

Revisão de GLC e Analisadores Descendentes

Marcelo Johann

Conteúdo da aula

1. Exemplos de Gramáticas
2. Propriedades: Ambíguas, sem ciclos, ϵ -livres, fatoradas à esquerda, recursivas à esquerda, simplificadas
3. Transformações: Eliminação de produções vazias, de recursividade à esquerda (direta, indireta), fatoração
4. Analisadores Descendentes
 - Recursivos com Retrocesso
 - Recursivos Preditivos
 - Conjunto FIRST e Implementação

INF01033 - Compiladores B - Marcelo Johann - 2012/1

Aula 07 : Slide 2

Exemplos

Gramática para Comandos

```
cmd → if expr then cmd else cmd | print
expr → ( bool ) | expr && expr | expr || expr
bool → true | false
```

INF01033 - Compiladores B - Marcelo Johann - 2012/1

Aula 07 : Slide 3

Exemplos

Gramática para Expressões

```
E → E OP E | "( E )" | x
OP → "*" | "/" | "+" | "-"
```

INF01033 - Compiladores B - Marcelo Johann - 2012/1

Aula 07 : Slide 4

Exemplos

Gramática para Listas

```
S → a | "[ L ]"
L → S "," L | S
```

INF01033 - Compiladores B - Marcelo Johann - 2012/1

Aula 07 : Slide 5

Ressaltando

- **Convenções**
 - Símbolos que representam terminais em minúsculos:
 - u, v, x, y, \dots
 - Símbolos que representam não-terminais em maiúsculos:
 - $X, Y, \text{TERM}, S, \dots$
 - Símbolos que representam formas sentenciais (seqüências de terminais e não-terminais): letras gregas ou sublinhadas:
 - $\alpha, \beta, \omega, \underline{w}, \underline{z}$
- **Outras...**

INF01033 - Compiladores B - Marcelo Johann - 2012/1

Aula 07 : Slide 6

Derivações

- Derivação em um passo

$$A \Rightarrow xy$$

- Derivação em múltiplos passos

Se $A \Rightarrow w_2 \Rightarrow w_3 \dots \Rightarrow w_n$ dizemos que

$$A \Rightarrow^* w_n$$

Tipos e Características

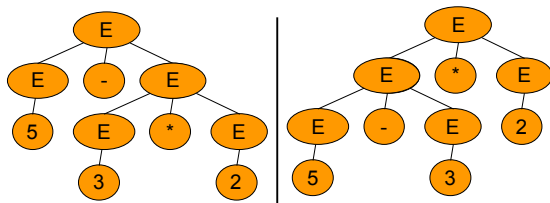
- Gramáticas Ambíguas
- Gramáticas sem ciclos
- Gramáticas ϵ -livres
- Gramáticas fatoradas à esquerda
- Gramáticas recursivas à esquerda
- Gramáticas simplificadas

Gramática Ambígua

$$E \rightarrow E \text{ OP } E \mid \text{"(" E "}" \mid x$$

$$\text{OP} \rightarrow \text{"*"} \mid \text{" / " } \mid \text{" + " } \mid \text{" - " }$$

- Considere $5 - 3 * 2$



- Gramática sem ciclos:**

– Uma gramática sem ciclos é uma GLC que não possui derivações da forma

$$A \Rightarrow^+ A \text{ para algum } A \in N$$

- Gramática ϵ -livre :**

– GLC que não possui produções vazias do tipo

$$A \rightarrow \epsilon$$

exceto a produção $S \rightarrow \epsilon$ (S é o símbolo inicial).

- Gramática fatorada à esquerda:**

– GLC que **não** possui produções do tipo $A \rightarrow \alpha\beta_1 \mid \alpha\beta_2$ para alguma forma sentencial α .

- Gramática recursiva à esquerda:**

– GLC que permite a derivação

$$A \Rightarrow^+ A\alpha \text{ para algum } A \in N$$

O não terminal A deriva ele mesmo, de forma **direta** ou **indireta**, como símbolo mais à esquerda de uma subpalavra gerada.

Transformações de GLCs

- (A) Eliminação de produções vazias
- (B) Eliminação de recursividade à esquerda:
 - recursão direta
 - recursão indireta
- (C) Fatoração de uma gramática

Eliminação de Produções Vazias

- **Objetivo:**
 - eliminar produções da forma $A \rightarrow \epsilon$.
- **Algoritmo:** seja $G = (N, T, P, S)$ uma GLC
 - **Etapa 1:**
 - construir N_ϵ , o conjunto de não-terminais que geram a palavra vazia:
$$N_\epsilon = \{A \mid A \rightarrow \epsilon\}; \text{ // é um conjunto de símbolos}$$

Repita

$$N_\epsilon = N_\epsilon \cup \{X \mid X \rightarrow X_1 \dots X_n \in P \text{ tq } X_1, \dots, X_n \in N_\epsilon\}$$

Até que o cardinal de N_ϵ não aumente.

INF01033 - Compiladores B - Marcelo Johann - 2012/1

Aula 07 : Slide 13

Eliminação de Produções Vazias

- **Etapa 2:**
 - construir o conjunto de produções sem produções vazias:
- gera $G_1 = (N, T, P_1, S)$, onde P_1 é construído como segue:
- $$P_1 = \{A \rightarrow \alpha \mid \alpha \neq \epsilon\};$$
- Repita
- Para toda produção $A \rightarrow \alpha \in P_1$ e $X \in N_\epsilon$ tal que
- $$\alpha = \alpha_1 X \alpha_2 \text{ e } \alpha_1, \alpha_2 \neq \epsilon$$
- Faça $P_1 = P_1 \cup \{A \rightarrow \alpha_1 \alpha_2\}$
- Até que o cardinal de P_1 não aumente

INF01033 - Compiladores B - Marcelo Johann - 2012/1

Aula 07 : Slide 14

Eliminação de Produções Vazias

- **Etapa 3:**
 - incluir a geração da palavra vazia, se necessário:
 - Se a palavra vazia pertence à linguagem, então a gramática resultante é
- $$G_2 = (N, T, P_2, S), \text{ onde } P_2 = P_1 \cup \{S \rightarrow \epsilon\}$$

INF01033 - Compiladores B - Marcelo Johann - 2012/1

Aula 07 : Slide 15

(B) Eliminação de recursividade à esquerda

- Exemplo de GLC recursiva à esquerda:
- $$A \rightarrow Aa \mid b$$
- Gramáticas transformadas equivalentes:

Com a palavra vazia

$$A \rightarrow bX$$

$$X \rightarrow aX \mid \epsilon$$

Sem a palavra vazia

$$A \rightarrow b \mid bX$$

$$X \rightarrow a \mid aX$$

Obs: pode ainda haver recursão indireta!

INF01033 - Compiladores B - Marcelo Johann - 2012/1

Aula 07 : Slide 16

(B) Outro exemplo de recursividade

- $E \rightarrow E+T \mid T$
 - $T \rightarrow T^*F \mid F$
 - $F \rightarrow (E) \mid Id$
- A regra $E \rightarrow E+T \mid T$ se torna:
- $$E \rightarrow TE'$$
- $$E' \rightarrow +TE' \mid \epsilon$$
- A regra $T \rightarrow T^*F \mid F$ se torna:
- $$T \rightarrow FT'$$
- $$T' \rightarrow ^*FT' \mid \epsilon$$

INF01033 - Compiladores B - Marcelo Johann - 2012/1

Aula 07 : Slide 17

(C) Fatoração de uma gramática

- Elimina a indecisão de qual produção aplicar quando duas ou mais produções iniciam com a mesma forma sentencial

$$A \rightarrow \alpha\beta_1 \mid \alpha\beta_2$$

Se torna:

$$A \rightarrow \alpha X$$

$$X \rightarrow \beta_1 \mid \beta_2$$

INF01033 - Compiladores B - Marcelo Johann - 2012/1

Aula 07 : Slide 18

(C) Exemplo de Fatoração a Esquerda

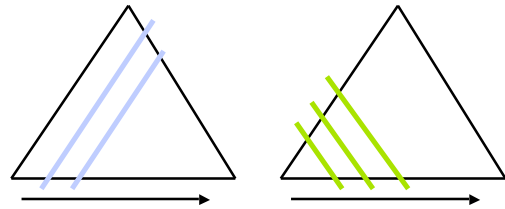
$\text{Cmd} \rightarrow \text{if Expr then Cmd else Cmd}$
 $\text{Cmd} \rightarrow \text{if Expr then Cmd}$
 $\text{Cmd} \rightarrow \text{Outro}$

- Fatorando a esquerda:
 $\text{Cmd} \rightarrow \text{if Expr then Cmd ElseOpc}$
 $\text{Cmd} \rightarrow \text{Outro}$
 $\text{ElseOpc} \rightarrow \text{else Cmd} \mid \epsilon$

INF01033 - Compiladores B - Marcelo Johann - 2012/1

Aula 07 : Slide 19

Análise top-down e bottom-up



INF01033 - Compiladores B - Marcelo Johann - 2012/1

Aula 07 : Slide 20

Analísadores Descendentes

- Também chamados recursivos ou top-down
- Recursivos com Retrocesso
 - Testam cada possível derivação
 - Exemplo
- Recursivos Preditivos
 - Determinam produção pelo símbolo terminal atual
 - Exemplo
- Conjunto FIRST e método de encontrá-lo
- Implementação

INF01033 - Compiladores B - Marcelo Johann - 2012/1

Aula 07 : Slide 21

Exemplo

Gramática para Listas

$S \rightarrow a \mid "[L]"$
 $L \rightarrow S \text{ " , " } L \mid S$

Entrada: $[a] \$$

INF01033 - Compiladores B - Marcelo Johann - 2012/1

Aula 07 : Slide 22

Top-Down: Backtracking

$S \rightarrow AB$		S	cbca	tentar $S \rightarrow AB$
$A \rightarrow c \mid \epsilon$		AB	cbca	tentar $A \rightarrow c$
$B \rightarrow cbB \mid ca$		cB	cbca	casar c
		B	bca	sem-saída, tentar $A \rightarrow \epsilon$
		ϵB	cbca	tentar $B \rightarrow cbB$
		cbB	cbca	casar c
		bB	bca	casar b
		B	ca	tentar $B \rightarrow cbB$
		cbB	ca	casar c
		bB	a	sem-saída, tentar $B \rightarrow ca$
		ca	ca	casar c
		a	a	casar a -> Fim!

• Gera $S \Rightarrow^* cbca?$

INF01033 - Compiladores B - Marcelo Johann - 2012/1

Aula 07 : Slide 23

Observações sobre o método recursivo com retrocesso

- É fácil de implementar.
- É necessário:
 - Que a gramática não seja recursiva à esquerda
 - $A \rightarrow Aa$ se tornará
 - $\text{ReconheceA}() \{ \text{ReconheceA}(); \dots \}$
 - Recursão infinita!

INF01033 - Compiladores B - Marcelo Johann - 2012/1

Aula 07 : Slide 24

Analísadores Descendentes

- Também chamados recursivos ou top-down
- Recursivos com Retrocesso
Testam cada possível derivação
Exemplo
- Recursivos Preditivos
Determinam produção pelo símbolo terminal atual
Exemplo
- Conjunto FIRST e método de encontrá-lo
- Implementação

INF01033 - Compiladores B - Marcelo Johann - 2012/1

Aula 07 : Slide 25

Exemplo

Gramática onde é fácil distinguir produções

```
CMD → "if" EXPR "then" CMD
CMD → "while" EXPR "do" CMD
CMD → "repeat" LISTA "until" EXPR
CMD → ID ":@" EXPR
```

INF01033 - Compiladores B - Marcelo Johann - 2012/1

Aula 07 : Slide 26

Exemplo

Mesma Gramática com mais Não-Terminais

```
CMD → COND | ITER | ATRIB
COND → "if" EXPR "then" CMD
ITER → "while" EXPR "do" CMD |
      "repeat" LISTA "until" EXPR
ATRIB → ID ":@" EXPR
```

INF01033 - Compiladores B - Marcelo Johann - 2012/1

Aula 07 : Slide 27

Exemplo

Outra Gramática

```
CMD: IF | ASS | WHILE | BL;
BL: '{' CMDL '}';
CMDL: CMD RESTO;
RESTO: ';' CMDL | epsilon;
IF: "if" E "then" CMD;
WHILE: "while" E "do" CMD;
ASS: L '=' E;
L: '*' E | E;
E: NUM | ID;
```

INF01033 - Compiladores B - Marcelo Johann - 2012/1

Aula 07 : Slide 28

Definição: Conjuntos "First"

- Seja α qualquer sequência de símbolos
 - terminais ou não terminais
- **First(α)**:
 - Definição informal:
 - conjunto de todos os terminais que começam qualquer sequência derivável de α .
 - Definição formal:
 - Se existe um $t \in T$ e um $\beta \in V^*$ tal que $\alpha \Rightarrow^* t\beta$ então $t \in \text{First}(\alpha)$
 - Se $\alpha \Rightarrow^* \epsilon$ então $\epsilon \in \text{First}(\alpha)$

$A \rightarrow B C D$	$\text{First}(A) = \{b, c, d\}$
$B \rightarrow b$	$\text{First}(B) = \{b\}$
$C \rightarrow c$	$\text{First}(C) = \{c\}$
$D \rightarrow d$	$\text{First}(D) = \{d\}$

INF01033 - Compiladores B - Marcelo Johann - 2012/1

Aula 07 : Slide 29

Condição para que se possa usar um analisador preditivo

- **Informalmente**: no caso que em os First() dos lados direitos das regras de produção sejam "simpáticos", não terá retrocesso.
- **Formalmente**: para qualquer produção
 $A \rightarrow \alpha_1 | \alpha_2 | \dots | \alpha_n$,
 quer-se:
 $\text{First}(\alpha_1) \cap \text{First}(\alpha_2) \cap \dots \cap \text{First}(\alpha_n) = \emptyset$

INF01033 - Compiladores B - Marcelo Johann - 2012/1

Aula 07 : Slide 30

proc First(α : string of symbols)

```
Repeat {
  Para todas as produções  $\alpha \rightarrow X_1 X_2 X_3 \dots X_n$  do
    if  $X_1 \in T$  then // caso simples onde  $X_1$  é um terminal
      First( $\alpha$ ) := First( $\alpha$ )  $\cup$  { $X_1$ }
    else {           // caso menos simples:  $X_1$  é um não-terminal
      First( $\alpha$ ) = First( $\alpha$ )  $\cup$  First( $X_1$ )  $\setminus$  { $\epsilon$ };
      for (i=1 ; i<=n ; i++) {
        if  $\epsilon$  is in First( $X_i$ ) and in First( $X_2$ ) and in... First( $X_{i-1}$ )
          First( $\alpha$ ) := First( $\alpha$ )  $\cup$  First( $X_i$ )  $\setminus$  { $\epsilon$ }
      }
    }
  if ( $\alpha \Rightarrow^* \epsilon$ )
    then First( $\alpha$ ) := First( $\alpha$ )  $\cup$  { $\epsilon$ }
end do
} until no change in any First( $\alpha$ )
```

INF01033 - Compiladores B - Marcelo Johann - 2012/1

Aula 07 : Slide 31

Observações sobre o método recursivo preditivo

- É fácil de implementar.
- É necessário:
 1. Que a gramática não seja recursiva à esquerda
 - $A \rightarrow Aa$ se tornará
ReconheceA() { ReconheceA();... }
 - Recursão infinita!
 2. Que a gramática seja fatorada à esquerda
 - Senão, há ambiguidade na escolha da derivação.
 3. Que os primeiros terminais deriváveis possibilitem a decisão de uma produção a aplicar!
 - Não há retrocesso sobre não-terminais...

INF01033 - Compiladores B - Marcelo Johann - 2012/1

Aula 07 : Slide 32

Implementação

- CMD: IF | ASS | WHILE | BL;
- BL: '{' CMDL '}';
- CMDL: CMD RESTO;
- RESTO: ';' CMDL;
- IF: "if" E "then" CMD;
- WHILE: "while" E "do" CMD;
- ASS: L '=' E;
- L: '*' E | E;
- E: NUM | ID;

INF01033 - Compiladores B - Marcelo Johann - 2012/1

Aula 07 : Slide 33

Etapas...

...voltando ao yacc

INF01033 - Compiladores B - Marcelo Johann - 2012/1

Aula 07 : Slide 34

Leituras e Tarefas sugeridas

Ler e Rer o Livro
Implementar outra GLC
Ligar sua implementação com yacc
Ligar com autômato

INF01033 - Compiladores B - Marcelo Johann - 2012/1

Aula 07 : Slide 35