V – Análise Sintática

V.1 – Fundamentos Teóricos

(Capítulos IV e V da apostila de LFC)

V.1.1 – Gramáticas Livres de Contexto

Definições de GLC

1 – G = (Vn, Vt, P, S) onde

$P = \{A \rightarrow \alpha \mid A \in Vn \ \wedge \alpha \in (Vn \cup Vt)^{+}\}$

2 – GLC ε - LIVRES :

S → ε pode pertencer a P, desde que:

S seja o símbolo inicial de G

S não apareça no lado direito das Produções de P

3 – G = (Vn, Vt, P, S) onde

$P = \{A \rightarrow \alpha \mid A \in Vn \ \wedge \alpha \in (Vn \cup Vt)^{*}\}$

Ou seja α pode ser ε !!!

Toda **GLC** pode ser transformada em uma **GLC ε - Livre**

Exemplos:

Árvore de Derivação

- Representação estruturada das derivações de G

Definição:

Seja G = (Vn, Vt, P, S) uma G.L.C. G.

Uma **Árvore** é uma **Árvore de Derivação** de G, se:

a) Todos os nodos forem rotulados por símbolos de Vn ∪ Vt;
b) A raiz da árvore for S, o símbolo inicial de G;
c) Todo nodo com descendentes for um símbolo ∈ Vn;
d) Para todo nodo A com descendentes diretos A_1, A_2, ..., A_k existir em P uma produção da forma: $A \rightarrow A_1\ A_2 \ldots A_k$

Exemplo:

S → a S c | B

B → b B | ε

Limite de uma AD

- Concatenação das folhas da AD ≡ Forma sentencial

Formas de Derivação

- Derivação + a Esquerda
- Derivação + a Direita

Exemplo: S → A B | S c

A → a A | ε

B → b B | ε

Gramáticas Ambíguas

- G é ambígua se

$$\exists\ x \in L(G) \mid x \text{ possua mais de uma AD}$$

- Exemplos

  1- $S \rightarrow S\ b\ S \mid a$

  2- $E \rightarrow E + E \mid E * E \mid (E) \mid id$

  3- A gramática do "if"

Linguagens inerentemente ambíguas

- Linguagens que só possuem representações ambíguas

$$\mathbf{L(G) = \{\ a^n\ b^m\ c^k \mid n = m \vee m = k\ \}}$$

Transformações em GLC

1 - Eliminação de Símbolos Inúteis

- Inalcançáveis e/ou inférteis

$$\mathbf{S \overset{*}{\Rightarrow} w\,X\,y \overset{*}{\Rightarrow} w\,x\,y}$$

2 – Transformação de GLG em GLC ε - Livre

- Eliminação de ε-produções

3 – Eliminação de produções Simples

- produções da forma $A \rightarrow B \mid A \wedge B \in Vn$

4 - Fatoração de GLC

- Uma GLC G é dita FATORADA, se ela NÃO possui A ∈ Vn | A derive seqüências que iniciam com o mesmo símbolo por mais de um caminho (usando derivações distintas)

Processo de Fatoração

- **Não-Fatoração Direta**
  Substituir produções da forma:

  $A \rightarrow \alpha\,\beta \mid \alpha\,\gamma$

  Pelo seguinte conjunto de produções:

  $A \rightarrow \alpha\,A'$

  $A' \rightarrow \beta \mid \gamma$

- **Não-Fatoração Indireta**

  Transformar em Direto via derivações sucessivas

Exemplos:

1) S → a S | a B | d S
B → b B | b

2) S → AB | BC
A → a A | ε
B → b B | d
C → c C | c

3) S → a S | A
A → a A c | ε

5 - Eliminação de Recursão à Esquerda

- **Não Terminal Recursivo**

$A \overset{+}{\Rightarrow} \alpha A \beta$, para $\alpha \wedge \beta \in V^*$.

se $\alpha \overset{*}{\Rightarrow} \varepsilon$ então A é Recursivo à Esquerda

- **G é Rec. à Esquerda se possui NT Rec. à Esquerda**

Processo de eliminação

- **R.E. Direta**

Substituir produções da forma:

$A \rightarrow A\alpha 1 \mid A\alpha 2 \mid \ldots \mid A\alpha n \mid \beta 1 \mid \beta 2 \mid \ldots \mid \beta m$

Por produções da forma:

$A \rightarrow \beta 1A' \mid \beta 2A' \mid \ldots \mid \beta mA'$

$A' \rightarrow \alpha 1A' \mid \alpha 2A' \mid \ldots \mid \alpha nA' \mid \varepsilon$

Exemplos:

1) $S \rightarrow S\ a \mid b$

2) $E \rightarrow E + T \mid T$
   $T \rightarrow T * F \mid F$
   $F \rightarrow (E) \mid id$

- **R.E. Indireta**

Transformar em Recursão à esquerda Direta, via derivações sucessivas.

Exemplo: $S \rightarrow Aa \mid S\ b$
$A \rightarrow S\ c \mid d$

FIRST

Definição → Conjunto de terminais que podem iniciar uma seqüência de símbolos

Exemplos:

- **Se $\alpha = a\,\beta$ ∴ first (α) = {a}**
- **Se $\alpha = \varepsilon$ ∴ first (α) = ε**
- **Se $\alpha = B\,\beta$ ∴ first (α) = ?? ?**

Algoritmo:

(* para todo X ∈ Vn ∪ Vt *)

1 – Se X ∈ Vt → first (X) = {X}

2 – Se X ∈ Vn ∧ X → aα ∈ P ⇒ a ∈ First (X)

Obs: Se X → ε ∈ P ⇒ ε ∈ First (X)

3 – Se X → y1 y2 ... yk ∈ P ⇒ First (y1) ∈ First (X)

- **Se ε ∈ First (y1) ⇒ First (y2) também ∈ first(X)**
- **Se ε ∈ First (y2) ⇒ ...**
- **Se ε ∈ First (yk) ⇒ ε também ∈ First(X)!!!**

Exercícios:

1) S → Ab | ABc
B → bB | Ad | ε
A → aA | ε

2) S → ABC
A → aA | ε
B → bB | ACd
C → cC | ε

Follow

Definição: Seguidores validos de um símbolo!

$\therefore$ Se $S \Rightarrow^{+} \alpha\ A\ a\ B \rightarrow a \in FOLLOW(A)$

Se $S \Rightarrow^{+} \alpha\ A\ B\ c\ \gamma \rightarrow First(Bc\gamma) \in Follow(A)$

Algoritmo:

(* Para todo $A \in Vn$ *)

1 – Se A é o símbolo inicial da gramática

→ **$ ∈ Follow(A)**

2 – Se $A \rightarrow \alpha B \beta \in P \wedge \beta \neq \varepsilon$

→ **adicione first(β) em Follow(B)**

3 – Se $A \rightarrow \alpha B$ (ou $A \rightarrow \alpha B \beta$, onde $\varepsilon \in First(\beta)) \in P$

→ **adicione Follow(A) em Follow(B)**

Exemplos:

1) S → ABC
A → aA | ε
B → bB | ACd
C → cC | ε

2) E → T E'
E' → + T E' | ε
T → F T'
T' → * F T' | ε
F → (E) | id

3) S → AbCD | EF
A → aA | ε
C → ECF | c
D → CD | dDd | ε
E → eE | ε
F → FS | fF | g

4) S → AC | CeB | Ba
A → aA | BC
C → cC | ε
B → bB | AB | ε

Notações (+ usuais) de GLC

- BNF – Backus-Naur Form

  Exemplos: **1) <S> :: = a <S> | ε**
  2) <E> :: = <E> + id | id

- BNF Estendida (notação de Wirth)

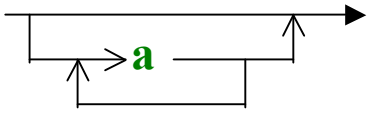
  Exemplos: **1) <S> :: = {a}**
  2) <E> :: = id {+ id}

- RRP (ER estendidas com NT)

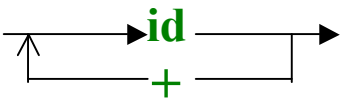
  Exemplos: **1) <S> :: = a***
  2) <E> :: = id $^{£}$ +

- Diagramas Sintáticos (Conway)

  Exemplos: 1) <S> :



2) <E> :



Principais Aplicações de GLC

1 – Especificação de linguagens de programação;

2 – Formalização de parsing / implementação de parser's;

3 – Esquemas de tradução dirigidos pela sintaxe

4 – Processamento de string's, de modo geral.

V.1.2 – Teoria de Parsing

(capítulo V da apostila)

Termos Básicos:

- Parser → Analisador Sintático
- Parsing → Analise Sintática
- Parse → Representação da analise efetuada
  - **Ascendentes: $S \Rightarrow^{+} x$ (* Seq. Invertida de derivações mais a direita ≡ Reducao *)**
  - **Descendentes: $S \Rightarrow^{+} x$ (* Seq. Normal de derivações mais a esquerda ≡ derivação *)**

Exemplo:

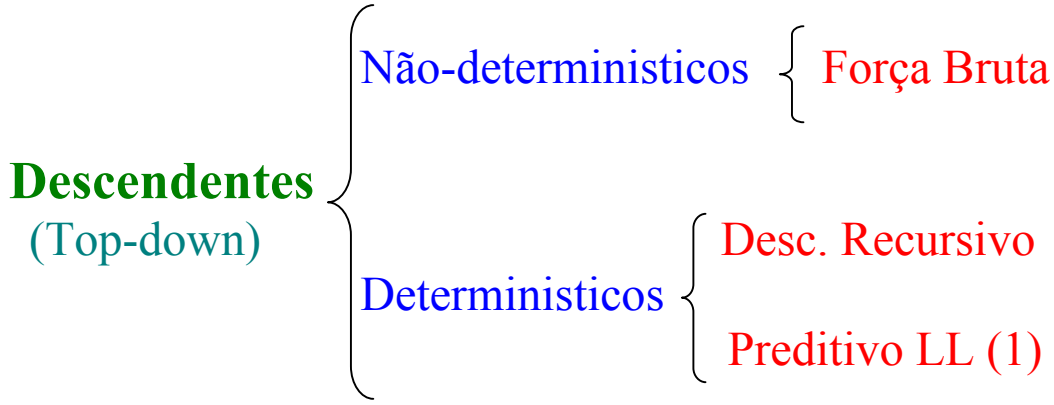
1: S → AbC
2, 3: A → aA | a
4, 5: C → cC | c

x = abc PAx =
PDx =

Classes de Analisadores

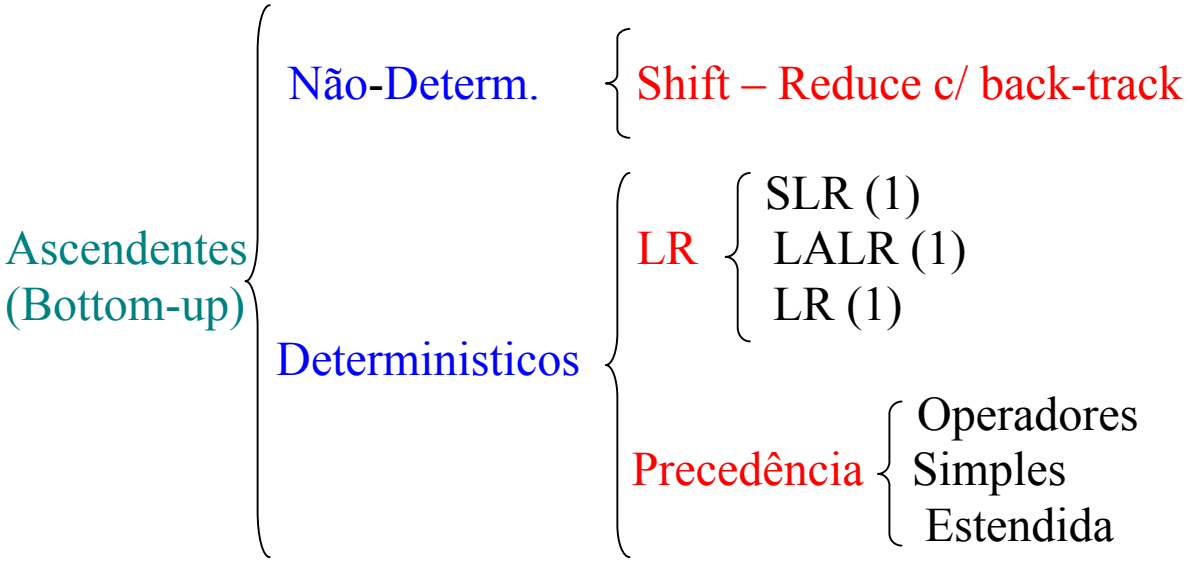
- **Descendentes (Top-down)**
  - **Símbolo inicial → Sentença**
  - **Uso de derivação**

- **Principais técnicas**



- **Ascendentes (Bottom-up)**
  - **Sentença → Símbolo inicial**
  - **Uso de redução**

- **Principais técnicas**



- **Técnicas Não-Determinísticas**
  - exigem implementação com back-track
  - não limitam a classe de GLC que pode ser analisadas
  - complexidade exponencial

    Exemplos:

    Asc. – Alg. geral SHIFT-REDUCE

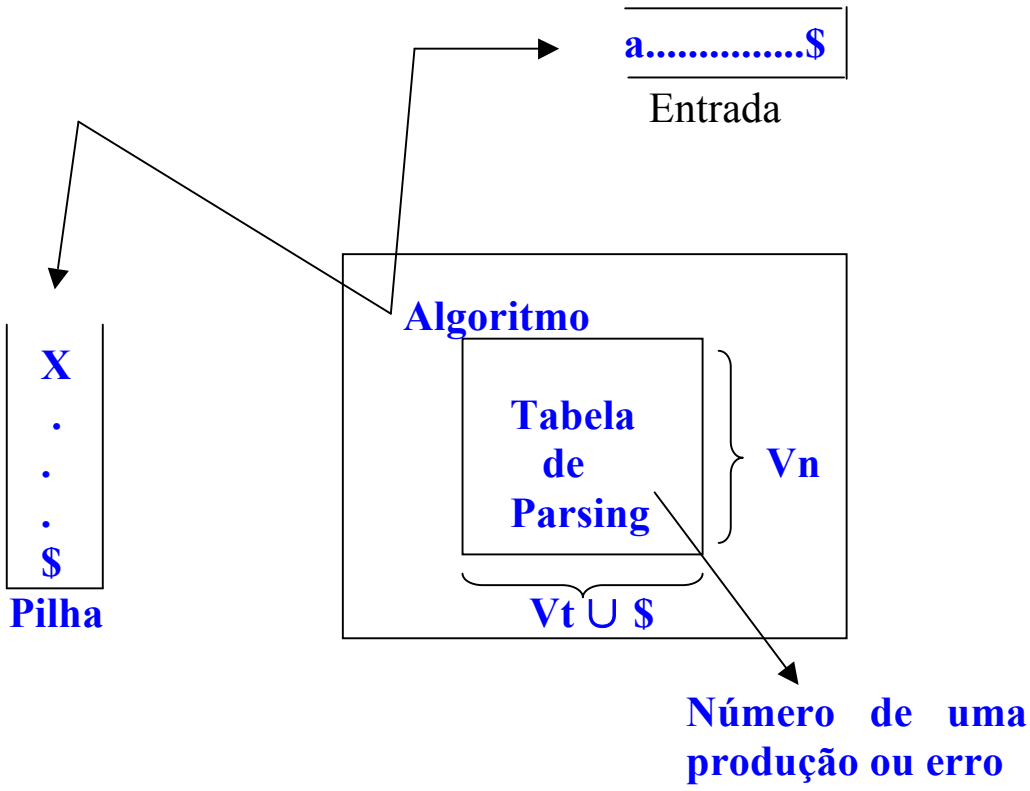
    Desc. – Algoritmo da Força Bruta

- **Técnicas Determinísticas**
  - Implementação sem back-track (determ.)
  - Limitam a classe de GLC que pode ser analisada
  - Algoritmos eficientes - complexidade linear (espaço requerido proporcional ao tamanho da gramática e tempo de análise proporcional ao tamanho da sentença)
  - Parser's automatizáveis
  - Principais Técnicas:

    Descendentes (Top-Down)
    - Desecendente Recursiva
    - Preditiva (LL(1))

    Ascendente (Bottom-Up)
    - Precedência
      - Simples, de Operadores, Estendida
    - LR
      - LR(1),LALR(1),SLR(1)

- **Descendentes (Top-down)**
  - **Força-Bruta (Não-deterministica–c/ back-track)**
  - **Descendente recursivo**
    - **Procedures mutuamente recursivas**
      - **(uma para cada não-terminal)**
    - **Vantagens X Desvantagens**
    - **Exemplo: S → aBc**
      B → bB | d | ε
  - **Preditivo (LL(1))**
    - **Estrutura Geral**



- **Algoritmo Base**

Se x ∈ Vt

Se x = a = $ → fim
Se x = a ≠ $ → reconhece a
Se x ≠ a → erro sintático

Se x ∈ Vn

Se TP(x,a) = Nº de produção → deriva!
Se TP(x,a) = erro → erro sintático!

Exemplo:

G: 1: E → TE'
2,3: E' → +TE'|ε
4: T → FT'
5,6: T' → * FT'|ε
7,8: F → (E) | id

TP:	id	(	)	+	*	$
E	1	1	-	-	-	-
E'	-	-	3	2	-	3
T	4	4	-	-	-	-
T'	-	-	6	6	5	6
F	8	7	-	-	-	-

- **Exercício - Analisar as seguintes sentenças:**
  - **x = id + id $**
  - **Y = id * (id id) $**

Construção da tabela de parsing LL(1)

- **Condição LL(1)**

  1) **Não possuir recursão à esquerda**

  2) **Estar fatorada**

  3) **Para todo $A \in Vn \mid A \overset{*}{\Rightarrow} \varepsilon$, $First(A) \cap Follow(A) = \varphi$**

$\therefore$ Somente GLC que satisfazem estas condições podem ser analisadas (deterministicamente) pelos Analisadores Descendentes LL(1).

Algoritmo para construção da T.P. LL(1)

1) **Para cada produção**

$$A \rightarrow \alpha \in P$$

Execute os passos 2 e 3:

2) **Para todo a $\in$ First(α), exceto ε,**
coloque o número da produção $A \rightarrow \alpha$ em TP(A,a)

3) **Se $\varepsilon \in$ First (α)**
coloque o número da produção $A \rightarrow \alpha$
em TP(A,b), para todo b $\in$ Follow(A)

4) **Coloque "erro" nas posições da TP que ficaram indefinidas.**

Exercícios

1 - C → if E then C else C
| if E then C
| com
E → exp

2 - S → bCDE | Cef
C → cC | ε
E → eSf | ε
D → dDA | aDd | b
A → cAa | aa

3 - P → begin D C end
D → int id I
I → , id I | ε
C → C; T = E
| T = E
| com
E → E + T | T
T → id | id [E]