



# Compiladores

## Gramáticas livres de contexto

Artur Pereira <artur@ua.pt>,  
Miguel Oliveira e Silva <mos@ua.pt>

DETI, Universidade de Aveiro

Ano letivo de 2022-2023

## Sumário

- ① Gramáticas livres de contexto (GLC)
- ② Derivação e árvore de derivação
- ③ Ambiguidade
- ④ Projeto de gramáticas
- ⑤ Operações sobre GLC
- ⑥ Limpeza de gramáticas

# Gramáticas

## Definição

Uma gramática é um quádruplo  $G = (T, N, P, S)$ , onde

- $T$  é um conjunto finito não vazio de **símbolos terminais**;
- $N$ , com  $N \cap T = \emptyset$ , é um conjunto finito não vazio de **símbolos não terminais**;
- $P$  é um conjunto de **produções** (ou regras de rescrita), cada uma da forma  $\alpha \rightarrow \beta$ ;
- $S \in N$  é o **símbolo inicial**.

- $\alpha$  e  $\beta$  são designados por **cabeça da produção** e **corpo da produção**, respetivamente.
- No caso geral  $\alpha \in (N \cup T)^* \times N \times (N \cup T)^*$  e  $\beta \in (N \cup T)^*$ .
- Em ANTLR:
  - os terminais são representados por ids começados por letra maiúscula
  - os não terminais são representados por ids começados por letra minúscula

# Gramáticas livres de contexto – GLC

## Definição

$\mathcal{D}$  Uma gramática  $G = (T, N, P, S)$  diz-se **livre de contexto** (ou **independente do contexto**) se, para qualquer produção  $(\alpha \rightarrow \beta) \in P$ , as duas condições seguintes são satisfeitas

$$\begin{aligned}\alpha &\in N \\ \beta &\in (T \cup N)^*\end{aligned}$$

- A linguagem gerada por uma gramática livre de contexto diz-se livre de contexto
- **As gramáticas regulares são livres de contexto**
- As gramáticas livres de contexto são fechadas sob as operações de reunião, concatenação e fecho
  - **mas não o são** sob as operações de intersecção e complementação.

- Note que: se  $\beta \in T^* \cup T^* N$ , então  $\beta \in (T \cup N)^*$

## Derivação

### Exemplo

Q Considere, sobre o alfabeto  $T = \{a, b, c\}$ , a gramática

$$S \rightarrow \varepsilon \mid a B \mid b A \mid c S$$

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

$$B \rightarrow a B B \mid b S \mid c B$$

e transforme o símbolo inicial  $S$  na palavra  $aabcbcb$  por aplicação sucessiva de produções da gramática

R

$$\begin{aligned} S &\Rightarrow aB \Rightarrow aaBB \Rightarrow aabSB \Rightarrow aabcSB \Rightarrow aabcB \Rightarrow aabcbS \\ &\Rightarrow aabcbcbS \Rightarrow aabcbcb \end{aligned}$$

- Acabou de se obter uma **derivação à esquerda** da palavra  $aabcbcb$
- Cada passo dessa derivação é uma **derivação direta à esquerda**

- Quando há dois ou mais símbolos não terminais, opta-se por **expandir primeiro o mais à esquerda**

## Derivação

### Definições

D Dada uma palavra  $\alpha A \beta$ , com  $A \in N$  e  $\alpha, \beta \in (N \cup T)^*$ , e uma produção  $(A \rightarrow \gamma) \in P$ , com  $\gamma \in (N \cup T)^*$ , chama-se **derivação direta** à rescrita de  $\alpha A \beta$  em  $\alpha \gamma \beta$ , denotando-se

$$\alpha A \beta \Rightarrow \alpha \gamma \beta$$

D Dada uma palavra  $\alpha A \beta$ , com  $A \in N$ ,  $\alpha \in T^*$  e  $\beta \in (N \cup T)^*$ , e uma produção  $(A \rightarrow \gamma) \in P$ , com  $\gamma \in (N \cup T)^*$ , chama-se **derivação direta à esquerda** à rescrita de  $\alpha A \beta$  em  $\alpha \gamma \beta$ , denotando-se

$$\alpha A \beta \xRightarrow{E} \alpha \gamma \beta$$

D Dada uma palavra  $\alpha A \beta$ , com  $A \in N$ ,  $\alpha \in (N \cup T)^*$  e  $\beta \in T^*$ , e uma produção  $(A \rightarrow \gamma) \in P$ , com  $\gamma \in (N \cup T)^*$ , chama-se **derivação direta à direita** à rescrita de  $\alpha A \beta$  em  $\alpha \gamma \beta$ , denotando-se

$$\alpha A \beta \xRightarrow{D} \alpha \gamma \beta$$

## Derivação

### Definições

$\mathcal{D}$  Chama-se **derivação** a uma sucessão de zero ou mais derivações diretas, denotando-se

$$\alpha \Rightarrow^* \beta \quad \equiv \quad \alpha = \gamma_0 \Rightarrow \gamma_1 \Rightarrow \dots \Rightarrow \gamma_n = \beta$$

onde  $n$  é o comprimento da derivação.

$\mathcal{D}$  Chama-se **derivação à esquerda** a uma sucessão de zero ou mais derivações diretas à esquerda, denotando-se

$$\alpha \Rightarrow^E * \beta \quad \equiv \quad \alpha = \alpha_0 \xRightarrow{E} \alpha_1 \xRightarrow{E} \dots \xRightarrow{E} \alpha_n = \beta$$

onde  $n$  é o comprimento da derivação.

$\mathcal{D}$  Chama-se **derivação à direita** a uma sucessão de zero ou mais derivações diretas à direita, denotando-se

$$\alpha \xRightarrow{D} * \beta \quad \equiv \quad \alpha = \gamma_0 \xRightarrow{D} \gamma_1 \xRightarrow{D} \dots \xRightarrow{D} \gamma_n = \beta$$

onde  $n$  é o comprimento da derivação.

## Derivação

### Exemplo

$\mathcal{Q}$  Considere, sobre o alfabeto  $T = \{a, b, c\}$ , a gramática seguinte

$$S \rightarrow \varepsilon \mid a B \mid b A \mid c S$$

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

$$B \rightarrow a B B \mid b S \mid c B$$

Determine as derivações à esquerda e à direita da palavra  $aabcbcb$

$\mathcal{R}$

à esquerda

$$\begin{aligned} S &\Rightarrow aB \Rightarrow aaBB \xRightarrow{1^\circ} aabSB \xRightarrow{2^\circ} aabcSB \\ &\Rightarrow aabcB \Rightarrow aabcbS \Rightarrow aabcbcbS \Rightarrow aabcbcb \end{aligned}$$

à direita

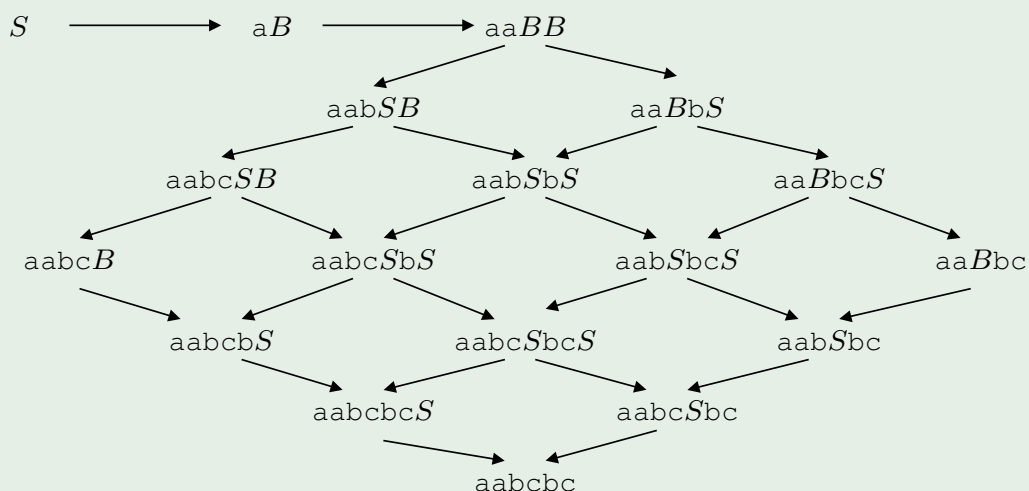
$$\begin{aligned} S &\Rightarrow aB \Rightarrow aaBB \xRightarrow{2^\circ} aaBbS \xRightarrow{1^\circ} aaBbcbS \\ &\Rightarrow aaBbcb \Rightarrow aabSbcb \Rightarrow aabcbSbcb \Rightarrow aabcbcb \end{aligned}$$

• Note que se usou  $\Rightarrow$  em vez de  $\xRightarrow{D}$  e  $\xRightarrow{E}$

## Derivação

### Alternativas de derivação

- O grafo seguinte capta as alternativas de derivação. Considera-se novamente a palavra `aabcbcb` e a gramática anterior



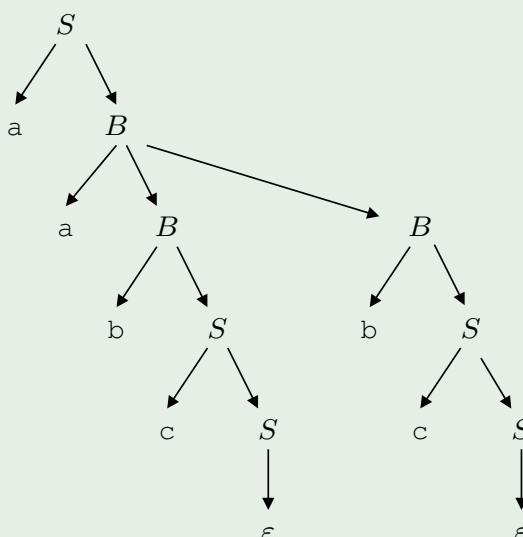
- Identifique os caminhos que correspondem às derivações à direita e à esquerda

## Derivação

### Árvore de derivação

- $\mathcal{D}$  Uma **árvore de derivação** (*parse tree*) é uma representação de uma derivação onde os nós-ramos são símbolos não terminais e os nós-folhas são símbolos terminais

- A árvore de derivação da palavra `aabcbcb` na gramática anterior é

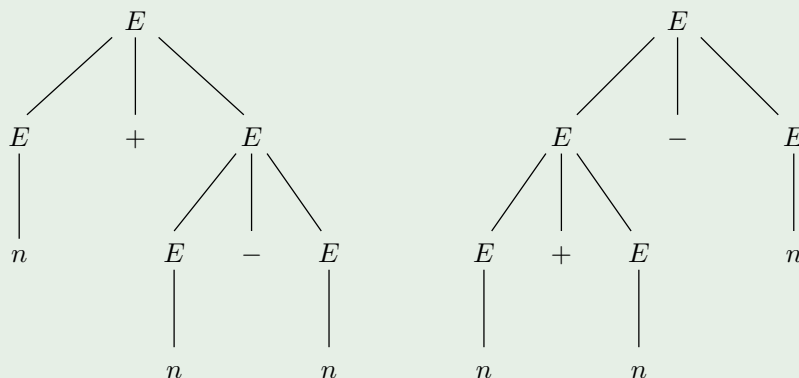


# Ambiguidade

## Ilustração através de um exemplo

- Considere a gramática  $S \rightarrow S + S \mid S - S \mid ( S ) \mid n$  e desenhe a árvore de derivação da palavra  $n+n-n$

$\mathcal{R}$  Podem obter-se duas árvores de derivação diferentes



- Pode haver duas interpretações diferentes para a palavra; há **ambiguidade**

# Ambiguidade

## Definição

$\mathcal{D}$  Diz-se que uma palavra é derivada **ambiguamente** se possuir duas ou mais árvores de derivação distintas

$\mathcal{D}$  Diz-se que uma gramática é **ambígua** se possuir pelo menos uma palavra gerada ambigualmente

- Frequentemente é possível definir-se uma gramática não ambígua que gera a mesma linguagem que uma ambígua
- No entanto, há gramáticas **inerentemente ambíguas**

Por exemplo, a linguagem

$$L = \{a^i b^j c^k \mid i = j \vee j = k\}$$

não possui uma gramática não ambígua que a represente.

# Ambiguidade

## Remoção da ambiguidade

$\mathcal{R}$  Considere-se novamente a gramática

$$S \rightarrow S + S \mid S - S \mid ( S ) \mid n$$

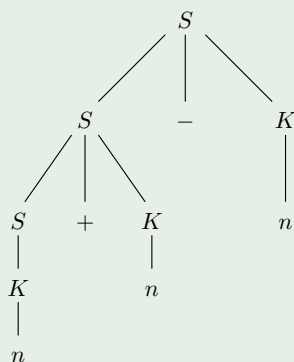
e obtenha-se uma gramática não ambígua equivalente

$\mathcal{R}$

$$S \rightarrow K \mid S + K \mid S - K$$

$$K \rightarrow n \mid ( S )$$

$\mathcal{Q}$  Desenhe a árvore de derivação da palavra  $n+n-n$  na nova gramática



# Projeto de gramáticas

## Exemplo #1, solução #1

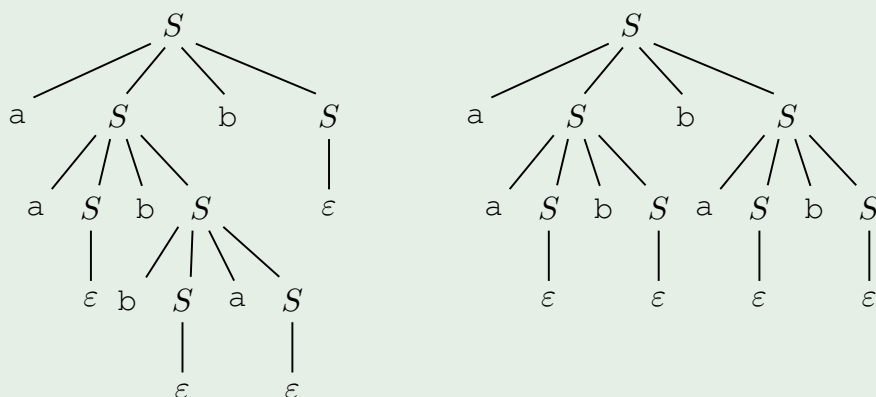
$\mathcal{Q}$  Sobre o conjunto de terminais  $T = \{a, b\}$ , determine uma gramática livre de contexto que represente a linguagem

$$L_1 = \{\omega \in T^* : \#(a, \omega) = \#(b, \omega)\}$$

$\mathcal{R}_1$

$$S \rightarrow \varepsilon \mid a S b S \mid b S a S$$

$\mathcal{Q}$  A gramática é ambígua? Analise a palavra aabbab



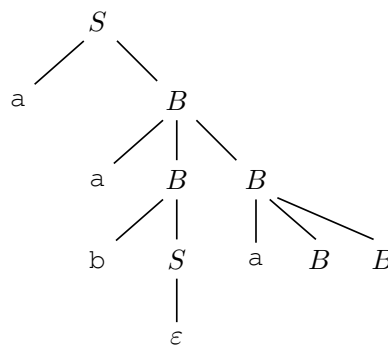
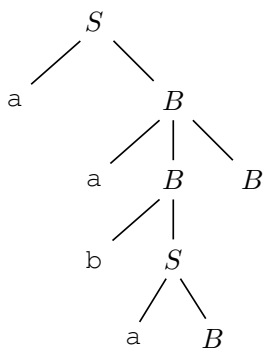
### Exemplo #1, solução #2

Q Sobre o conjunto de terminais  $T = \{a, b\}$ , determine uma gramática livre de contexto que represente a linguagem

$$L_1 = \{\omega \in T^* : \#(a, \omega) = \#(b, \omega)\}$$

 $\mathcal{R}_2$ 
$$S \rightarrow \varepsilon \mid a B \mid b A$$
$$A \rightarrow a \ S \mid b \ A \ A$$
$$B \rightarrow a \ B \ B \mid b \ S$$

Q A gramática é ambígua?  
Analise a palavra aababb.



- Falta expandir alguns nós

# Projeto de gramáticas

### Exemplo #1, solução #3

Q Sobre o conjunto de terminais  $T = \{a, b\}$ , determine uma gramática livre de contexto que represente a linguagem

$$L_1 = \{\omega \in T^* : \#(a, \omega) = \#(b, \omega)\}$$

 $\mathcal{R}_3$ 
$$S \rightarrow \varepsilon \mid a B S \mid b A S$$
$$A \rightarrow a \mid b \ A \ A$$
$$B \rightarrow a \ B \ B \mid b$$

Q A gramática é ambígua? Analise a palavra aababb



## Projeto de gramáticas

### Exemplo #2

Q Sobre o conjunto de terminais  $T = \{a, b, c\}$ , determine uma gramática livre de contexto que represente a linguagem

$$L_2 = \{\omega \in T^* : \#(a, \omega) = \#(b, \omega)\}$$

R

$$S \rightarrow \varepsilon \mid a B S \mid b A S \mid c S$$

$$A \rightarrow a \mid b A A \mid c A$$

$$B \rightarrow a B B \mid b \mid c B$$

Q A gramática é ambígua? *ver aSbbaa*

*Sim é ambígua*

## Projeto de gramáticas

### Exemplo #3, solução #1

Q Sobre o conjunto de terminais  $T = \{a, b, c\}$ , determine uma gramática livre de contexto que represente a linguagem

$$L_3 = \{\omega \in T^* : \#(a, \omega) = \#(b, \omega) \wedge \\ \forall i \leq |\omega| \#(a, \text{prefix}(i, \omega)) \geq \#(b, \text{prefix}(i, \omega))\}$$

R<sub>1</sub>

$$S \rightarrow \varepsilon \mid a S b S \mid c S$$

Q A gramática é ambígua? Analise a palavra aababb

- O número de ocorrências das letras a e b é igual, mas em qualquer prefixo das palavras da linguagem não pode haver mais bs que as, ou seja o a aparece antes
- Solução inspirada na do exemplo 1.1, removendo a produção  $S \rightarrow b S a S$

## Projeto de gramáticas

### Exemplo #3: solução #2

Q Sobre o conjunto de terminais  $T = \{a, b, c\}$ , determine uma gramática livre de contexto que represente a linguagem

$$L_3 = \{\omega \in T^* : \#(a, \omega) = \#(b, \omega) \wedge \forall_{i \leq |\omega|} \#(a, \text{prefix}(i, \omega)) \geq \#(b, \text{prefix}(i, \omega))\}$$

$\mathcal{R}_2$

$$S \rightarrow \varepsilon \mid a B \mid c S$$

$$B \rightarrow a B B \mid b S \mid c B$$

Q A gramática é ambígua? Analise a palavra aababb

- Solução inspirada na do exemplo 1.2, removendo a produção  $S \rightarrow b A$  e as começadas por  $A$

## Projeto de gramáticas

### Exemplo #3: solução #3

Q Sobre o conjunto de terminais  $T = \{a, b, c\}$ , determine uma gramática livre de contexto que represente a linguagem

$$L_3 = \{\omega \in T^* : \#(a, \omega) = \#(b, \omega) \wedge \forall_{i \leq |\omega|} \#(a, \text{prefix}(i, \omega)) \geq \#(b, \text{prefix}(i, \omega))\}$$

$\mathcal{R}_3$

$$S \rightarrow \varepsilon \mid a B S \mid c S$$

$$B \rightarrow a B B \mid b \mid c B$$

Q A gramática é ambígua? Analise a palavra aababb

- Solução inspirada na do exemplo 1.3, removendo a produção  $S \rightarrow b A S$  e as começadas por  $A$

## Projeto de gramáticas

### Exercício

- Q Sobre o conjunto de terminais  $T = \{a, b, c, (, ), +, *\}$ , determine uma gramática independente do contexto que represente a linguagem

$$L = \{ \omega \in T^* : \\ \omega \text{ representa uma expressão regular sobre o alfabeto } \{a, b, c\} \}$$

- R Em ANTLR, poder-se-ia fazer

```
S → E
E → E '*'
   | E E
   | E '+' E
   | '(' E ')'
   | 'a' | 'b' | 'c'
```

mas em geral não, porque, em geral, as alternativas estão todas ao mesmo nível

- Como escrever a gramática de modo à precedência ser imposta por construção?

- 
- Está a usar-se o operador + em vez do |

## Projeto de gramáticas

### Exercício (cont.)

- R Em geral

```
S → E
E → E '+' T
   | T
T → T F
   | F
F → F '*'
   | O
O → '(' E ')'
   | 'a' | 'b' | 'c'
```

- Uma expressão é vista como uma 'soma' de termos
- Um termo é visto como um 'produto' (concatenação) de fatores
- Um fator é visto como um 'fecho' de operandos
- Um operando ou é um elemento base ou uma expressão entre parêntesis

- 
- Está a usar-se o operador + em vez do |

# Reunião de GLC

## Exemplo

- Q Sobre o conjunto de terminais  $T = \{a, b, c\}$ , determine uma gramática livre de contexto que represente a linguagem

$$L = \{ \omega \in T^* : \#(a, \omega) = \#(b, \omega) \vee \#(a, \omega) = \#(c, \omega) \}$$

R

$L_1 = \{ \omega \in T^* : \#(a, \omega) = \#(b, \omega) \}$	$S_1 \rightarrow \varepsilon \mid a S_1 b S_1$ $\mid b S_1 a S_1 \mid c S_1$
$L_2 = \{ \omega \in T^* : \#(a, \omega) = \#(c, \omega) \}$	$S_2 \rightarrow \varepsilon \mid a S_2 c S_2$ $\mid b S_2 \mid c S_2 a S_2$
$L = L_1 \cup L_2$	$S \rightarrow S_1 \mid S_2$ $S_1 \rightarrow \varepsilon \mid a S_1 b S_1$ $\mid b S_1 a S_1 \mid c S_1$ $S_2 \rightarrow \varepsilon \mid a S_2 c S_2$ $\mid b S_2 \mid c S_2 a S_2$

- Para esta linguagem, mesmo que as gramáticas de  $L_1$  e  $L_2$  não sejam ambíguas, a de  $L$  será ambígua. Porquê?

Handwritten notes:  $cabbbcc$  is in  $L_1$  and  $L_2$ , so  $L$  is ambiguous.

## Operações sobre GLCs

### Reunião

- D Sejam  $G_1 = (T_1, N_1, P_1, S_1)$  e  $G_2 = (T_2, N_2, P_2, S_2)$  duas gramáticas livres de contexto quaisquer, com  $N_1 \cap N_2 = \emptyset$ .

A gramática  $G = (T, N, P, S)$  onde

$$T = T_1 \cup T_2$$

$$N = N_1 \cup N_2 \cup \{S\} \text{ com } S \notin (N_1 \cup N_2)$$

$$P = \{S \rightarrow S_1, S \rightarrow S_2\} \cup P_1 \cup P_2$$

é livre de contexto e gera a linguagem  $L = L(G_1) \cup L(G_2)$

- As novas produções  $S \rightarrow S_i$ , com  $i = 1, 2$ , permitem que  $G$  gere a linguagem  $L(G_i)$
- Esta definição é idêntica à que foi dada para a operação de reunião nas gramáticas regulares

# Concatenação de GLC

## Exemplo

Q Sobre o conjunto de terminais  $T = \{a, b, c\}$ , determine uma gramática livre de contexto que represente a linguagem

$$L = \{ \omega_1 \omega_2 : \omega_1, \omega_2 \in T^* \}$$

$$\wedge \#(a, \omega_1) = \#(b, \omega_1) \wedge \#(a, \omega_2) = \#(c, \omega_2) \}$$

R

$L_1 = \{ \omega \in T^* : \#(a, \omega) = \#(b, \omega) \}$	$S_1 \rightarrow \varepsilon \mid a S_1 b S_1$ $\mid b S_1 a S_1 \mid c S_1$
$L_2 = \{ \omega \in T^* : \#(a, \omega) = \#(c, \omega) \}$	$S_2 \rightarrow \varepsilon \mid a S_2 c S_2$ $\mid b S_2 \mid c S_2 a S_2$
$L = L_1 \cdot L_2$	$S \rightarrow S_1 S_2$ $S_1 \rightarrow \varepsilon \mid a S_1 b S_1$ $\mid b S_1 a S_1 \mid c S_1$ $S_2 \rightarrow \varepsilon \mid a S_2 c S_2$ $\mid b S_2 \mid c S_2 a S_2$

# Operações sobre gramáticas:

## Concatenação

D Sejam  $G_1 = (T_1, N_1, P_1, S_1)$  e  $G_2 = (T_2, N_2, P_2, S_2)$  duas gramáticas livres de contexto quaisquer, com  $N_1 \cap N_2 = \emptyset$ .

A gramática  $G = (T, N, P, S)$  onde

$$T = T_1 \cup T_2$$

$$N = N_1 \cup N_2 \cup \{S\} \text{ com } S \notin (N_1 \cup N_2)$$

$$P = \{S \rightarrow S_1 S_2\} \cup P_1 \cup P_2$$

é livre de contexto e gera a linguagem  $L = L(G_1) \cdot L(G_2)$

- A nova produção  $S \rightarrow S_1 S_2$  justapõe palavras de  $L(G_2)$  às de  $L(G_1)$
- Esta definição é **diferente** da que foi dada para a operação de concatenação nas gramáticas regulares

# Fecho de Kleene de GLC

## Exemplo

- Q Sobre o conjunto de terminais  $T = \{a, b, c\}$ , determine uma gramática livre de contexto que represente a linguagem

$$L = \{\omega \in T^* : \#(a, \omega) \geq \#(b, \omega)\}$$

R

$X = \{\omega \in T^* : \#(a, \omega) = \#(b, \omega)\}$	$X \rightarrow \varepsilon \mid a B \mid b A \mid c X$ $A \rightarrow a X \mid b A A \mid c A$ $B \rightarrow a B B \mid b X \mid c B$
$A = \{\omega \in T^* : \#(a, \omega) = \#(b, \omega) + 1\}$	Basta usar o $A$ anterior como símbolo inicial
$L = X \cup A^*$	$S \rightarrow \varepsilon \mid A S \mid X$ $X \rightarrow \varepsilon \mid a B \mid b A \mid c X$ $A \rightarrow a X \mid b A A \mid c A$ $B \rightarrow a B B \mid b X \mid c B$

- O fecho de  $A$  inclui a palavra vazia mas não as outras palavras com  $\#_a = \#_b$

# Operações sobre gramáticas

## Fecho de Kleene

Seja  $G_1 = (T_1, N_1, P_1, S_1)$  uma gramática livre de contexto qualquer. A gramática  $G = (T, N, P, S)$  onde

$$\begin{aligned} T &= T_1 \\ N &= N_1 \cup \{S\} \quad \text{com } S \notin N_1 \\ P &= \{S \rightarrow \varepsilon, S \rightarrow S_1 S\} \cup P_1 \end{aligned}$$

é livre de contexto e gera a linguagem  $L = (L(G_1))^*$

- A produção  $S \rightarrow \varepsilon$ , per si, garante que  $L^0(G_1) \subseteq L(G)$
- As produções  $S \rightarrow S_1 S$  e  $S \rightarrow \varepsilon$  garantem que  $L^i(G_1) \subseteq L(G)$ , para qualquer  $i > 0$
- Esta definição é **diferente** da que foi dada para a operação de fecho nas gramáticas regulares

# Símbolos produtivos e improdutivos

## Exemplo de ilustração

Q Sobre o conjunto de terminais  $T = \{a, b, c, d\}$ , considere a gramática

$$\begin{aligned} S &\rightarrow a A b \mid b B \\ A &\rightarrow c C \mid b B \mid d \\ B &\rightarrow d D \mid b \\ C &\rightarrow A C \mid B D \mid S D \\ D &\rightarrow A D \mid B C \mid C S \\ E &\rightarrow a A \mid b B \mid \varepsilon \end{aligned}$$

- Tente expandir (através de uma derivação) o símbolo não terminal  $A$  para uma sequência apenas com símbolos terminais ( $S \Rightarrow^* u$ , com  $u \in T^*$ )
  - $A \Rightarrow d$
- Faça o mesmo com o símbolo  $C$ 
  - Não consegue
- $A$  é um símbolo **produtivo**;  $C$  é um símbolo **improdutivo**

# Símbolos produtivos e improdutivos

## Definição de símbolo produtivo

- Seja  $G = (T, N, P, S)$  uma gramática qualquer
- Um símbolo não terminal  $A$  diz-se **produtivo** se for possível expandi-lo para uma expressão contendo apenas símbolos terminais
- Ou seja,  $A$  é produtivo se

$$A \Rightarrow^+ u \quad \wedge \quad u \in T^*$$

- Caso contrário, diz-se que  $A$  é **improdutivo**
- Uma gramática é **improdutiva** se o seu símbolo inicial for improdutivo

- Na gramática

$$\begin{aligned} S &\rightarrow a b \mid a S b \mid X \\ X &\rightarrow c X \end{aligned}$$

- $S$  é produtivo, porque  $S \Rightarrow ab \quad \wedge \quad ab \in T^*$
- $X$  é improdutivo, porque  $X \Rightarrow cX \Rightarrow ccX \Rightarrow^* c \cdots cX$

# Símbolos produtivos

## Algoritmo de cálculo

- O conjunto dos símbolos produtivos,  $N_p$ , pode ser obtido por aplicação sucessiva das seguintes regras construtivas

```
if  $(A \rightarrow \alpha) \in P$  and  $\alpha \in T^*$  then  $A \in N_p$   
if  $(A \rightarrow \alpha) \in P$  and  $\alpha \in (T \cup N_p)^*$  then  $A \in N_p$ 
```

- Algoritmo de cálculo:

```
let  $N_p \leftarrow \emptyset$ ,  $P_p \leftarrow P$       #  $N_p$  – símbolos produtivos  
repeat  
  nothingAdded  $\leftarrow$  true  
  foreach  $(A \rightarrow \alpha) \in P_p$  do  
    if  $\alpha \in (T \cup N_p)^*$  then          # se todos são terminais ou produtivos,  $A$  é produtivo  
      if  $A \notin N_p$  then                # se ainda não pertence aos produtivos  
         $N_p \leftarrow N_p \cup \{A\}$     # é lá colocado  
        nothingAdded  $\leftarrow$  false    # e é preciso repetir o processo  
         $P_p \leftarrow P_p - \{A \rightarrow \alpha\}$  # a produção já não precisa de ser processada mais  
until nothingAdded or  $N_p = N$ 
```

# Símbolos acessíveis e inacessíveis

## Exemplo de ilustração

Q Sobre o conjunto de terminais  $T = \{a, b, c, d\}$ , considere a gramática

```
 $S \rightarrow a A b \mid b B$   
 $A \rightarrow c C \mid b B \mid d$   
 $B \rightarrow d D \mid b$   
 $C \rightarrow A C \mid B D \mid S D$   
 $D \rightarrow A D \mid B C \mid C S$   
 $E \rightarrow a A \mid b B \mid \varepsilon$ 
```

- Tente alcançar (através de uma derivação) o símbolo não terminal  $C$  a partir do símbolo inicial ( $S$ ) ( $S \Rightarrow^* \alpha C \beta$ , com  $\alpha, \beta \in (T \cup N)^*$ )
  - $S \Rightarrow b B \Rightarrow b d D \Rightarrow b d B C$
- Faça o mesmo com o símbolo  $E$ 
  - Não consegue
- $C$  é um símbolo **acessível**;  $E$  é um símbolo **inacessível**



# Símbolos acessíveis e inacessíveis

## Definição de símbolo acessível

- Seja  $G = (T, N, P, S)$  uma gramática qualquer
- Um símbolo terminal ou não terminal  $x$  diz-se **acessível** se for possível expandir  $S$  (o símbolo inicial) para uma expressão que contenha  $x$
- Ou seja,  $x$  é acessível se
$$S \Rightarrow^* \alpha x \beta$$
- Caso contrário, diz-se que  $x$  é **inacessível**

- Na gramática

$$S \rightarrow \varepsilon \mid a S b \mid c C c$$
$$C \rightarrow c S c$$
$$D \rightarrow d X d$$
$$X \rightarrow C C$$

- $D, d,$  e  $X$  são inacessíveis
- Os restantes são acessíveis

# Símbolos acessíveis

## Algoritmo de cálculo

- O conjunto dos seus símbolos acessíveis,  $V_A$ , pode ser obtido por aplicação das seguintes regras construtivas

$$S \in V_A$$
$$\text{if } A \rightarrow \alpha B \beta \in P \text{ and } A \in V_A \text{ then } B \in V_A$$

- Algoritmo de cálculo:

$$V_A \leftarrow \{S\}$$

# no fim, ficará com todos os símbolos acessíveis

$$N_A \leftarrow \{S\}$$

# conjunto de símbolos não terminais acessíveis a processar

**repeat**
$$X \leftarrow \text{elementOf}(N_A)$$
# retira um elemento qualquer de  $N_A$ **foreach**  $(X \rightarrow \alpha) \in P$  **do****foreach**  $x$  **in**  $\alpha$  **do****if**  $x \notin V_A$  **then**

# se ainda não está marcado como acessível

$$V_A \leftarrow V_A \cup \{x\}$$

# passa a estar

**if**  $x \in N$  **then**

# se adicionalmente é não terminal

$$N_A \leftarrow N_A \cup \{x\}$$

# terá de ser processado

**until**  $N_A = \emptyset$

# Gramáticas limpas

## Algoritmo de limpeza

- Numa gramática, os símbolos inacessíveis e os símbolos improdutivos são **símbolos inúteis**
- Se tais símbolos forem removidos obtém-se uma gramática equivalente
- Diz-se que uma gramática é **limpa** se não possuir símbolos inúteis
- Para limpar uma gramática deve-se:
  - <sup>eliminar</sup> começar por a expurgar dos símbolos improdutivos
  - só depois remover os inacessíveis

# Gramáticas limpas

## Exemplo #1

Q Sobre o conjunto de terminais  $T = \{a, b, c, d\}$ , determine uma gramática limpa equivalente à gramática seguinte

$$\begin{aligned} S &\rightarrow a A b \mid b B \\ A &\rightarrow c C \mid b B \mid d \\ B &\rightarrow d D \mid b \\ C &\rightarrow A C \mid B D \mid S D \\ D &\rightarrow A D \mid B C \mid C S \\ E &\rightarrow a A \mid b B \mid \varepsilon \end{aligned}$$

- Cálculo dos símbolos produtivos

- 1 Inicialmente  $N_p \leftarrow \emptyset$
- 2  $A \rightarrow d \wedge d \in T^* \implies N_p \leftarrow N_p \cup \{A\}$
- 3  $B \rightarrow b \wedge b \in T^* \implies N_p \leftarrow N_p \cup \{B\}$
- 4  $E \rightarrow \varepsilon \wedge \varepsilon \in T^* \implies N_p \leftarrow N_p \cup \{E\}$
- 5  $S \rightarrow a A b \wedge a, A, b \in (T \cup N_p)^* \implies N_p \leftarrow N_p \cup \{S\}$
- 6 Nada mais se consegue acrescentar a  $N_p \implies C$  e  $D$  são improdutivos

# Gramáticas limpas

## Exemplo #1, cont.

- Gramática após a remoção dos símbolos improdutivos

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

$$A \rightarrow b B \mid d$$

$$B \rightarrow b$$

$$E \rightarrow a A \mid b B \mid \varepsilon$$

- Cálculo dos símbolos não terminais acessíveis sobre a nova gramática

1  $S$  é acessível, porque é o inicial

2 sendo  $S$  acessível, de  $S \rightarrow a A b$ , tem-se que  $A$  é acessível

3 sendo  $S$  acessível, de  $S \rightarrow b B$ , tem-se que  $B$  é acessível

4 de  $A$  só se chega a  $B$ , que já foi marcado como acessível

5 de  $B$  não se chega a nenhum não terminal

6 Logo  $E$  não é acessível, pelo que a gramática limpa é

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

$$A \rightarrow b B \mid d$$

$$B \rightarrow b$$