Gramáticas Livres de Contexto - GLCs

Conteúdo

- Gramáticas Livres de Contexto
- Árvores de Derivação
- Ambiguidade
- Forma Normal de Backus
  - Backus-Naur Form (BNF)

Linguagens Livres de Contexto

- Quando uma linguagem é livre de contexto?

- Desafio:
  - Criar uma Expressão Regular para a linguagem:

$$L = \{ (^n)^n \mid n >= 0 \}$$

Gramática Livre de Contexto (GLC)

- Um GLC é definida por uma quadrupla:
- G = (V, T, P, S)
  - V = Conjunto de Variáveis ou Não terminais
  - T = Conjunto de Terminais ou Constantes
  - P = Regras da Produção na forma:

    $A \rightarrow \alpha$, onde: $A \in V$, e $\alpha \in T$
  - S = Símbolo inicial da gramática

Exemplos de GLC

- G = (V , T , P, S)
- G = ({A}, {0,1}, P, A)
- P:

  A → 0A
  A → 1A
  A → 0

Pode-se escrever:

A → 0A | 1A | 0

Linguagem definida pela GLC

- Atingida por um processo de inferência chamado derivação

- Os Não terminais são substituídos por suas regras de produção até formarem palavras, partindo do símbolo inicial

Exemplo de Derivação

=> denota derivação

Seja G = ({A}, {0,1}, {A → 0A | 1A | 0 }, A)

A => 0A => 00A => 001A => 0010

- A palavra 0010 pertence a linguagem definida pela gramática G

Exemplo Passo a Passo

Seja G = ({A}, {0,1}, {A → 0A | 1A | 0}, A)

A => 0A

0010 – pertence a linguagem

Exemplo Passo a Passo

Seja G = ({A}, {0,1}, {A → 0A | 1A | 0}, A)

A => 0A => 00A

0010 – pertence a linguagem

Exemplo Passo a Passo

Seja G = ({A}, {0,1}, {A → 0A | 1A | 0}, A)

A => 0A => 00A => 001A

0010 – pertence a linguagem

Exemplo Passo a Passo

Seja G = ({A}, {0,1}, {A → 0A | 1A | 0}, A)

A => 0A => 00A => 001A => 0010

A derivação termina quando não há mais símbolos Não Terminais

0010 – pertence a linguagem

Outro Exemplo

G = ({S,A},{a,b},P,S)

P : S →abA

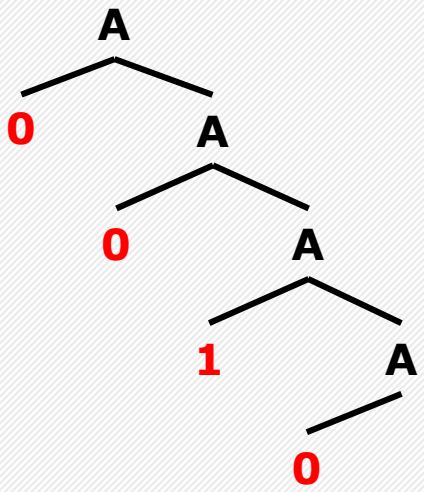
A →aA | bA | ε

- S => abA => abaA => ababA => abab
- S => abA => ab
- S => abA => abaA => abaaA => abaa

Árvore de Derivação

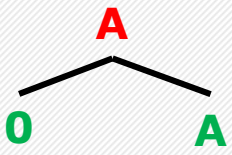
- Também chamada de árvore gramatical
- Ilustra as derivações em forma de árvore
- As folhas da árvore contém a palavra gerada

A => 0A => 00A => 001A => 0010



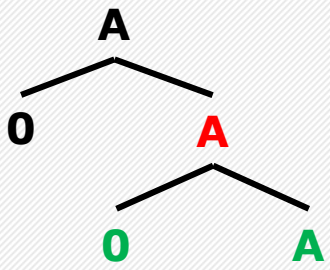
Árvore de Derivação

A => 0A



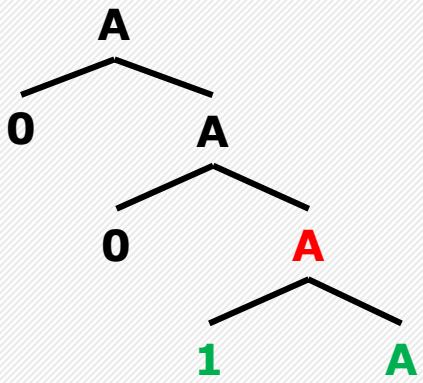
Árvore de Derivação

A => 0A => 00A



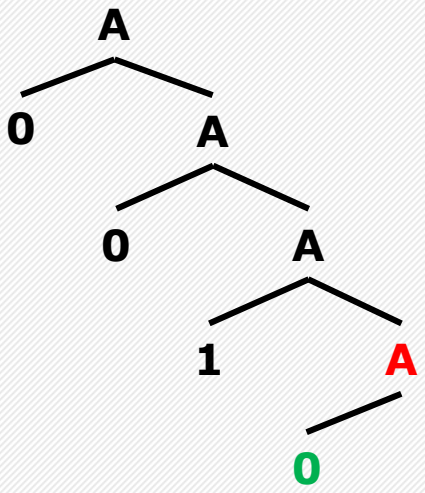
Árvore de Derivação

A => 0A => 00A => 001A



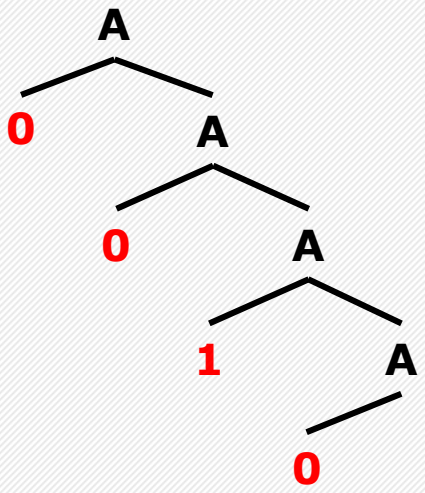
Árvore de Derivação

A => 0A => 00A => 001A => 0010



Árvore de Derivação

A => 0A => 00A => 001A => 0010



Características das Regras de Produção de uma GLC

- Recursão à esquerda
  - S → Sab | ε
- Recursão à direita
  - S → abS | ε
- Recursão indireta
  - S →aAa

    A →a | S
- Ambiguidade
  - Duas ou mais árvores diferentes podem ser geradas para uma mesma palavra

Forma Normal de Backus (BNF)

- Backus – Naur Form
- Backus – Naur Formalism
- Notação frequentemente usada para descrever GLC de linguagens de programação

- Existem variações
  - EBNF = Extended BNF
  - ABNF = Augmented BNF

Forma Normal de Backus (BNF)

- Notação

  <variavel> ::= terminal

  <variavel> ::= <variavel> terminal

  ou

  <naoterminal> ::= terminal | <naoterminal> terminal

- Exemplo

  <programa> ::= inicio <lista_cmd> fim

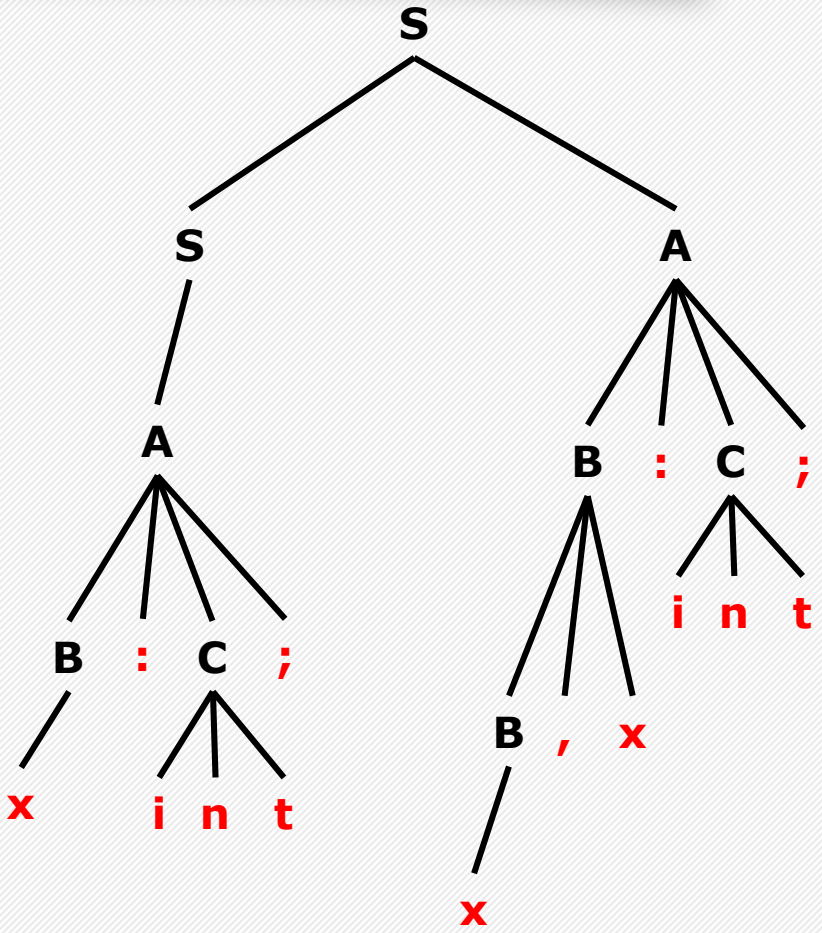
  <lista_cmd> ::= <lista_cmd> <cmd> | <cmd>

  <cmd> ::= x := num

Árvore de Derivação

GLC
S → SA | A
A → B : C ;
B → B , x | x
C → int

Palavra
x : int;
x, x : int;



Gramática Ambígua

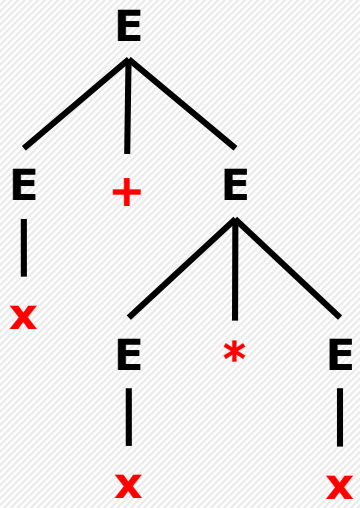
GLC

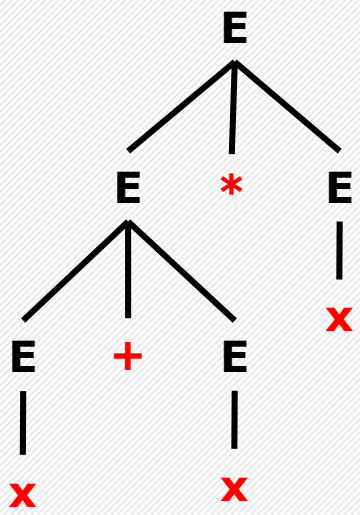
E → E + E
| E * E
| x

Palavra

x + x * x

Uma Gramática é ambígua se existem duas árvores possíveis para uma mesma palavra da linguagem.





Eliminando a Ambiguidade

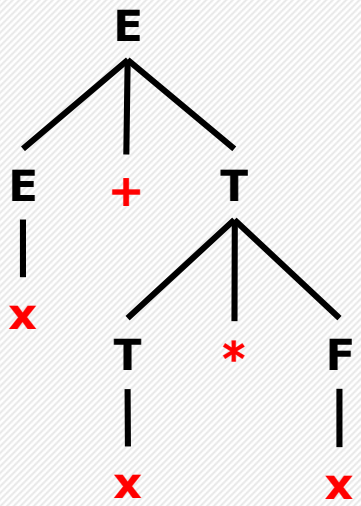
GLC

E → E + T | T
T → T * F | F
F → x

Palavra

x + x * x

Esta é a única árvore possível



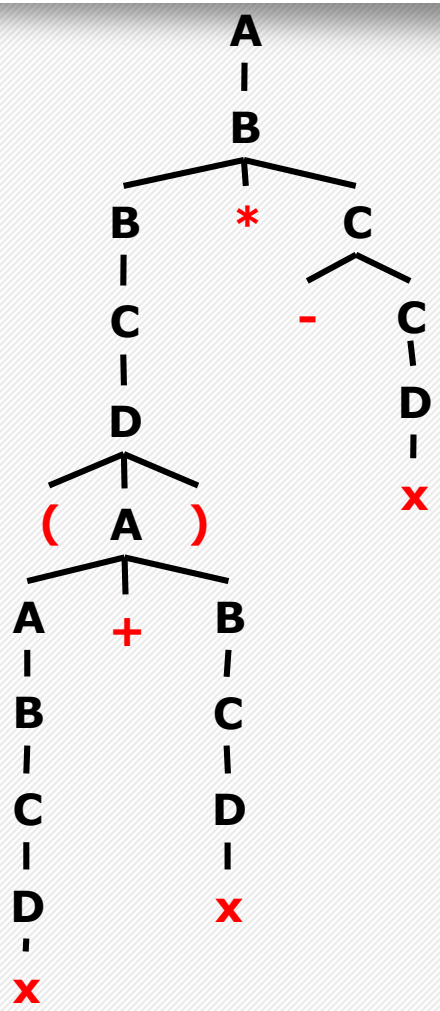
Precedência de Operações

GLC

A → A + B | A – B | B
B → B * C | B / C | C
C → - C | D
D → (A) | x

Palavra

(x + x) * - x



Operações de maior prioridade/ precedência ficam mais abaixo

Exemplos BNF

<programa> ::= inicio <lista_cmd> fim
<lista_cmd> ::= <lista_cmd> <cmd> | <cmd>
<cmd> ::= x := num

Token num : [0-9]+

```
inicio
    x := 33
    x := 66
fim
```

Exemplos BNF

<exp> ::= <exp> + <exp> | <exp> * <exp> | num | id

Tokens:

num : [0-9]+

id : [a-z][a-z0-9]*

10 + 11

20 * 20

a * 10

Exercício de BNF

Like a Turtle

<l_cmd> ::= <l_cmd> <cmd> | <cmd>
<cmd> ::= <dir> num
<dir> ::= pe | pd | pf | pt

Crie algumas sentenças que seriam reconhecidas por esta GLC.

http://www.transum.org/software/Logo/

http://www.calormen.com/jslogo

http://logo.twentygototen.org/

https://turtleacademy.com/learn.php

Exercício de BNF

Like a Turtle

<l_cmd> ::= <l_cmd> <cmd> | <cmd>
<cmd> ::= <dir> num
<dir> ::= pe | pd | pf | pt

Tokens:

pd : “pd”
pt : “pt”
pf : “pf”
pe : “pe”

num : [0-9]+
repita: “repita”
“[“
“]”

: [\n\t\s\r\ ]*

Exercício complementar de BNF

- A linguagem Logo permite repetições na forma:

```
repita 4 [
  pf 100
  pd 90
]
```

Podem ocorrer várias repetições (até aninhadas).
Altere a GLC **Like a Turtle** para permitir isso.