编译原理

Compiler Construction Principles

朱 青

信息学院计算机系，

中国人民大学，

zqruc2012@aliyun.com

第7章: 符号表

- 7.1 符号表的组织和使用
- 7.2 符号表的管理
  - 7.2.1 线性表
  - 7.2.2 对折查找与二叉树
  - 7.2.3 杂凑技术
- 7.3 名字的作用范围
  - 7.3.1 FORTRAN的符号表组织
  - 7.3.2 一个PASCAL符号表的组织

7.1 符号表的组织和使用

- 符号表的作用
- 收集符号属性
  - 根据说明语句在符号表建立符号的相应属性信息
- 上下文语义合法性检查的依据
  - 标识符可能在程序多个地方出现
  - 根据符号表信息检查一致性

  例如： int i[3];

  …

  float i[3];
- 目标代码生成阶段地址分配的依据
  - 根据定义的位置，确定符号变量被分配的区域
  - 根据出现的次序，决定符号变量在某个区域的具体位置

符号表的内容

- 关键字，操作符，标识符通常有各自的符号表
- 标识符符号表的项目属性
  - 符号名
    - 重名标识符按作用域规则处理
  - 符号类型
    - 种属：常量，变量，数组，函数，过程，标号等
    - 类型：整形，实型，字符型，布尔型等
  - 存储地址
  - 所属区域信息
    - 分程序嵌套结构
  - …

符号表的操作

- 创建符号表
  - 在编译开始时进行
- 插入表项
  - 在遇到新的标识符声明时进行
- 查询表项
  - 在引用声明过的标识符时进行
- 修改表项
  - 在获得新的语义值信息时进行
- 释放符号表的空间
  - 在编译结束前进行

符号表的操作

- 翻译过程中,要不断查填符号表. 符号表中记源程序中的各种名字及其相关信息,符号表可以是一张表或若干张表(如:变量表,常量表,标号表,外部过程表,…).

- 符号表的组成:

  名字栏 (NAME)　信息栏(INFORMATION)

- 不同的表所含信息不同.常见的有类型(整型,实型,…),种属(一般变量,数组,形参,...).使用性出现或定义性出现,地址… .
- 一般地,在词法分析和语法分析阶段填符号表,而在语义分析和生成代码阶段查符号表.

编译程序对符号表的五类基本操作;

1) 对给定名字,确定此名是否在表中.

2) 填如新名.

3) 对给定名字,访问它的有关信息.

4) 对给定名字,填写或更新它的某些信息.

5) 删除一个或一组无用的项.

符号表的形式:

(一)标识符直接存放在名字栏中:

	名字栏	信息栏
1	X1	
2	MATRIX	
3	A	

(二)标识符存放在字符串表中的一种方式:(间接)

名字栏	信息栏
,2	
,1	
,12	

A 1 B I N N E R P R O D U C T…　　字符串表

第7章: 符号表

- 7.1 符号表的组织和使用
- 7.2 符号表的管理
  - 7.2.1 线性表
  - 7.2.2 对折查找与二叉树
  - 7.2.3 杂凑技术
- 7.3 名字的作用范围
  - 7.3.1 FORTRAN的符号表组织
  - 7.3.2 一个PASCAL符号表的组织

7.2 符号表的管理

符号表机制允许用户方便地加入新的表项并能有效地查找已经存在的表项.实际上,对符号表的管理有:创建,插入,删除,查找,修改,…等功能操作.而构造和查找是最重要的.

符号表常用的三种构造法和处理法:线性表,二叉树和杂凑技术.

符号表的组织方式

- 简单方式
  - 特定：各项各栏存储长度固定
  - 优点：易于组织，更改，查找
  - 缺点：浪费存储空间
- 间接存储方式
  - 特定：符号表由多个表构成，各符号属性存放在不同位置
  - 优点：占用空间少
  - 缺点：管理稍复杂

符号表的数据结构

- 线性表
  - 填写：按出现顺序依次填入
  - 查找
    - 正序，反序：复杂度O (n)
    - 填写新名字前要先查找
  - 优点：实现简单，占用存储空间小
  - 缺点：查找效率低
- 排序组织
  - 根据名字的机器内码大小对线性表顺序化
  - 查找：二分法，复杂度O(logn)
  - 缺点：排序代价大

符号表的数据结构

- 二叉树
  - 遇到的首个名字为根结点
  - 大于根结点的放在左（右）子树，小于根节点的放在右（左）子树
  - 查找复杂度O(logn)
- 哈希表
  - 用哈希函数H实现从标识符名称到表项位置的直接映射
  - 采用链表法解决地址冲突。
  - 访问表项的时间效率约为O(1)。

7.2.1 线性表:

构造方法:

按关键字出现的顺序填写各项.

查找方法:

从第一个开始,顺序查找.

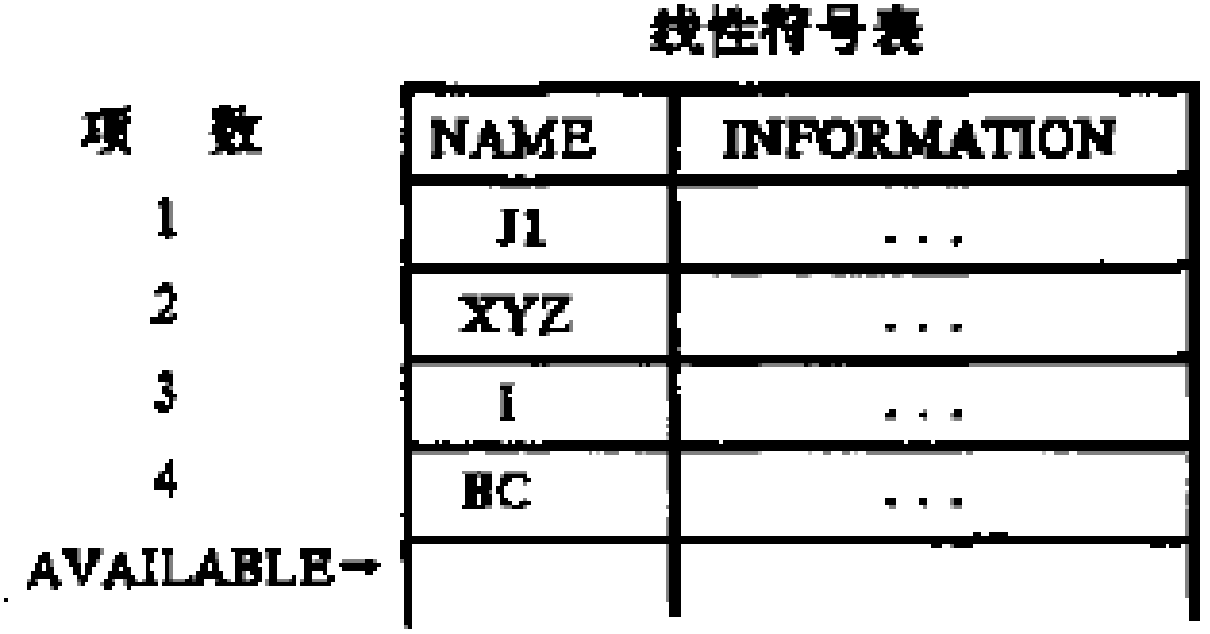
数据结构:

结构数组式符号表, 链表式符号表.

图示(P226) 图8.5

特点:填表快,查表慢.

线性符号表——结构数组表示



7.2.2 对折查找与二叉树

构造方法:

在造表的同时把表格中的项按名字的“大小”顺序整理排列. 构造成一个有序二叉树(如:左大,右小). 图示(P227)

查找的方法:

对折法.(算法思想:P227)

有序二叉树的遍历.(算法:P228)

特点:填表慢,查表快.

对折查找： 把表格中的项按名字的大小顺序整理排列。大小：名字的内码二进制值。

线性符号表

项 数	NAME	INFORMATION
1	BC	. . .
2	I	. . .
3	J1	. . .
4	XYZ	. . .
AVAILABLE→		

图 线性表

二叉树：

令每项是一个结点，每个结点附设两个指示器栏，一栏为**left**，一栏为**right**。每个结点的主栏内码值被看成是代表该结点的值。

要求：

任何结点**p**右枝的所有结点值均应小于结点**p**的值，而左枝的任何结点枝均应大于结点**p**的值。

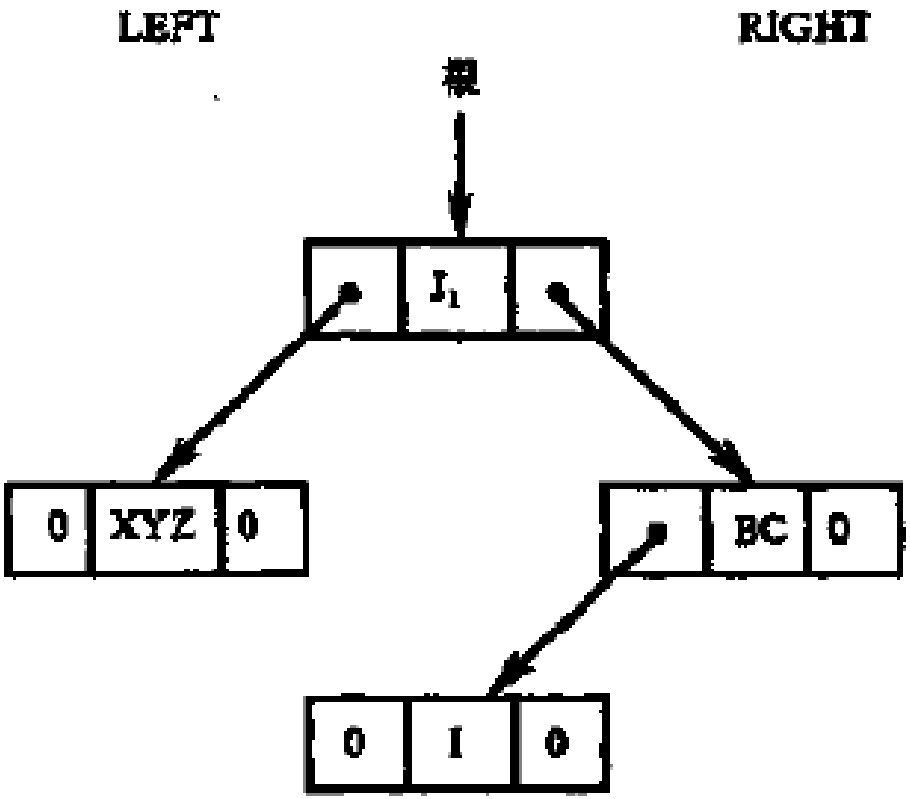


图　　线性表的二叉树表示

7.2.3 杂凑技术:

采用一种更有效的方法:查表填表都高速进行.构造地址函数H(杂凑函数).

地址函数H有两点要求:

1) 函数的计算要简单,高效.
2) 函数值能比较均匀地分布在0~N-1之间.

若N为质数,可用除留余数法.

H(SYM)=SYM%N

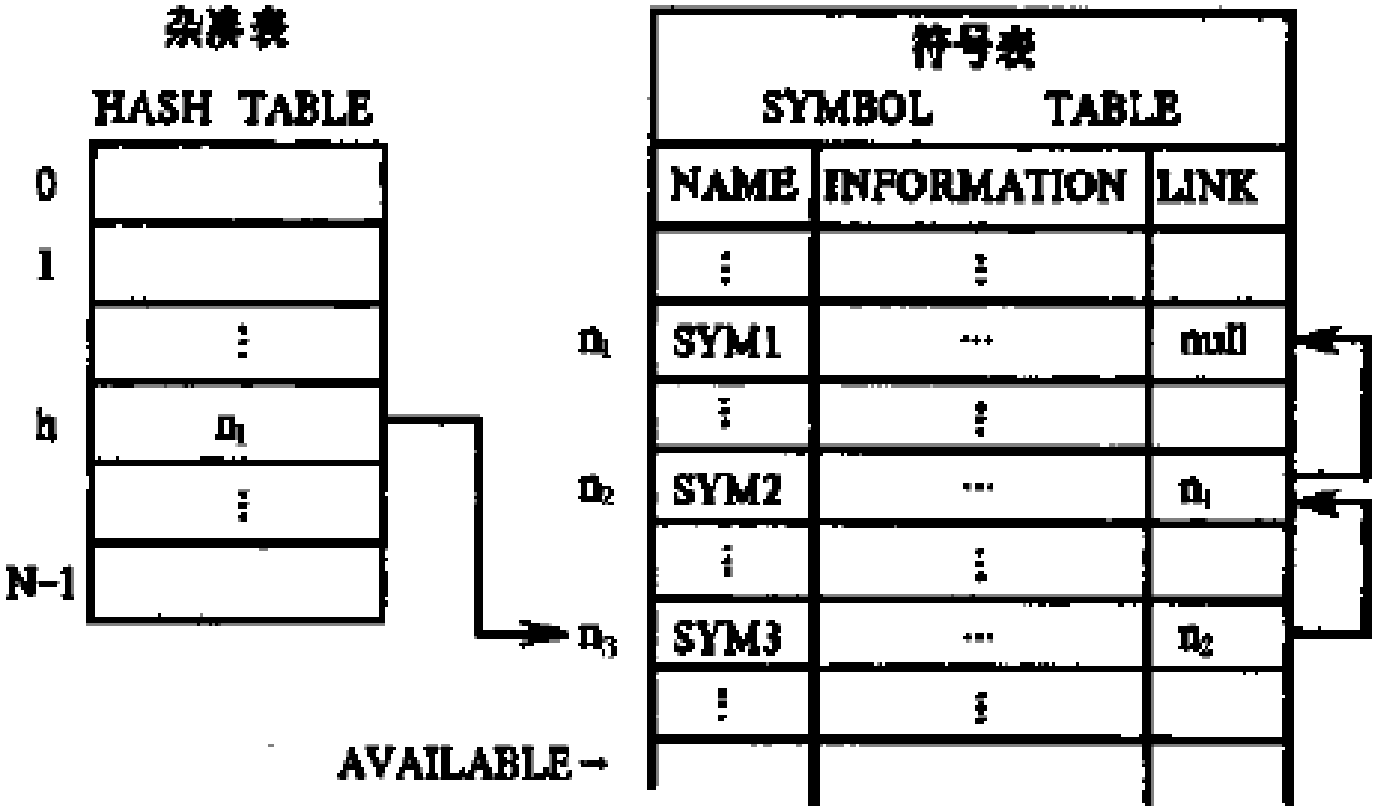
解决地址冲突的办法:

线性查填解决法(线性探测再散列).

随机探测再散列.

例如:(P229)一张杂凑(链)表通过间接方式查填符号表.

杂凑技术示意图


杂凑表
HASH TABLE
0
1
h
N-1
n_1
符号表
SYMBOL TABLE
NAME
INFORMATION
LINK
n_1
SYM1
null
n_2
SYM2
n_1
n_3
SYM3
n_2
AVAILABLE

第7章: 符号表

- 7.1 符号表的组织和使用
- 7.2 符号表的管理
  - 7.2.1 线性表
  - 7.2.2 对折查找与二叉树
  - 7.2.3 杂凑技术
- 7.3 名字的作用范围
  - 7.3.1 FORTRAN的符号表组织
  - 7.3.2 一个PASCAL符号表的组织

7.3 名字的作用范围

- 在许多程序语言中,名字往往有一个确定的作用域,在不同的地方可能被说明为标识不同的对象.即不同的标识符,具有不同的性质,要求分配不同的存储空间.

- 为标识符的正确引用,每一个作用域(分程序或程序段)都保持一个单独的符号表.

  一个名字看成是一个二元式:

  (分程序编号，标识符)

7.3.1 FORTRAN的符号表组织

介绍一个一遍扫描完成的FORTRAN编译的符号表的组织.此编译共有七张表.

局部表有:

名表NT,可执行标号表LT,格式标号FT,常数表CT,循环层次表DT.(一段程序处理完就可拆除).

全局性表:

标准过程名表BEF,外部过程名表PT

表在内存中的存放顺序:(P230 图8.9)

1)名表NT和形实结合

名表NT的表项:

勾链字 TYPE(类型) CAT(属性)

DIM(参数个数) ADD(地址)

NAME(名字)

不同属性名字,地址不同.(p235)

变量 : 栈地址.

公用变量 : 公共区地址.

数组 : 数组字的头地址.

语句函数名 : 入口地址.

哑元 : 栈地址(存放实元).

哑元结合形式有三种:

传地址，传值，传名

2) 可执行标号表LT和拉链返填

表项形式: 勾链字

U(0--定义性出现.1--使用性出现.)

AV(0--不能被引用, 1--能被引用)

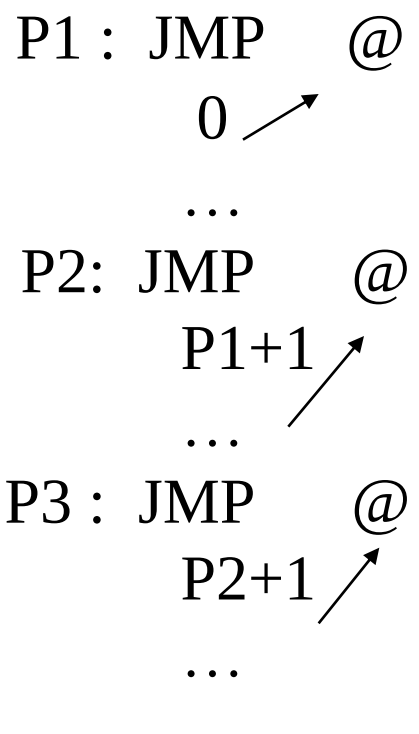
ADD(地址)

为解决标号的先使用后定义,采用拉链返填技术.

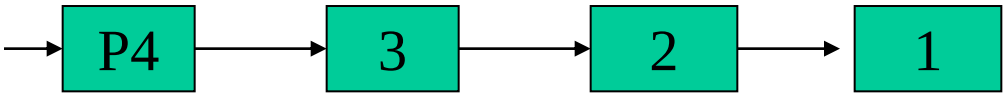
如:

```
        GOTO   100   (1)
          …
        GOTO  100    (2)
          …
        GOTO   100   (3)
         …
        100   A=B+C   (4)
```

逆方向建立链表:



当执行完语句(4)时,将定义的地址,沿着这个链填回到使用处.

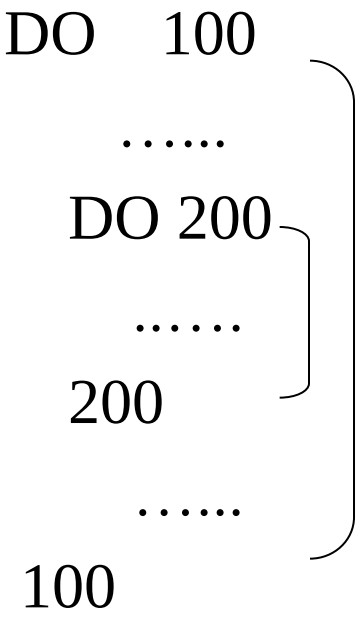


P1：JMP @
P4
…
P2: JMP @
P4
…
P3：JMP @
P4
…
P4: (100定义地址)

3)循环层次表DT

循环体头　终结句标号

CE(进入循环体后, 将标号表的当前位置记录在相应链中).



4)外部过程名表PT

数据项:

NAME(外部过程名)

U= 0 (定义性出现)

1 (使用性出现)

F= 0 (子程序)

1 (函数)

DIM (参数个数)

MASK (参数种属链)

ADD(地址:存放过程段入口地址的地址).

7.3.2 一个Pascal符号表的组织

Pascal是分程序结构,名字的作用域是它被说明了的最小分程序.

Pascal是嵌套结构,内层说明的变量不可在外层被引用. 因此符号表是一个栈.

一个分程序结束时,它所定义的局部名已不能再引用,但为了后继遍历必须保留.

通常,把符号分成两部分,一部分记尚未处理完的“活名”,另一部分记处理完的“死名”.

用杂凑链组织符号表

Pascal符号表的组织有:

1) 符号表: 分成两部分(死名,活名).

每项有四个内容.

(1) 一个指示器(指向名字在字符数组里的位置,及长度).

(2) 一个数:表示此名所属分程序的编号.

(3)一个指向信息区的指示器.

(4)只是具有同一杂凑值的链指示器.

2)字符串数组: 存放所有的名字.

3)信息区:存放信息.(中间代码中凡名字
都写的是信息区中的有关地址.)
4)杂凑表:指针数组,每项的内容是一个链
头,该链是由具有同一杂凑值的活名组成.
5)分程序表:两部分(活性分程序,已处理
完的分程序).每项有三个内容.
(1)指向分程序在符号表中的位置的指针.
(2)指向字符数组中分程序局部名开始
位置的指针.
(3)记本分程序中局部名个数的计数器.

分程序结构的符号表组织

- 名字的作用域：最近嵌套原则
- 单表结构：
  - 所有的标识符都记录在同一张符号表中
  - 表项中设置一个描述作用域嵌套层次的属性
  - 设置一个记录当前作用域嵌套层次的变量，每进入一个作用域，当前嵌套层次增1
  - 同名标识符，嵌套层次最深的最先被检索到
  - 每退出一个作用域，删除该作用域中声明的所有标识符

分程序结构的符号表组织

- 子表结构：栈式结构
  - 为每个过程建立一张单独的符号表，用于记录在当前作用域中声明的标识符
  - 各过程的符号表指针都保存在一个后进先出的作用域栈中
  - 每进入一个过程，建立一个空的符号表，该符号表的指针入栈
  - 访问一个标识符时，按照自顶向下的顺序在作用域栈的各符号表中查找
  - 每退出一个作用域，释放相应的符号表的空间，该符号表的指针出栈

分程序结构的符号表组织

	单表结构	分表结构
优点	访问标识符的操作非常方便。	退出作用域的操作非常方便。
缺点	退出作用域时，要遍历整张表以便删除该作用域中声明的标识符。	访问全局变量时，要先在各局部作用域中查找，增加了时间开销。

Pascal符号表的组织的实例

如图8.10 (P232)

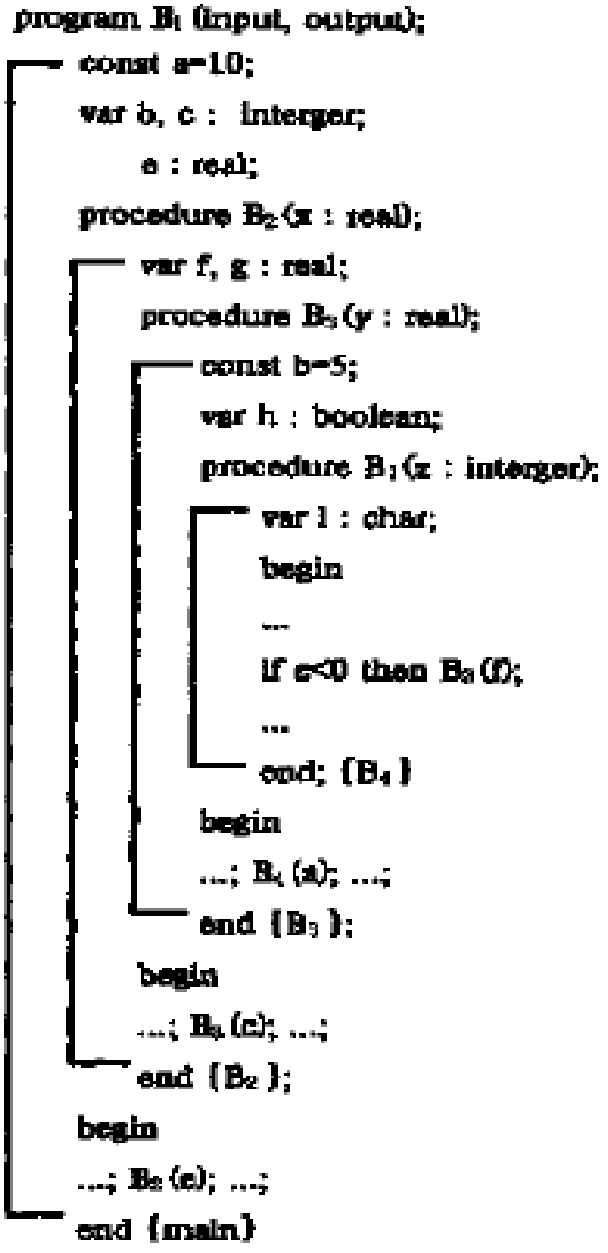
符号表的管理:

1.填.

2.查.

3.处理分程序头.

4.处理分程序尾.


program B1 (input, output);
const a=10;
var b, c : integer;
e : real;
procedure B2 (x : real);
var f, g : real;
procedure B3 (y : real);
const b=5;
var h : boolean;
procedure B4 (z : integer);
var l : char;
begin
...
if e<0 then B3 (f);
...
end; {B4 }
begin
...; B4 (a); ...;
end {B3 };
begin
...; B3 (c); ...;
end {B2 };
begin
...; B2 (e); ...;
end {main}

如图8.10 (P232)

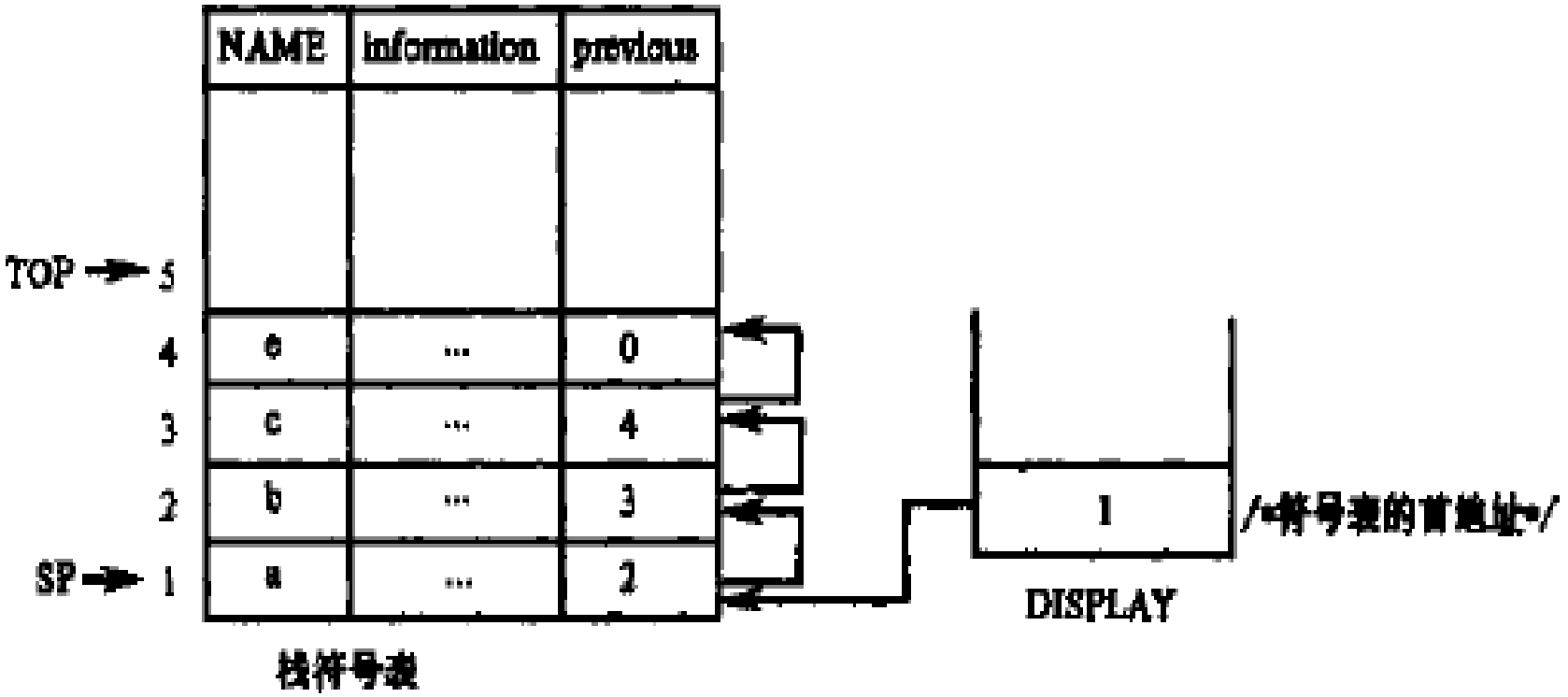

NAME
information
previous
TOP
5
4
e
...
0
3
c
...
4
2
b
...
3
SP
1
a
...
2
1
/*符号表的首地址*/
DISPLAY
栈符号表

图 8.11　栈符号表

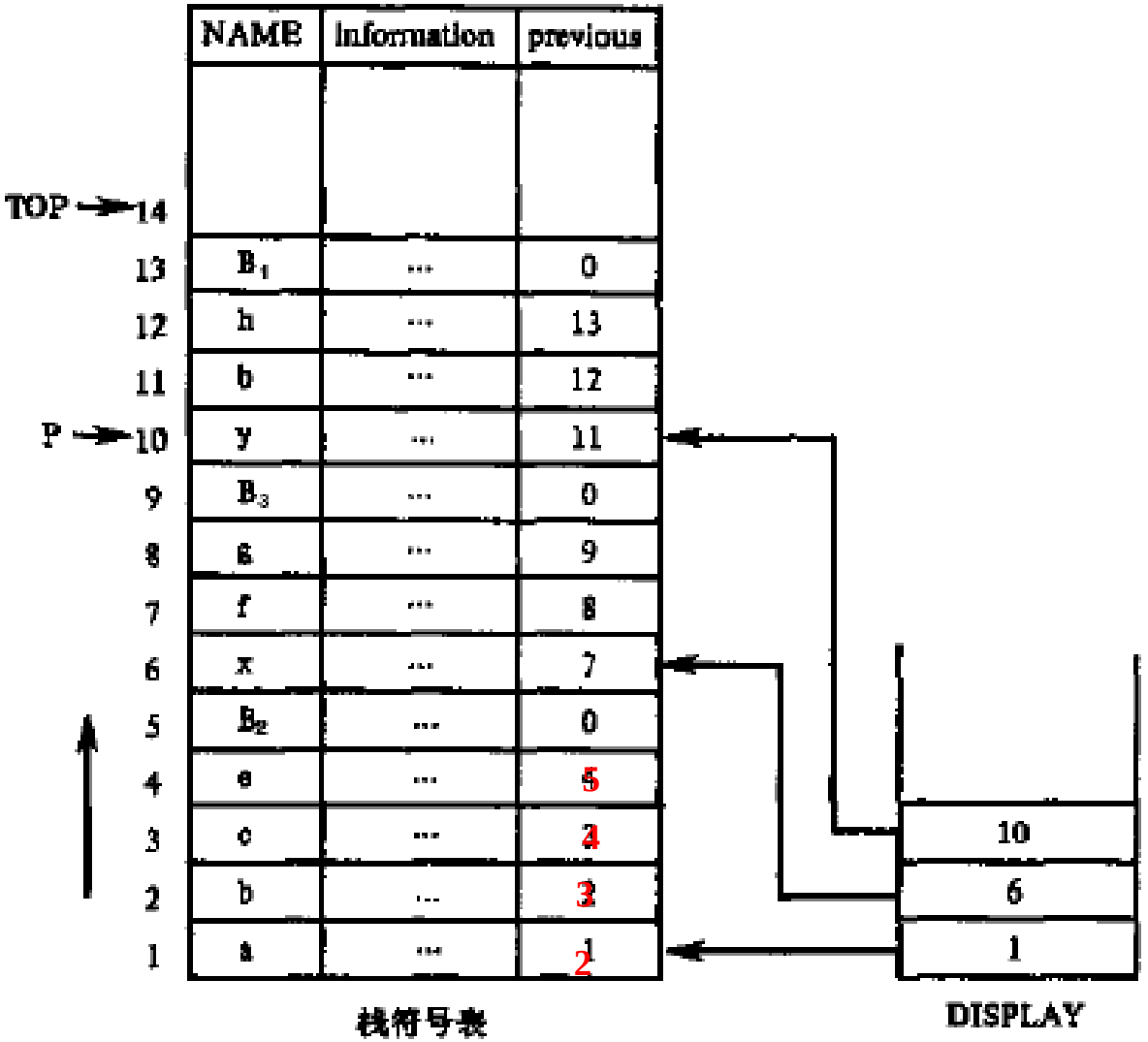

NAME
information
previous
TOP
P
14
13
12
11
10
9
8
7
6
5
4
3
2
1
B_1
h
b
y
B_3
g
f
x
B_2
e
c
b
a
...
0
13
12
11
0
9
8
7
0
5
4
3
2
1
10
6
1
栈符号表
DISPLAY

图 8.12　栈符号表