第八章 语法制导翻译和中间代码生成

8.8 数组和结构的翻译

广东工业大学计算机学院

8.8 数组和结构的翻译

- 8.8.1 数组说明和数组元素的引用

数组特点

- **(1)** 从逻辑上说，一个数组是由同一类型数据所组成的某种**n**维矩形结构。每维的下标只能在该维的上、下限之内变动。

- **(2)** 数组的每个元素(也称下标变量)是由数组名连同个维的下标值命名的，如：

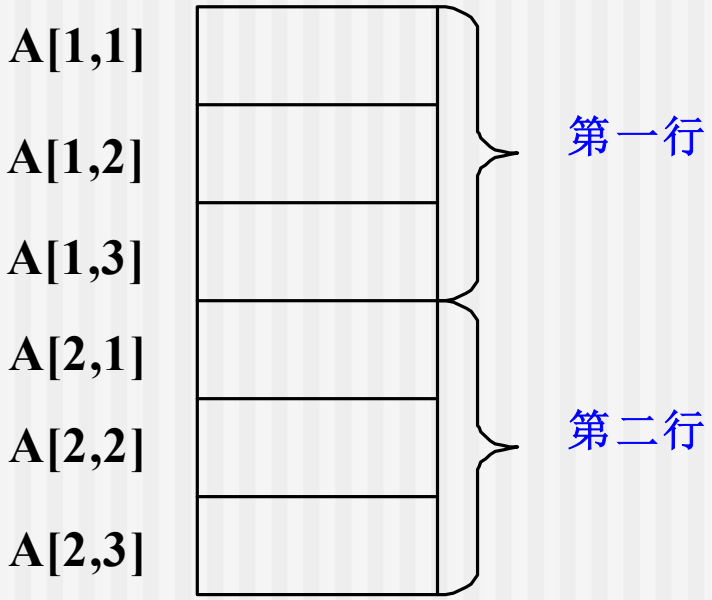
$$A[i_1, i_2, \ldots, i_n]$$

- **(3)** 如果一个数组所需的存储空间的大小在编译时就已知道，则称此数组是一个静态数组；否则，称为可变数组或者动态数组。

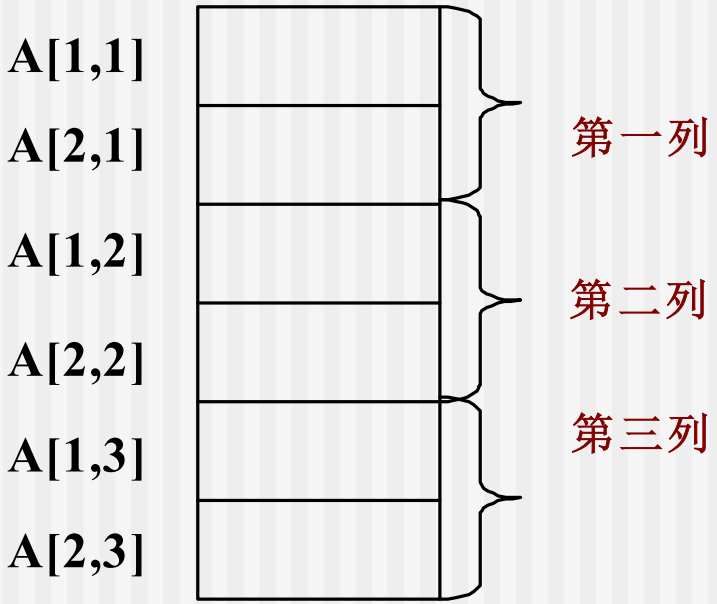
数组的存储表示

- 数组的存储表示有多种形式，最简单的一种是把整个数组按行(或按列)存放在一片连续存储区中。

- 例如，假设A是一个2*3的二维数组：



数组按行存放



数组按列存放

数组元素的地址计算举例

- 在以行为序的情形下，如何计算数组元素的地址？

- 例如：一个二维数组**A**各维的下限为**1**，上限为**20**，每个元素占用一个机器字**(**设存储器是以字编址**)**，数组元素**A(i, j)**的地址为：

$$a+(i-1)\times 20+(j-1)$$

- 或等价的表示成：$a-21+(20i+j)$

数组元素的地址计算的一般方法

- 设**A**是下面说明语句所定义的一个**n**维数组

$$array \quad A[l_1 : u_1, l_2 : u_2, \ldots, l_n : u_n]$$

- 令$d_i = u_i - l_i + 1$，a是数组首地址，那么按行排列时，元素$A[i_1, i_2, \ldots, i_n]$的地址D是：

$$D = a + (i_1 - l_1)d_2 d_3 \cdots d_n + (i_2 - l_2)d_3 d_4 \cdots d_n + \cdots + (i_{n-1} - l_{n-1})d_n + (i_n - l_n)$$

- 经因子分解后得：**D = CONSTPART + VARPART**

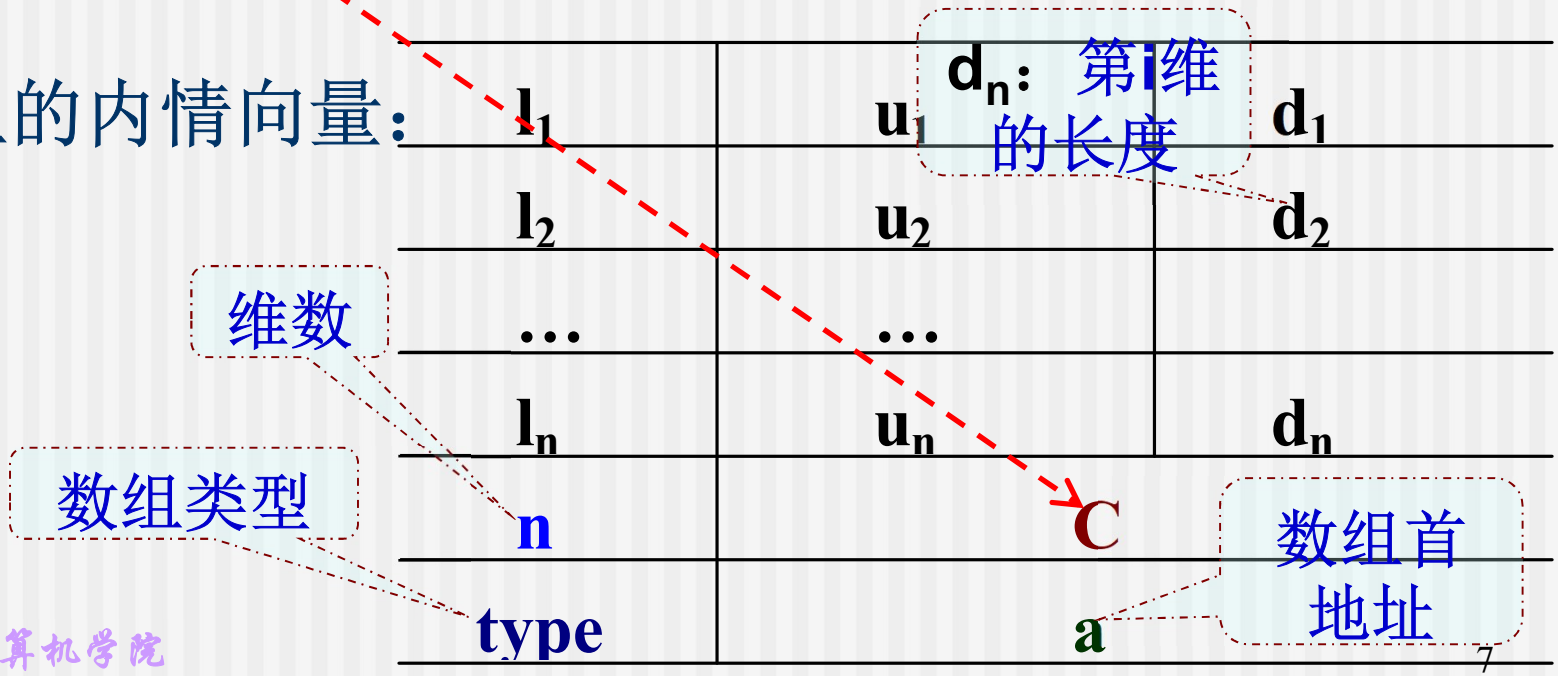
$$CONSPART = a - C$$

$$C = (\cdots((l_1 d_2 + l_2)d_3 + l_3)d_4 + \cdots + l_{n-1})d_n + l_n$$

$$VARPART = (\cdots((i_1 d_2 + i_2)d_3 + i_3)d_4 + \cdots + i_{n-1})d_n + i_n$$

数组的内情向量

- 对于数组$A[l_1 : u_1, l_2 : u_2, \ldots, l_n : l_n]$，CONSPART只需计算一次，在编译时计算出来。
- 在计算它的任何元素的地址时就可直接引用此值。
- 数组的内情向量：



内情向量的算法(了解)

- 数组A[$l_1 : u_1, l_2 : u_2, \ldots, l_n : l_n$]
- **begin**
- **i := 1; m := 1; c := 0;**
- **while i <= n do**
- **begin**
- **di = ui –li + 1;**
- **m := m * di;**
- **c := c * di + li;**
- **li, ui和di --> 向量表区;**
- **i := i + 1;**
- **end;**
- 申请m个单元的数组空间，令该空间的首地址是a;
- **end;**

l_1	u_1	d_1
l_2	u_2	d_2
…	…	
l_n	u_n	d_n
n	C	
type	a	

访问数组元素$A[i_1, i_2, \ldots, i_n]$

- 产生两组计算数组元素地址的四元式：
- ① 一组计算**VARPART**，将它放在某个临时单元**T**中；
- ② 一组计算**CONSPART**，将它放在另一个临时单元**T1**中。
- 同时用**T1[T]**表示数组元素的地址。
- 对应数组元素引用和对数组元素赋值的四元式：
- **(1)** 变址取数的四元式：

  (=[], T1[T], -, X) **/* 相当于X := T1[T] */**
- **(2)** 变址存数的四元式：

  ([]=, X, -, T1[T]) **/* 相当于T1[T] := X */**

数组元素的翻译举例1

- 例如A是一个10*20的数组，即d1 = 10，d2 = 20。则赋值语句X := A[I, J]的四元式序列：
- (*, I , 20, T1)　　/* 其中20指d2 */
- (+, J , T1, T1)　　/* T1为VARPART */
- (–, A, 21, T2)　　/* 相当于T2: = a – C */
- (=[ ], T2[T1], _, T3)　　/* T3= T2[T1] */
- (:=, T3, _ , X)

数组元素的翻译举例2

- A[I+2, J+1] := M + N该如何翻译成四元式?
- (+, I , 2, T1)
- (+, J ,1, T2)
- (*, T1 , 20, T3)
- (+, T2, T3, T3)　　/* T3= VARPART */
- (- , A , 21, T4)　　/* T4= CONStPART */
- (+, M, N, T6)
- ([ ]=, T6, _ , T4[T3])　　/* T4[T3]= T6 */

练习（07级考题）

- 考虑以下语法制导翻译的定义

产生式	语义规则
$S \rightarrow L_1 \ . \ L_2$	Print(L_1.val + L_2.val * $2^{-L2.num}$)
$L \rightarrow L_1 \ B$	L.val := 2 * L_1.val + B.val L.num := L_1.num + 1
$L \rightarrow B$	L.val := B.val L.num := 1
$B \rightarrow 0$	B.val := 0
$B \rightarrow 1$	B.val := 1

- （1）写出句子**11.01**的带注释分析树（属性计算过程）。
- （2）给出处理该句子**11.01**的结果（**Print**输出结果）。

练习（09级考题）

（10分）试完成下列语句翻译的四元式序列。

已知源程序如下：

```
        prod:=0;
        i:=1;
        while i≤20 do
         begin
prod:=prod+a[i]*b[i];
i:=i+1
         end;
```

其中，设A是数组a的起始地址，B是数组b的起始地址；机器按字节编址，每个数组元素占四个字节。

练习（09级考题）

```
100  prod:=0
101   i:=1
 102   if i≤20 goto 104
 103   _____
 104   T1:=4*i
 105   T2:=A-4
 106   _____
 107   T4:=4*i
 108   T5:=B-4
 109   _____
 110   T7:=T3*T6
 111   _____
 112   i:=i+1
 113   _____
 114   …
```

103 goto 114
106 T3:=T2[T1]
109 T6:=T5[T4]
111 prod:=prod+T7
113 goto 102

（备注：**T3**和**T6**可混用）

作业

- 写上日期
- 及时上交

主观题 10分

设置

2. 给定文法 $G[S]$：

$S \rightarrow (L) \mid a$

$L \rightarrow L, S \mid S$

如下是相应于 $G[S]$的一个属性文法(或翻译模式)：

$S \rightarrow (L)$	$\{S.\text{num} := L.\text{num} + 1;\}$
$S \rightarrow a$	$\{S.\text{num} := 0;\}$
$L \rightarrow L_1, S$	$\{L.\text{num} := L_1.\text{num} + S.\text{num};\}$
$L \rightarrow S$	$\{L.\text{num} := S.\text{num};\}$

图 7.19 分别是输入串(a,(a))的语法分析树和对应的带标注语法树，但后者的属性值没有标出，试将其标出(即填写图 7.19 右图中符号＝右边的值)。

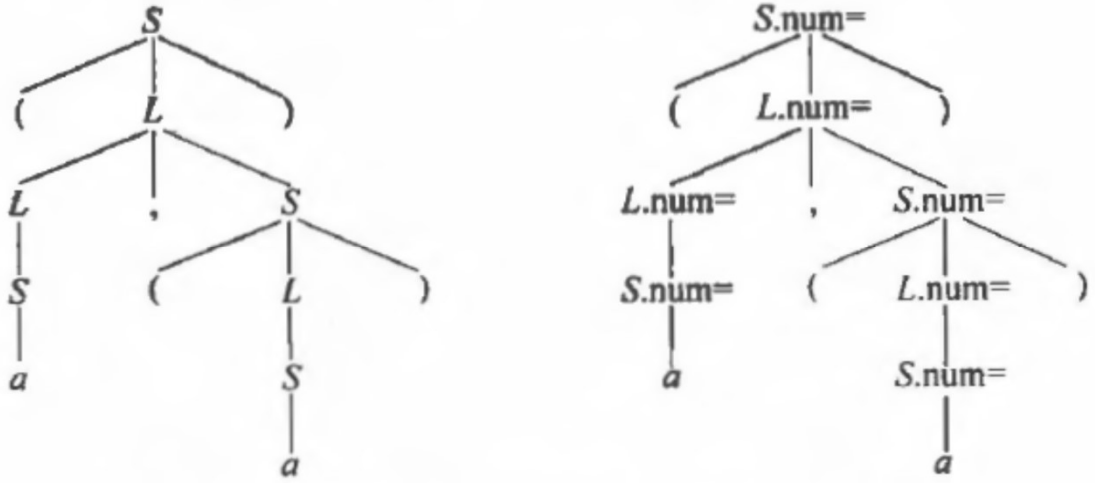


图 7.19 题 2 的语法分析树和带标注语法树

作答

主观题 10分

设置

3. 下面是一个简单表达式文法 $G[S]$的一个仅含综合属性的属性文法(开始符号为 S):

$S \rightarrow E$	{print(E. val)}
$E \rightarrow E_1 + T$	{E. val := E_1. val + T. val}
$E \rightarrow T$	{E. val := T. val}
$T \rightarrow T_1 * F$	{T. val := T_1. val × F. val}
$T \rightarrow F$	{T. val := F. val}
$F \rightarrow (E)$	{F. val := E. val}
$F \rightarrow d$	{F. val := d. lexval}

其中,d. lexval 是由词法分析程序所确定的属性; F. val、T. val 和 E. val 都是综合属性; 语义函数 print(E. val)用于显示 E. val 的结果值。不难理解,这个属性文法描述了一个基于简单表达式文法进行算术表达式求值的语义计算模型。

试给出表达式 3 * (5+4) 的语法分析树和相应的带标注语法分析树。

正常使用主观题需2.0以上版本雨课堂

作答

主观题 10分

设置

将布尔表达式a>b and c<d or e>f翻译成三地址码序列（从100开始编号），并将去往真假出口的三地址码分别拉成真链、假链，同时指出真链、假链的链首（三地址码编号）。

正常使用主观题需2.0以上版本雨课堂

作答

主观题 10分

设置

请将把下列语句翻译成四元式，将下列四元式序列填写完整。

While (A>B) do

If (C>D) then X:=X*Y

Else X:=Y+Z

100 if A>B goto 102

101 goto ①

102 if C>D goto ②

103 goto 107

104 T1:=X*Y

105 ③

106 goto 109

107 ④

108 X:=T2

109 goto ⑤

作答

附录

- 递归下降语法制导的翻译

递归下降语法制导的翻译

自下而上分析法适应更多的文法，因此，自下而上分析制导翻译技术也受到普遍重视。但是，自上而下分析法也有自下而上分析法不可取代的优点，它可以在一个产生式的中间调用语义子程序。

例如，假定我们正在为非终结符 **A** 寻找匹配，并已确定用**A**的候选 **BCD** (即用**A→BCD**规则)，那么可分别在识别出 **B**、**C** 和 **D** 之后直接调用某些语义子程序，而无须等到整个候选式匹配完之后。

递归下降分析制导翻译技术，它的特点是将语义子程序嵌入到每个递归过程中，通过递归子程序内部的局部量和参数来传递语义信息。

考虑关于算术表达式的文法**G[E]**：

E→T{+T}

T→F{*F}

F→i | (E)

制导翻译程序

```
E()                                    /* E→T{+T} */
{
  E(1).place=T( );                     /* 调用过程 T */
  do {
     scaner( );          /* 读进下一个符号 */
     E(2).place=T( );
     temp = newtemp;
     emit(temp=E(1).place+E(2).place);
     E(1).place = temp;
  } while( sym =='+');
  return(E(1).place);
}
```

```
T()                                          /* T→F{*F} */
{
   T(1).place=F();
   do {
      scaner();
      T(2).place=F();
      temp=newtemp;
      emit(temp=T(2).place * T(1).place);
      T(1).place = temp;
   }while(sym=='*');
   return(T(1).place);
}
```

```
F()                                  /* F→(E)|i */
{
  if(sym=='i') {
    scaner();
    return(i.place);
  }
  else if(sym=='(') {
    scaner();
    F.place=E();
    if(sym==')') {
      scaner();
      return(F.place);
    }
    else error();
  }
  else error();
}
```