Construção de Compiladores
Aula 7 - Expressões

Bruno Müller Junior

Departamento de Informática
UFPR

31 de Agosto de 2014

1 Expressões

- Gramática não ambígua
- Geração de Código
- Verificar os tipos
- Exemplo
- Implementação
- Expressões Boolenas
- Implementação

Expressões

- A análise de expressões apresentam três aspectos críticos:
  1. Obter uma gramática não ambígua.
  2. Gerar código;
  3. Conferir o tipo (análise semântica).

Gramática não ambígua

- O livro do Tomasz apresenta uma gramática não ambígua.
- A gramática já inclui regras de precedência.
- Iremos usar uma versão mais sucinta, que derivada daquela.
- Construa a árvore sintática para a entrada α="a+a and b"

```
E ::= E + T | E or T  | T
T ::= T * F | T and F | F
F ::= IDENT | (E)
```

Geração de Código

- Para gerar código, utilizamos a tradução dirigida pela sintaxe (TDD-aula 3).
- A idéia é acrescentar os nós executáteis ao final da regra, como feito no projeto `Posfixo`.
- Exemplo:

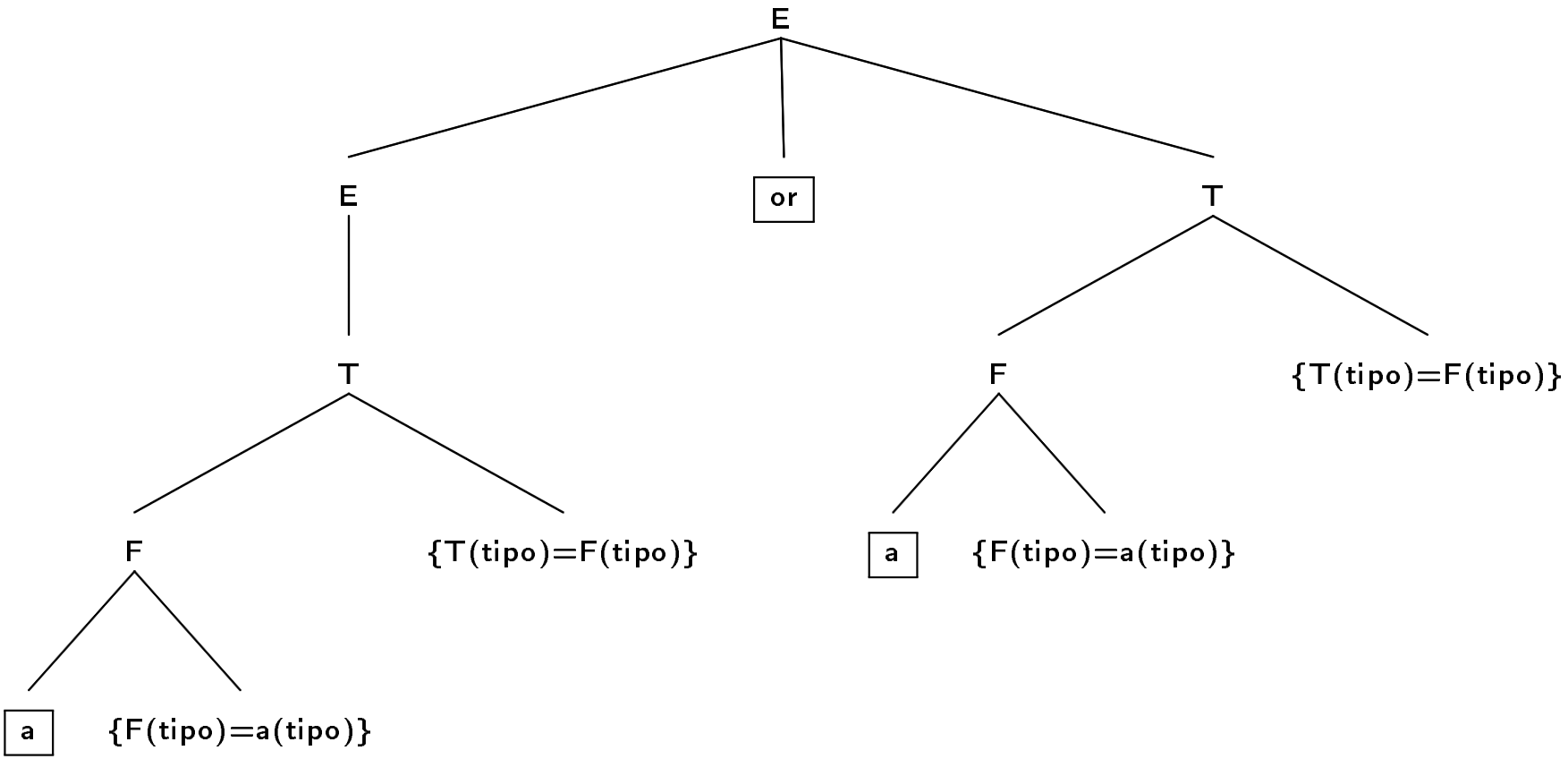
```
E ::= E + T    { printf("SOMA") |
      E or T   { printf("DISJ") |
      T
```

Verificar os tipos

- Também se utiliza a TDD, porém de uma forma mais elaborada:
  1. associar atributos a cada variável (lado esquerdo da produção).
  2. utilizar o nó executável para movimentar o valor do atributo de cima para baixo (atributo sintetizado) ou de baixo para cima (atributo herdado).
- Exemplo:

```
E1 ::= E2 + T    { Se (E2.tipo == T.tipo == E1.tipo == inteiro)
                       então E1.tipo := E.tipo }
     E2 or T     { Se (E2.tipo == T.tipo == E1.tipo == booleano)
                       então E1.tipo := E.tipo }
     T           { E1.tipo := T.tipo }
```

Exemplo


E
E
or
T
T
F
{T(tipo)=F(tipo)}
F
{T(tipo)=F(tipo)}
a
{F(tipo)=a(tipo)}
a
{F(tipo)=a(tipo)}

Implementação

- O bison contém mecanismo para referenciar os elementos de uma produção usando os símbolos $$, $1, $2, ...
- O nome usa a posição. Na produção `E ::= E + T`, temos:

  $$ == `E` (o da esquerda, antes do `::=`)
  $1 == `E` (o da direita, após do `::=`)
  $2 == `+`
  $3 == `T`

- Exemplo: `E ::= E + T {$$=$1+$2}`

Implementação

- Uma alternativa que deixa o código mais legível é utilizar uma ou mais pilhas.

```
E ::= E + T { t1 = desempilha(E);
              t2 = desempilha(T);
              se (t1 == t2)
                 então empilha(E,t1);
                 senão erro("....");
            }
```

Expressões Boolenas

- A MEPA contém várias instruções que consideram operandos booleanos.
- Todas são semelhantes: fazem a comparação entre M[s-1] com M[s], recolocando o resultado em M[s].

Instrução	Ação	
CMIG	M[s-1]=(M[s-1]==M[s]) s:=s-1 i:=i+1	Compara Igual
CMMA	M[s-1]= (M[s-1]>M[s]) s:=s-1; i:=i+1	Compara Maior
CMME		
CMDG		
...		
CONJ	M[s-1]=(M[s-1] and M[s]) s:=s-1 i:=i+1	AND
...		

Exemplo de Tradução	
Expressão	Código MEPA equivalente
a>b and b=c	CRVL a CRVL b CMMA CRVL b CRVL C CMIG CONJ

Implementação

- A regra que traduz expressões booleanas é:

```
25. <expressão> ::= <expressão simples> [ <relação> <expressão simples> ]
26. <relação>   ::= = | <> | < | <= | > | >=
```