B e C não podem ser a variável inicial. Adicionalmente, permitimos a regra $S \to \varepsilon$, onde S é a variável inicial.

TEOREMA 2.9

Qualquer linguagem livre-do-contexto é gerada por uma gramática livre-do-contexto na forma normal de Chomsky.

Forma Normal de Chomsky

DEFINIÇÃO 2.8

Uma gramática livre-do-contexto está na ***forma normal de Chomsky*** se toda regra é da forma

$$A \to BC$$
$$A \to a$$

onde a é qualquer terminal e A, B e C são quaisquer variáveis — exceto que B e C não podem ser a variável inicial. Adicionalmente, permitimos a regra $S \to \varepsilon$, onde S é a variável inicial.

TEOREMA 2.9

Qualquer linguagem livre-do-contexto é gerada por uma gramática livre-do-contexto na forma normal de Chomsky.

Por que isso é importante?

Porque possibilita um algoritmo eficiente de análise sintática (CYK – que será visto adiante). Embora existam outros analisadores também eficientes que não possuem essa exigência (ex: alg. de Earley)

Prova

- (S não pode aparecer do lado direito de nenhuma regra)

Prova

- Novo símbolo inicial S_0 e nova regra $S_0 \rightarrow S$

Prova

- Novo símbolo inicial S_0 e nova regra $S_0 \to S$
-
    - Somente S pode ter uma produção da forma $S \to \varepsilon$
    -
    -
-
    -
    -
-
    -
    -

Prova

- Novo símbolo inicial S_0 e nova regra $S_0 \rightarrow S$
- Para toda regra $A \rightarrow \varepsilon$, $A \neq S$,
  - Remove a regra $A \rightarrow \varepsilon$
  - Se existe $R \rightarrow \alpha A \beta$, , $\alpha, \beta \in (V \cup \Sigma)^*$
  - Se existe $R \rightarrow A$,

Prova

- Novo símbolo inicial S_0 e nova regra $S_0 \rightarrow S$
- Para toda regra $A \rightarrow \varepsilon$, $A \neq S$,
  - Remove a regra $A \rightarrow \varepsilon$
  - Se existe $R \rightarrow \alpha A \beta$, acrescenta $R \rightarrow \alpha\beta$, $\alpha, \beta \in (V \cup \Sigma)^*$
  - Se existe $R \rightarrow A$, acrescenta $R \rightarrow \varepsilon$
-
  -
  -
-
  -
  -

Prova

- Novo símbolo inicial S_0 e nova regra $S_0 \rightarrow S$
- Para toda regra $A \rightarrow \varepsilon$, $A \neq S$,
  - Remove a regra $A \rightarrow \varepsilon$
  - Se existe $R \rightarrow \alpha A \beta$, acrescenta $R \rightarrow \alpha\beta$, $\alpha, \beta \in (V \cup \Sigma)^*$
  - Se existe $R \rightarrow A$, acrescenta $R \rightarrow \varepsilon$
-
  - Produções da forma
  - $A \rightarrow BC$ ou
- $A \rightarrow a$
  -
  -

Prova

- Novo símbolo inicial S_0 e nova regra $S_0 \rightarrow S$
- Para toda regra $A \rightarrow \varepsilon$, $A \neq S$,
  - Remove a regra $A \rightarrow \varepsilon$
  - Se existe $R \rightarrow \alpha A \beta$, acrescenta $R \rightarrow \alpha\beta$, $\alpha, \beta \in (V \cup \Sigma)^*$
  - Se existe $R \rightarrow A$, acrescenta $R \rightarrow \varepsilon$
- Para toda regra $A \rightarrow B$
  - Remove a regra $A \rightarrow B$
  - Se existe $B \rightarrow \alpha$, acrescenta , $\alpha \in (V \cup \Sigma)^*$
-
  -
  -

Prova

- Novo símbolo inicial S_0 e nova regra $S_0 \rightarrow S$
- Para toda regra $A \rightarrow \varepsilon$, $A \neq S$,
  - Remove a regra $A \rightarrow \varepsilon$
  - Se existe $R \rightarrow \alpha A \beta$, acrescenta $R \rightarrow \alpha\beta$, $\alpha, \beta \in (V \cup \Sigma)^*$
  - Se existe $R \rightarrow A$, acrescenta $R \rightarrow \varepsilon$
- Para toda regra $A \rightarrow B$
  - Remove a regra $A \rightarrow B$
  - Se existe $B \rightarrow \alpha$, acrescenta $A \rightarrow \alpha$, $\alpha \in (V \cup \Sigma)^*$
-
  -
  -

Prova

- Novo símbolo inicial S_0 e nova regra $S_0 \rightarrow S$
- Para toda regra $A \rightarrow \varepsilon$, $A \neq S$,
  - Remove a regra $A \rightarrow \varepsilon$
  - Se existe $R \rightarrow \alpha A \beta$, acrescenta $R \rightarrow \alpha\beta$, $\alpha, \beta \in (V \cup \Sigma)^*$
  - Se existe $R \rightarrow A$, acrescenta $R \rightarrow \varepsilon$
- Para toda regra $A \rightarrow B$
  - Remove a regra $A \rightarrow B$
  - Se existe $B \rightarrow \alpha$, acrescenta $A \rightarrow \alpha$, $\alpha \in (V \cup \Sigma)^*$
- Substitui cada regra $A \rightarrow u_1u_2..u_k$, $u_i \in (V \cup \Sigma)$ por
  - $A \rightarrow u_1A_1$, $A_1 \rightarrow u_2A_2$, … $A_{k-2} \rightarrow u_{k-1}u_k$, se $k >= 3$
  - $A \rightarrow U_1U_2$, $U_1 \rightarrow u_1$, $U_2 \rightarrow u_2$ se $k = 2$ e $u_i \in \Sigma$

Exemplo

1. A GLC original G_6 é mostrada à esquerda. O resultado de se aplicar o primeiro passo para introduzir uma nova variável inicial aparece à direita.

$$\begin{aligned} S &\to ASA \mid \mathtt{a}B \\ A &\to B \mid S \\ B &\to \mathtt{b} \mid \varepsilon \end{aligned}$$

$$\begin{aligned} \boldsymbol{S_0} &\to \boldsymbol{S} \\ S &\to ASA \mid \mathtt{a}B \\ A &\to B \mid S \\ B &\to \mathtt{b} \mid \varepsilon \end{aligned}$$

Exemplo

2. Remova as regras ε $B \rightarrow \varepsilon$, mostrado à esquerda, e $A \rightarrow \varepsilon$, mostrado à direita.

$$
\begin{aligned}
S_0 &\rightarrow S \\
S &\rightarrow ASA \mid \mathtt{a}B \mid \mathbf{a} \\
A &\rightarrow B \mid S \mid \boldsymbol{\varepsilon} \\
B &\rightarrow \mathtt{b} \mid \varepsilon
\end{aligned}
\qquad\qquad
\begin{aligned}
S_0 &\rightarrow S \\
S &\rightarrow ASA \mid \mathtt{a}B \mid \mathtt{a} \mid \boldsymbol{SA} \mid \boldsymbol{AS} \mid \boldsymbol{S} \\
A &\rightarrow B \mid S \mid \varepsilon \\
B &\rightarrow \mathtt{b}
\end{aligned}
$$

Exemplo

3a. Remova regras unitárias $S \rightarrow S$, mostrado à esquerda, e $S_0 \rightarrow S$, mostrado à direita.

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

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

Exemplo

3b. Remova as regras unitárias $A \to B$ e $A \to S$.

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

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

Exemplo

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

4. Converta as regras remanescentes para a forma apropriada acrescentando variáveis e regras adicionais. A gramática final em forma normal de Chomsky, a seguir, é equivalente a G_6. (Na realidade, o procedimento dado no Teorema 2.9 produz diversas variáveis U_i juntamente com várias regras $U_i \to \mathtt{a}$. Simplificamos a gramática resultante usando uma única variável U e a regra $U \to \mathtt{a}$.)

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