DJS—8计算机

FORTRAN IV 编译系统设计说明

第四分册

连接编辑

中国科学院高能物理研究所七室
FORTRAN IV编译系统设计组

1981年12月

目　　录

C

§1. 连接编辑过程

1·1 控制

连接编辑生成最后的目标代码，在逻辑功能上是编译的最后一个阶段，同编译系统其余部分一样占有用户存贮空间。但是，它在系统结构中所处的地位不同于编译的前面三个阶段。它接受PPP系统作业控制进程的控制，作为一个独立的作业步。它在PPP系统态而不是在目标态下工作。

1·2 中间代码

连接编辑的输入是半目标模块集。

半目标模块的结构如下：

MOD
ESD
PUB
COM
HTD
RLD

其中MOD为模块信息，指示模块项的位置。ESD为外部符号字典，记录程序段访问的外部过程。PUB为共名字典，COM为公用字典，分别登记程序段中说明的共名量和公用量。HTD为半目标字典，包含程序段的全部可重定位的数据和指令。RLD为重定位字典，提供半目标数据和指令的相对位置和容量。

半目标模块集在编译的第三阶段形成后记录在编译工作文件BWG

上，用户也可以将其存放到纸带文件或磁带文件中。连接编辑前，当半目标模块集不在BWG时要先从其他存放介质转移到BWG上。为了减少用户对系统内存资源的依赖，半目模块集是以模块为单位调入用户工作区进行处理。

1·3 目标代码

连接编辑正常结束时，生成可运行和可保存的两种目标代码，前者映象于用户内存区，后者记录在编译工作文件MWG上。

目标代码包括：

(i) 过程表和共名量表

(ii) 初值数据

(iii) 各程序段的常数和信息区

(iv) 各程序段的目标指令

当用户需要保存目标代码时，可以将MWG的代码连同连接编辑后的全部现场记入用户纸带文件或磁带文件。对于复盖下的连接编辑，用户内存区常驻过程表、共各量表、初值数据以及主段的目标数据和指令，各复盖段的目标数据和指令当被调用时动态加载于内存复盖区。

连接编辑如果失败，不能形成正确的目标代码或不形成目标代码，这时输出错误信息，同时报11号事件。

1·4 连接编辑过程

第1步：打开BWG，定义並打开MWG，置连接编辑初态。

第2步：依半目标模块在BWG的次序（随源程序段进入系统的次序而定）逐块调入工作区。扫描ESD，PUB，COM，RLD，收集中间代码信息，建立相应的信息表。

第3步：按特定的次序调入半目标模块，执行确定程序调用

结构的算法，求出描述程序调用关系的树形结构。对于引用库过程的用户，从库文件PPP—LIB调入相应的库过程模块，库过程也用FORTRAN IV编写，分优化和不优化两种模块形式。对库过程模块的处理同用户半目标模块完全相同，並且要记进BWG，作为半目标模块的一部分。

第4步：确定各程序段的级次以及公用块分配时所隶属的程序段。

第5步：按调用的树形结构，从0级段（根段）开始依级次分配存贮，直至末级段（树梢段）。

第6步：按调用级次和调用次序调入半目标模块，完成外部过程访问的连接以及半目标代码地址的重定位代真。连接和代真后的段目标代码被记入编译工作文件MWG。

第7步：把目标代码分别从工作区和MWG映象于用户目标区

第8步：对PPP系统归还申请多余的内存资源，关闭並撤消BWG，关闭MWG，输出用户所占资源的信息。

§2. 表区

2·1 过程表PRT

PRT是全局表，在连接编辑过程中是工作用表，又要保留于目标代码中。每个程序段占用表的一项，最多容许48项，当用户程序超过48段时，可以修改表长单元（CPRTN）而予以扩充。

PRT项的中间形式：

K(6)	LEV(6)	NAME(36)	
TY(3)	CAT(3)	ADR(17)	DIM(8) NADH(16)
RLTH(24)		MODH(24)	
DT		0	
0		0	
DATH		DATT	

K——程序段变址保护层次

LEV——程序段调用级次

NAME——段名

TY，CAT，ADR，DIM，NADH——ESD中段名项相应信息

RLTH——程序段重定位表RLT的表项始地址

MODH——模块在BWC的始行号

DT——程序段变址区末地址＋1

DATH——程序段数据区始地址

DATT——程序段数据区末地址＋1

PRT项的最后形式：

K(6)	LEV(6)	NAME (36)
FAH (24)		FAT (24)
DATH		DATT
PH		PT
LTH		FILH
PRT1		DPT2

K，LEV，NAME，DATH，DATT——同上

FAH——程序段数组信息字区始地址

FAT——数组信息字区末地址＋1

PH——程序段指令区始地址

PT——程序段指令区末地址＋1

LTH——程序段行表区始地址

FILH——程序段目标代码记MWC的始行号

PRT1——调用本段的程序段PRT项地址（动态填入）

PRT2——本段所调用程序段的PRT项地址（动态填入）

2·2　共名量表PBT

PBT是为了在运行中读写共名量而设置的全局表，每一个共名量（共名变量或共名数组）占用表的一项，最大容许48项。当用户程序共名量超过48个时，可以通过修改表长单元（CPBTN）而予以扩充。

TY (3)	CAT (4)	NAME (41)
H (24)		LH (24)

TY——共名量类型，1整型，2实型，4复型，5逻辑型，6文字型

CAT——种属，3共名变量，7共名数组

H——共名量始地址

LH——共名量容量

2·3　块数据段表BLT

每个块数据段占用表的一项，最多容许8项，修改表长单元（CBKTN）可予以扩充。

NAME (48)

NAME——块数据段段名

2·4　公用块表CBT

每个公用块占用表的一项，最多容许48项，修改表长单元（CCBTN）可予以扩充。

L (1)	PRTN (7)	NAME (40)
H (24)		LH (24)

L——0表示不赋初值，1表示赋初值（在块数据段出现）

PRTN——PRT项的顺序号，表示在相应的程序段分配

NAME——公用块名

H——公用块始地址

LH——公用块容量

2·5　重定位表RLT

每一程序段占用表的一项，最多容许48项，当扩充PRT时RLT随着扩充。

CBAS 24	CLH 24
FABAS	F ALH
FBAS	F LH
WBAS	WLH
NADBAS	NADLH
RWBAS	R WLH
INITH	I NITLH
PBAS	P LH
TAGH	T AGLH
LTBAS	L TLH
DBAS	D LH
ZBAS	Z LH
HBA S	H LH
L1BAS	L 1LH
L2BAS	L 2LH

C

表项中左半部除去INITH，TAGH为相应的半目标模块RLD中

初值数据区和指令种属区始地址外，其余依次为常数区、数组信息字区、格式字区、优化常数区、参数地址表区、输入输出广指参数区、行表区、假变址区、临时工作单元区、赋初值一般变量和数组区以及一般变量和数组区所分配的基地址，表项右半部同相应的RLD完全相同，为各区的容量。

2·6　级次表LEVT

每个程序段占用一项，最多容许48项。当扩充PRT时LEVT随着扩充。

序号i：

L

L—PRT第i项的程序段的调用级次

2·7　公用块从属距阵M

每个公用块占用距阵的一行，最多容许48行。当扩充CBT时M的容许行数随着增加。

$$M=\left[m_{ij}\right]$$

$$m_{ij}=\begin{cases}0, & \text{CBT第 i 项的公用块不属于 PRT第 j 项的程序段。}\\ 1, & \text{CBT第 i 项的公用块属于 PRT第 j 项的程序段。}\end{cases}$$

2·8　调用布尔距阵C

描述用户程序中各程序段的调用关系的48×48布尔矩阵。当扩充PRT时，C的阶数随着扩大。

$$C=[c_{ij}]$$

$$C_{ij}=\begin{cases}0, & \text{PRT 第 j 项的程序段不被第 i 项的程序段调用，}\\ 1, & \text{PRT 第 i 项的程序段调用第 j 项的程序段。}\end{cases}$$

2·9　调用优先级矩阵 P

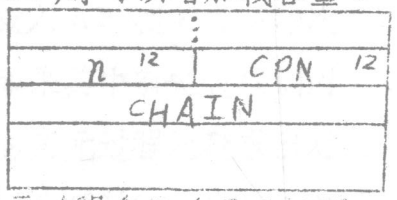
记录各程序段的优先级。当段 A 直接或间接调用段 B 时，A 称为 B 的优先级。

$$P=[p_{ij}]$$

$$p_{ij}=\begin{cases}0, & \text{PRT 第 j 项不是第 i 项的定先级，}\\ 1, & \text{PRT 第 j 项的程序段是第 i 项程序段的优先级。}\end{cases}$$

2·10　调用过程栈 CPS

在确定调用关系过程中，动态寄存依次出现的调用过程，以确定各程序段的直接调用段。每一调用占用栈的一项，最多可达96项。进出栈依先进后出次序，栈顶项被处理后随即退栈，当修改最大调用过程数（CCPSN)时可以增加栈容量。



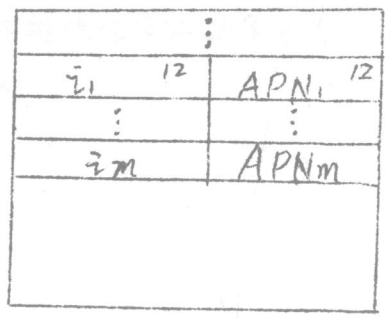
n——调用中的实元过程个数（无参过程 n＝0）

CPN——被调用段的 PRT项序号

CHAIN——APS栈（见2·11)中存调用实元过程的链头地址
（无实元过程时 CHAIN=0）

2·11　实元过程栈 APS

存放 CPS栈中所记录调用过程的实元过程。每一实元过程占用一项，最多可达192项。当修改最大调用实元过程数（CAPSN)时可以增加栈容量。APS也是先进后出栈，与 CPS同时进栈和退栈。



i₂，…，iₘ——实元过程在实元参数地址表中的序号

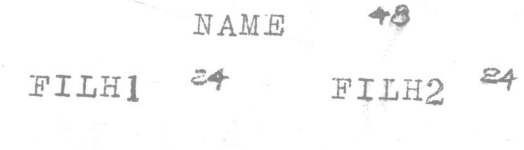
APN₂，…，APNₘ——实元过程（均为实过程）PRT项序号

2·12　实元过程寄存栈 APR

形式同 APS栈，每一实元过程占一项，共20项，APR用以寄存 APS每次退栈的内容，不是先进后出栈。

2·13　库过程表 LPRT

每个库过程占用一项，最大库过程数为64。当需要扩大库过程时，可以增加表区和相应的查表范围。



NAME——库过程名

FILH1——非优化模块在库文件 PPP—LIB中的始行

FILH2——优化模块在库文件 PPP—LIB中的始行

§3.　程序的调用结构

3·1　建立调用布尔矩阵的步骤

FORTRAN IV 允许过程作哑元，在调用中过程出现的方式有如下六种情况：

过程调用	实　元
实过程	(非过程)
实过程	实过程
实过程	哑过程
哑过程	(非过程)
哑过程	实过程
哑过程	哑过程

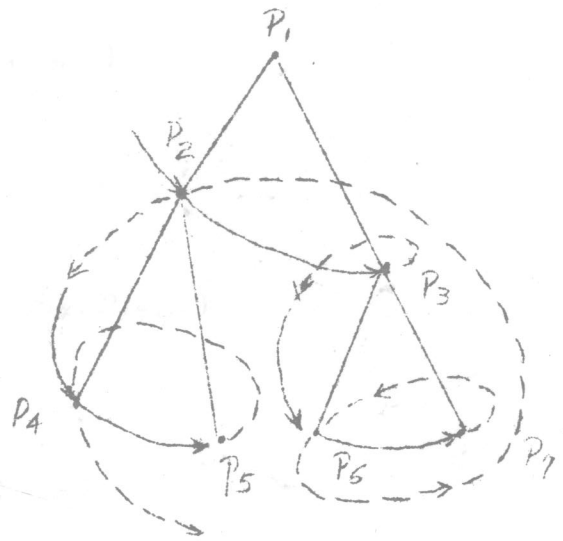
对于前三种情况，矩阵C的相应元素为1。对于后三种情况，需要确定在其他程序段调用当前段的各个调用语句中有哪些相应的实过程实元。如果其中有哑过程实元，还要再确定对应的是哪些实过程。

建立布尔矩阵C的实际步骤如下：首先确定主段调用的过程（全为实过程），对C的相应元素置1，同时把调用过程堆放入调用过程栈CPS，其对应的实元过程（全部为实过程）堆放入实元过程栈APS，主段调用的过程处理完后，取出CPS栈顶项以及APS中相应的各实元过程项，CPS和APS同时退栈。对刚取出的栈顶项采取同样的步骤将该段所调用的实过程堆放入CPS栈，其相应的实元过程堆放入APS栈，而在每次CPS和APS进栈前对C的相应元素置1。如果其中有哑过程调用，进CPS栈之前要通过同自APS栈已取出的实元过程比较求出对应的实过程。同样，如果调用中的实元过程为哑过程，在进APS栈之前，也要通过同自APS栈已取出的实元过程比较求出对应的实过程。

如果栈顶项的程序段不调用别的程序段，CPS和APS退栈后接着处理新的栈顶项。

重复以上步骤，一直到CPS栈中的过程处理完为止．设程序有程序段P_1，P_2，P_3，P_4，P_5，P_6，P_7，，其中P_1为主段，並且依次有P_1调用P_2和P_3，P_2调用P_4和P_5，P_3调用P_6和P_7，CPS和APS进栈和退栈过程如图3．1所示，实矢线表示进栈，虚矢线表示退栈。

C



在建立布尔矩阵C的过程中，同时进行有关调用的语法检查

(i) 检查调用过程同过程定义的一致性；

(ii) 检查实元与哑元的一致性。

如果程序中有库过程调用，从库文件PPP—LIB调入库过程模块，並把库过程纳入调用结构而作同样的处理。

3·2 建立布尔矩阵C的算法

用ℓ记BWG行号，e记ESD项地址，C记CPS栈地址，a记APS栈地址，i，J记PRT序号，nd记哑元参数地址表NAD地址，na记实元参数地址表NAD地址，apn记实元过程个数，an记实元个数。算法如下：

1） ℓ＝主段半目标模块记BWG始行，C=0，a=0，

2） 调入半目标模块BWG(ℓ)。i＝所及段PRT序号，e=ESD中首次调用外部过程项的地址，R=ESD末项地址。

3） e＞R转9）。否则，处理ESD(e)：如果ESD(e)非过程调用时转8)。当ESD(e)为实过程调用时，J=过程PRT

序号。当ESD(e)为哑过程调用时，j=APS退出的与之相结合的实过程PRT序号。C(i，j)=1。检查调用同过程定义的一致性。

4)如果调用是无参调用，转8)。否则，n_d=过程哑参地址表项地址，na=调用实参地址表项地址。an=apn=1。

5)检查NAD(na)同NAD(n_d)的一致性。如果NAD(na)不是实元过程，转6)。否则实元过程PRT项序号及an进APS栈，a=a+1，同时apn=apn+1，当实元过程为哑过程时，要从APS退出的实过程中求出与之相结合的实过程PRT序号。

6) an=an+1，na=na+1，n_d=n_d+1。an<实元个数，转5)；否则转7)

7) 调用过程PRT序号进及apn进CPS栈，C=C+1。

8) e=e+1，转 3)。

9) 如果C=0，转11)。否则CPS退栈，C=C−1。如果退出项是无参调用，转10)。否则，取出APR中相应的实元过程，以备在3)和5)中为确定同哑过程对应的实过程用。同时APS退栈，a=a−apn。

10) l=CPS栈中退出的顶项过程段半目标模块记BWG始行。转2).

11) 建立布尔矩阵结束。

注: 关于库过程的处理，包含在3)的求过程PRT序号的步骤中.

3·3 建立调用布尔矩阵C的例

一个可执行的FORTRAN程序的调用部分如下：

```
MASTER   P1
EXTERNAL   P5, P6, P13, P14, P15
.......
CALL   P2
```

C


```
.....
CALL P4(P5, P6 )
......
CALL P7
......
CALL P10(P13, P14, P15, X)
......
END
SUBROUTINE  P2
......
CALL   P3
......
END
SUBROUTINE  P3
.......
END
SUBROUTINE  P4(P,Q)
EXTERNAL Q
CALL P(Q)
END
SUBROUTINE  P5 (T,X)
.......
E=T(X)
.......
END
```

```
FUNCTION P6(Y)
......
P6=Y
......
END
SUBROUTINE  P7
CALL  P8
CALL  P9
......
END
SUBROUTINE  P8
......
END
SUBROUTINE  P9
......
END
SUBROUTINE  P10(U,V,W,Y)
......
EXTERNAL U,V,W
CALL P11(U,V,W,Y)
......
END
SUBROUTINE P11(Q,R,S,X)
......
EXTERNAL  Q,R,S
```



```
CALL P12(Q,R,S,X)
......
END
SUBROUTINE P12(F,G,H,Y)
......
EXTERNAL G,H
CALL F(G,H,Y)
......
END
SUBROUTINE P13 (M,N,Z)
......
E1=M(Z)
E2=N(Z)
......
END
FUNCTION  P14(C)
......
P14=C
......
END
FUNCTION P15(D)
......
P15=D
......
END
```

设：P1，P2，P3，P4，P5，P6，P7，P8，P9，P10，P11，P12，P13，P14，P15的PRT序号依次为1，2，3，4，5，6，7，8，9，10，11，12，13，14，15。

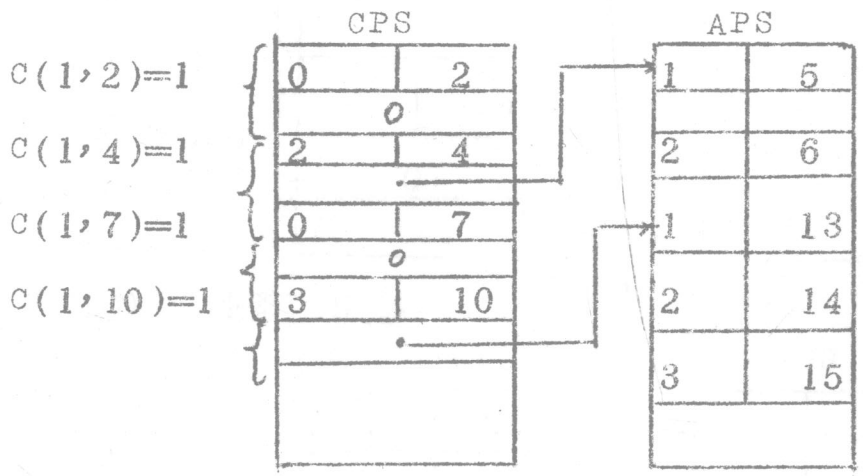
建立调用布尔矩阵C的过程中，C(i，j)以及CPS和APS的状态改变如下：

(1) 初始状态

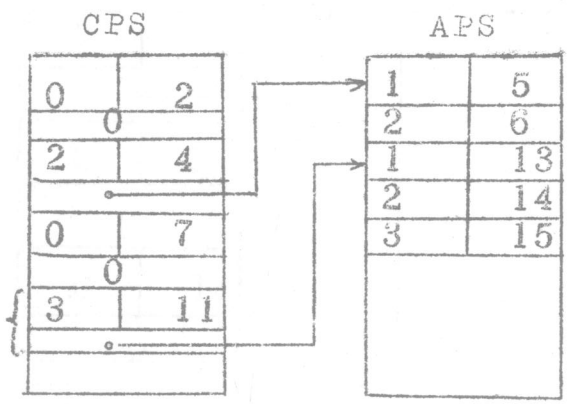
$$C = \begin{bmatrix} 0 \cdots 0 \\ \cdots\cdots \\ 0 \cdots 0 \end{bmatrix}$$

CPS，APS 空

(2) 处理主段P1

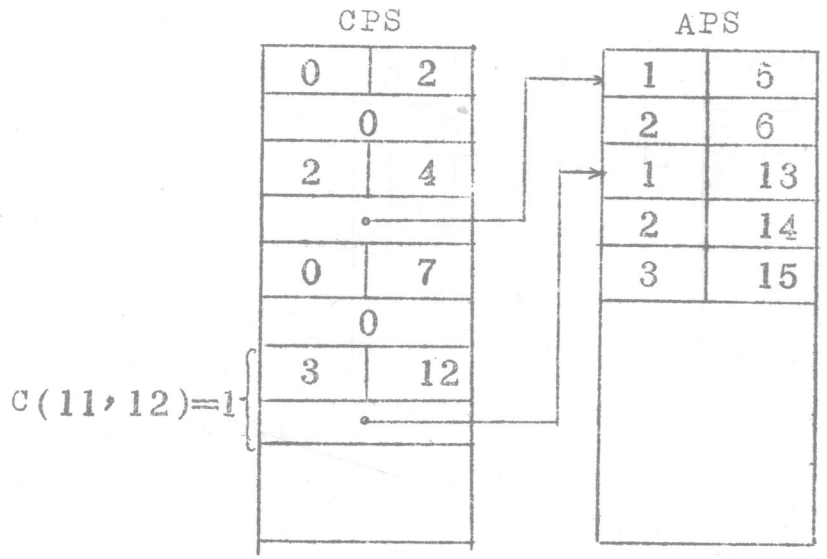


(3) 处理P10



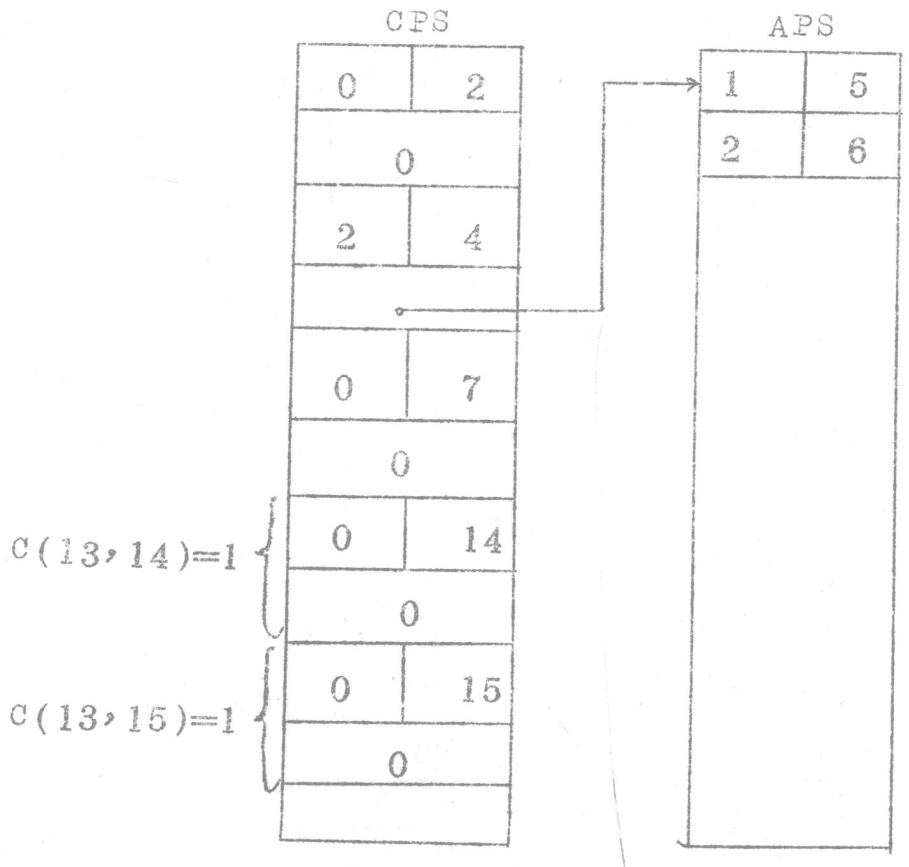
C

(4) 处理P11

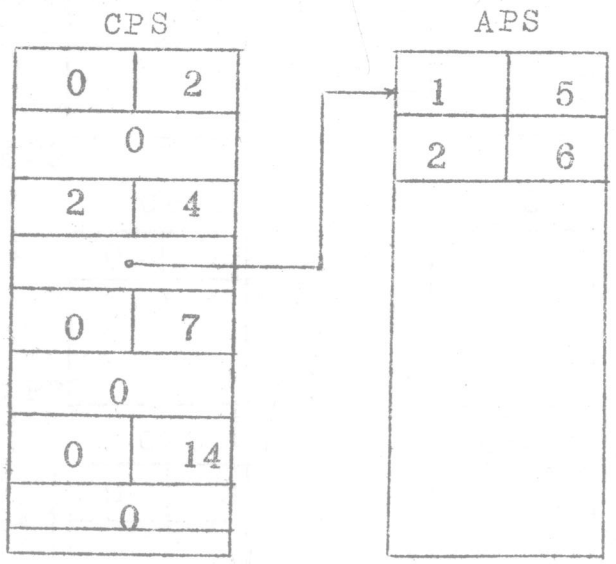


(5) 处理P12

(6) 処理P13

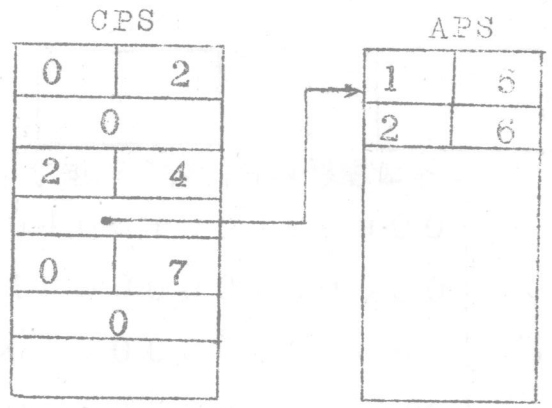


(7) 処理P15

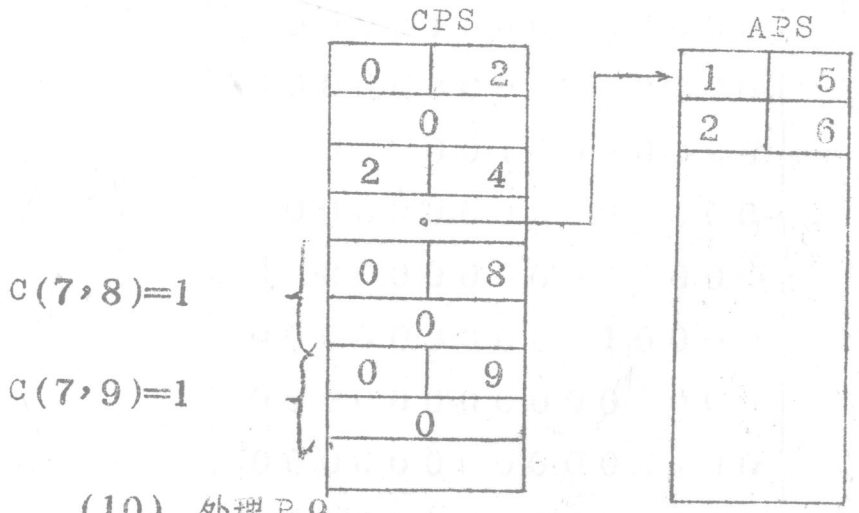


C

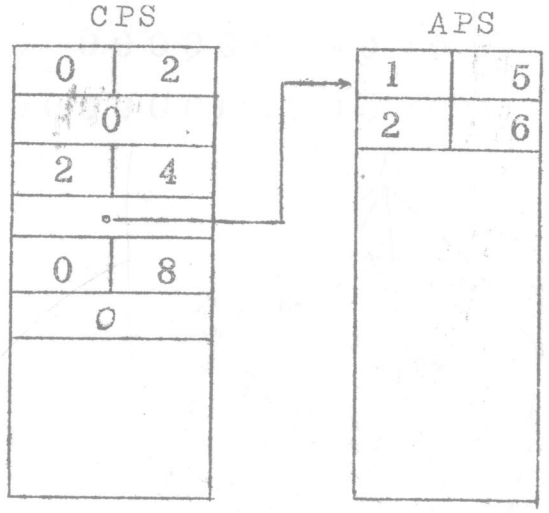
(8) 処理P14



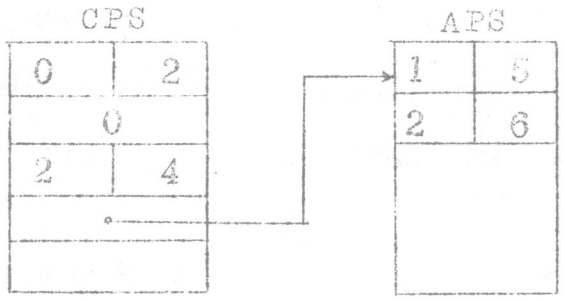
(9) 処理P7



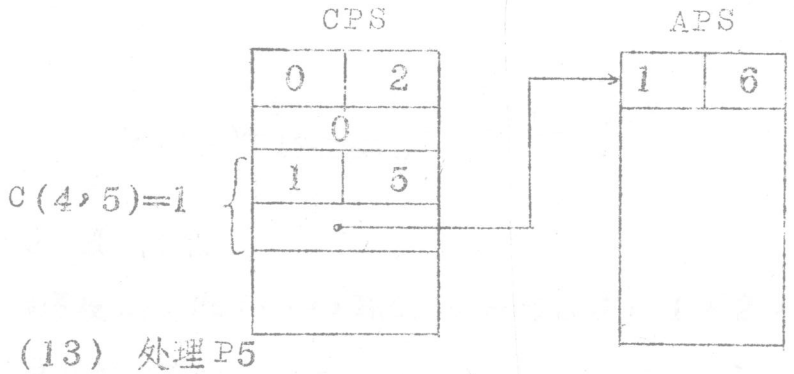
(10) 処理P9



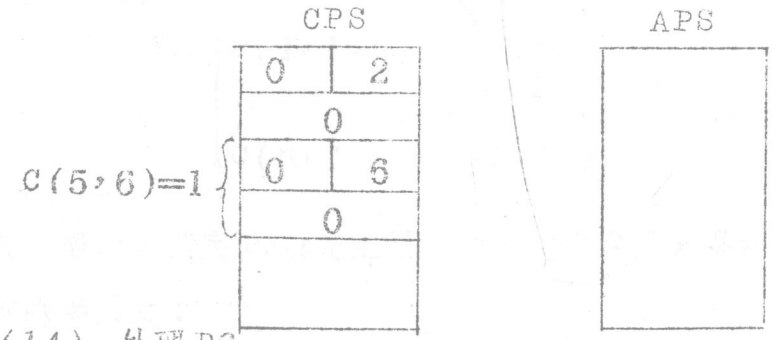
(11) 处理P8



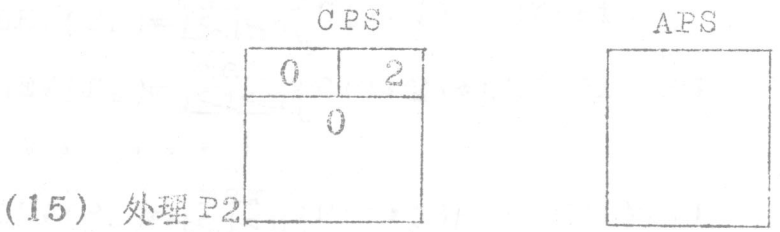
(12) 处理P4



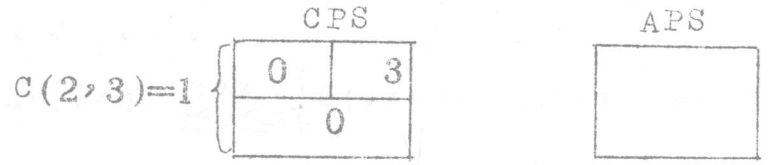
(13) 处理P5



(14) 处理P6

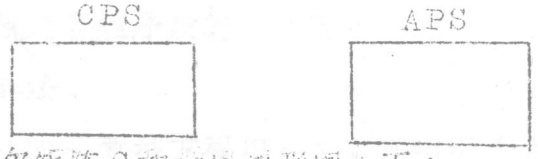


(15) 处理P2



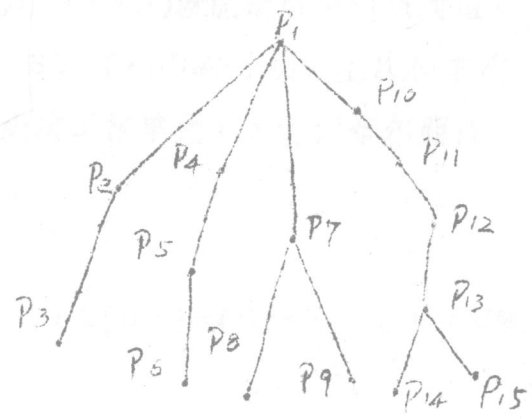
C

(16) 处理P3



最后得到调用布尔矩阵C和调用树形图如下：

$$C=\begin{bmatrix}
0&1&0&1&0&0&1&0&0&1&0&0&0&0&0\\
0&0&1&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&1&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&1&1&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&1&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&1&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&1&1\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0
\end{bmatrix}$$



§4． 调用级次

4·1 调用级次的定义

在调用关系中，调用段称为被调用段的直接优先级段。

设程序由程序段 P_1，P_2，···，P_n组成，我们把程序段 Pi 的级次定义如下：

（ （1） 设 Pk为主段，级次 LEV(Pk)==0

（11） 设 Pi段的直接优先级段为 Pi1，···，Pim，则级次

$$LEV(Pi)=\max_{1\le j\le m}\{LEV(Pij)\}+1。$$

4·2 级次方程

设程序段 P_1，P_2，···，P_n的PRT序号依次是 1，2，···，n，调用布尔矩阵为

$$C=\begin{bmatrix} C(1,1)\cdots & C(1,n) \\ \vdots & \vdots \\ C(n,1)\cdots & C(n,n) \end{bmatrix}$$

不失一般性，例定 P_1 段是主段。由级次的定义，得到下述表示级次的关系式：

$$\begin{cases} LEV(P_1)=\max\limits_{1\le j\le n}\{C(i,1)\cdot LEV(Pi)\} \\ LEV(P_2)=\max\limits_{1\le i\le n}\{C(i,2)\cdot LEV(Pi)\}+1 \\ \cdots\cdots\cdots \\ LEV(Pj)=\max\limits_{1\le i\le n}\{C(i,j)\cdot LEV(Pi)\}+1 \\ \cdots\cdots\cdots \\ LEV(Pn)=\max\limits_{1\le i\le n}\{C(i,n)\cdot LEV(Pi)\}+1 \end{cases}$$

C

我们可以把这组关系式看成级次所应满足的一组方程，称为级次方程。级次方程的解就是各程序段的级次值。

4·3 计算级次的算法

级次方程是一非线性的离散型方程组。可以证明级次方程的解是存在的。

由于级次方程具有迭代的形式，试用简单迭代法求解。设 $L^{(0)}=(l_1^{(0)},\cdots,l_n^{(0)})$是一组初值，其中 l_1，···，l_n是非负整数，按公式

$$L^{(k+1)}=(l_1^{(k+1)},\cdots,l_n^{(k+1)})=(\max_{1\le i\le n}C(i,1)\cdot l_i^{(k)},\cdots,\max_{1\le i\le n}C(i,n)\cdot l_i^{(k)})+(0,1,\cdots,1)$$

求各次近似解。

根据 FORTRAN程序的结构，在每次迭代过程中，凡已经被确定级次的段，其级次不再改变，又由迭代公式可如，经过第一次迭代，主段的级次被确定（初值）。以后每迭代一级，总有一些段的级次被确定，由于程序段的数目是有限的，在有限步迭代之后，必定求出所有段的级次值。事实上，迭代步数正好是最大级次值。这样得到的解，显然满足级次方程，但同级次定义可以差一常数。

具体求解时，取 L(0)=(0，···，0)就能最后得到级次值。设 L(n)是存级次值的数组，SUM1，SUM2分别存上次和本次级次迭代值总和，LEV为求级次工作单元，i，j记布尔矩阵 C 的行与列数。

算法如下：

1） 置初值 L=(L(1)，···，L(n))=(0，···，0)，SUM2=0

2） SUM1=SUM2， SUM2=0， j=1。

3） LEV=0， i=1。

4） 如果j=主枝PRT序号，转5）；否则，转8)。

5） 如果C（i，j)=0，转6)；否则LEV=max{LEV，L(i)}

6） i=i+1，如果i>n，转7)；否则，转5)。

7） L(j)=LEV+1，SUM2=SUM2+L(j)。

8） 如果j<n， j=j+1，转3)；否则

9） 如果SUM1<SUM2，转2)；否则

10） 迭代结束，L(1)，…，L(n)为求出的相应段的级次值。

4·4 调用的语法检查

在求级次的算法中，还可以加进有关调用的语法检查：

(1)如果 $\sum_{i=1}^{n} C(i,j)=\sum_{k=1}^{n} C(j,k)=0$，表示布尔矩阵C中的j行j列全为0，即PRT序号为j的程序段不被调用。

(2)如果在迭代中出现某个L（i）≥n，表示在调用路径上存在直接或间接递归调用，迭代是不收敛的。

§5· 存贮分配

5·1 内存总体布局

数存布局如图5·1指存布局如图5·2。

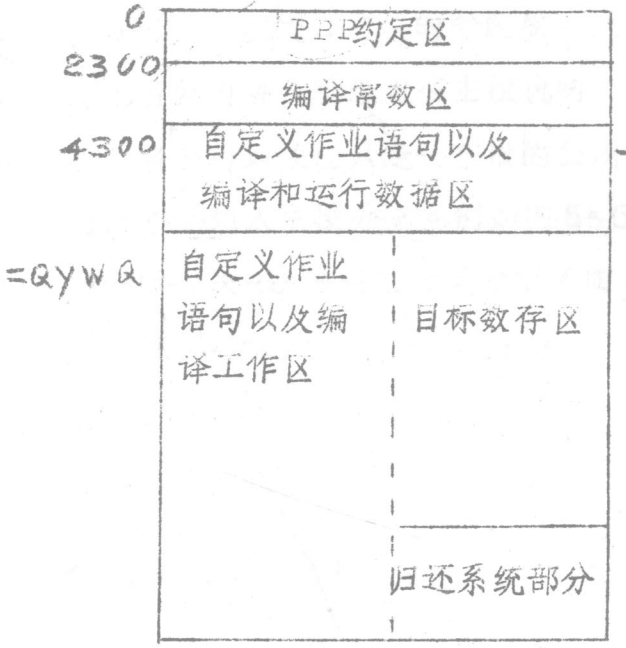


图5·1

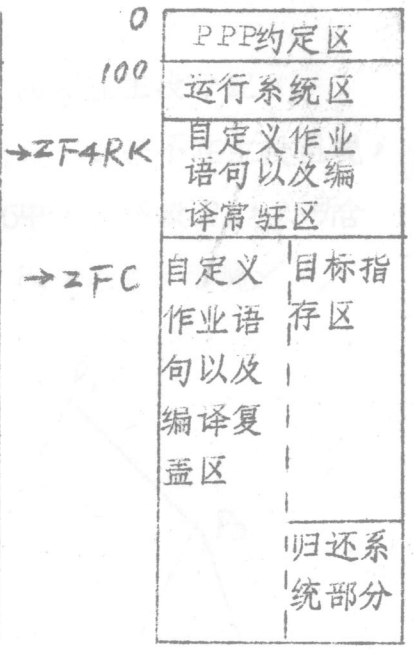


图5·2

5·2 分配方案

(1) 目标数存区

全局表PRT，PBT，程序段数据区（段常数区C，数组信息字区FA，格式信息区F，优化常数区W，参数地址表区NAD，输入输出广指参数区RW，行表区LT），共名量区以及初值数据区常驻内存，从目标数存区起始地址开始分配。

临时工作单元区，假变址区以及一般变量和一般数组区作为局部量依调用的树形结构按级次复盖分配。

公用块量作为局部公用量处理，每一公用块附属于一程序段，与该段局部量一起分配。

最后分配变址保护区。

目标数存分配方案如图5·3。

(2) 目标指存区

按调用的树形结构依级次顺序连接分配。

在各段目标之后，分配为解释执行带假变址指令而形成的附加指令区。

目标指存分配方案如图5·4。

(3)、变址分配

变址问数存局部量一样，依调用结构按级次复盖分配。但是，因为只有54个可用变址，在同级次段的变址分配中，当出现末变址超过54时，该级次各段变址从基变址（δ_{12}）开始分配。

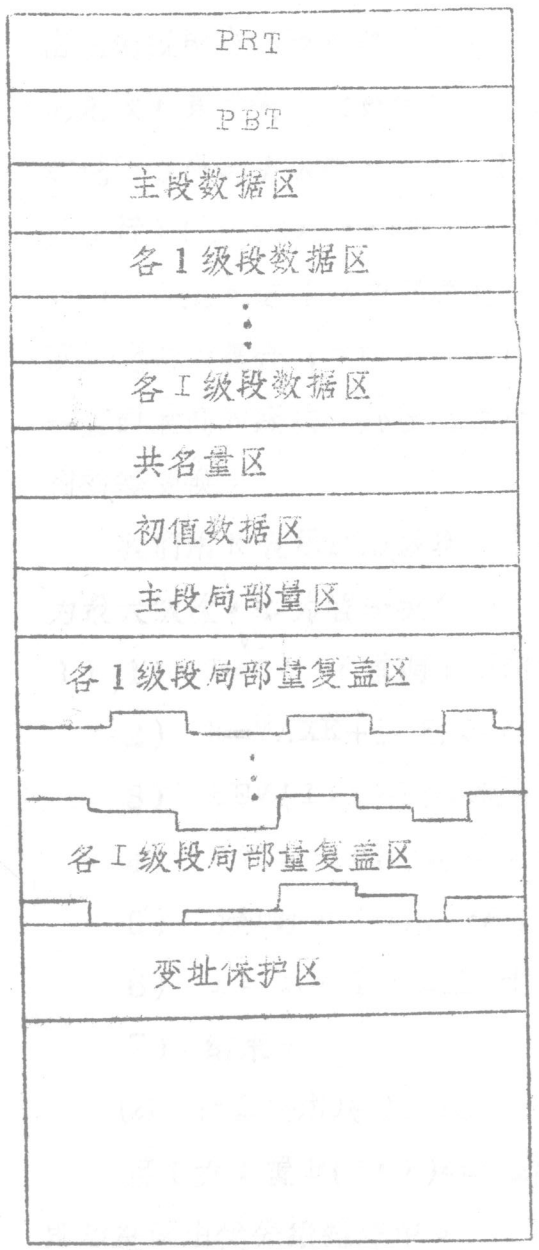


图5·3 目标数存

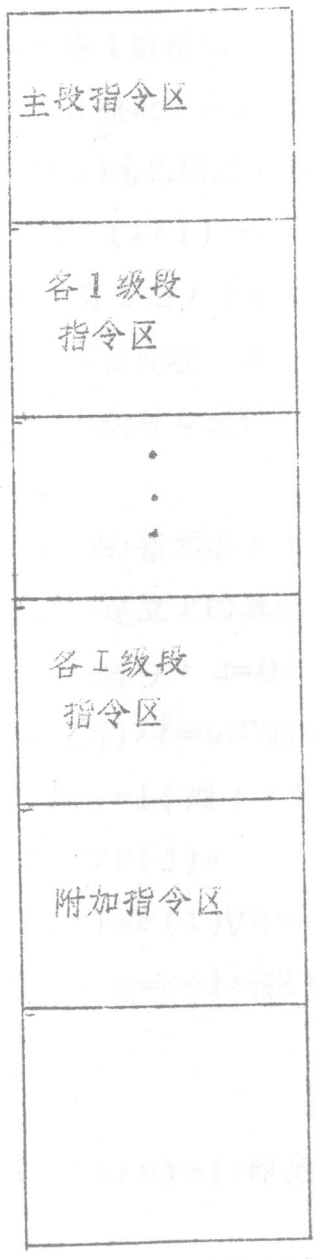


图5·4 目标指存

5·3 确定公用块的分配段

无名公用块一般总是在主段说明，固定在主段分配。

有名公用块往往只是一些段的公用量，如果不在主段出现，有可能不必列入主段分配。例如图5·5中，程序段P_2，P_3所含的公用块/X/必须在主段P_1分配，而P_4，P_5的公用块/Y/则可以在P_3分配，

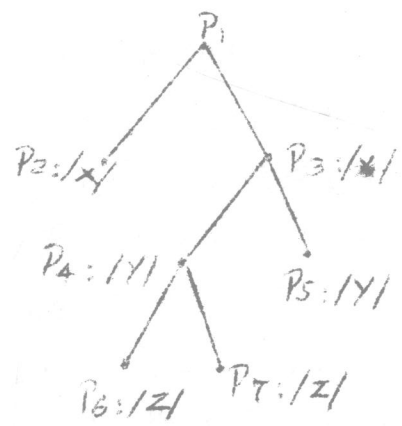


图5·5

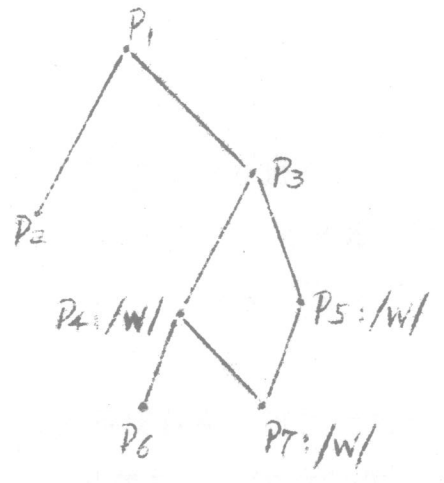


图 5·6

P_6，P_7的公用块/Z/可以在P_4分配，但是图5·6中的/W/块可以在P_3分配，也可以在P_7分配。我们采用后一种分配。

一般地，如果某公用块出现在程序段P_{i1}，···，Pim中，我们分别找到直接或间接调用每一段的程序段——优先级段的集合，同时把各段自身也加进集合内，然后取这m个集合的交集中最大单级次者，作为该公用块分配存贮的程序段。

为此，第一步首先建立调用优先级矩阵P。第二步，通过公用块从属矩阵M求公用块分配段，并把段的序号填进公用块表CBT中。

(1) 建立调用优先级矩阵P

在调用布尔矩阵C中，如果C(i，j)=1，表示程序段i（PRT项序号）调用j，因此，取P=C就得到优先级矩阵的直接优先级部分。接着，按级次顺序，从0级段（主段）开始，逐级逐段求出优先级段的集合。事实上，主段无优先级，各1级段以主段为优先级，其优先级段集合已经求出。设i段为2级段，而且P(i，j)=1，这说明j段是1级段，P(j，1)，…，P(j，n)既已确定，只需把j行P(j，1)，…，P(j，n)逻辑加入i行P(i，1)，…，P(i，n)上。这样处理完满足P(i，j)=1的所有j后，就得到i段的优先级集合 $\{j \mid P(i,j)=1\}$。2级段处理后处理 3级段，一直到最高的级次段处理完为止。这一过程实际是对C进行一系列初等变换。

我们用k表示调用级次，I表示级次表LEV的指示器，MAXK为最大级次，n为程序段个数，Q为工作单元。建立P的算法如下：

1) 1) P=C' （求调用布尔矩阵C的转置矩阵），k=0。

2) k=MAXK+1 时转7）；否则i=1(行)，l=LEV始地址。

3) LEV(l+i-1)≠k时转6）；否则，j=1（列），Q=0

4) P(i，j)≠1时转5）；否则，Q=Q∨P(j)。

5) j≠n时，j=j+1，转4)；否则，P(i)=P(i)∨Q。

6) i=i+1。i≤n时转3）；否则，k=k+1，转2)。

7) 结束。

(2) 计算公用块分配段

第1步：置P(1，1)=P(2，2)=…=P(n，n)=1，即把各段加进调用优先级矩阵中。

第2步：从CBT中取出一项，例如S。设S从属于程序段P_{S_1}，

…，Psm。令 $R_1=\{Pj \mid P(S_1,j)=1\}$（$P_{S_1}$段的优先级段的集合），

……

$Rm=\{Pj \mid P(sm,j)=1\}$（Psm段的优先级段的集合）。

取 $R=\bigcap_{k=1}^{m} Rk$。

如果 $Pi=\{Pi \in R, LEV(Pj)\neq LEV(Pi), LEV(Pj)<LEV(Pi), j\neq i\}$

则S在Pi段分配

重复执行第2步，直到处理完CBT的所有项为止。

设k=CBT项指示器，T=CBT自由项地址，L，Q为工作单元，n为程序段个数，MAXL为最大级次，I=LEV指示器。算法如下：

1) P=P+E，其中 $E=\begin{bmatrix} 1 & & \\ & \ddots & \\ & & 1 \end{bmatrix}$（把各段自身加进优先级矩阵）

2) k=CBT项始地址，i=1（公用块从属矩阵M的行计数）。

3) 当k=T时（公用块处理完），转16)；否则

4) 如果CBT(k)为无名块（固定在主段分配），转15)；否则

5) 如果CBT(k)出现在块数据段中（赋初值，不作局部公用量分配），转15)；否则 j=1（M列计数），Q=全1'

6) 如果M(i，j)≠1，转15)；否则

Q=Q∧P(j)（求i段的优先级段的交集）。

7) 如果j=n（Q中已求出i段各优先级段的交集），转8)；否则，j=j+1，转6)。

8) （求Q中最大的单级次段） L=MAXL，l=LEV始地址。

9) i=0（最大级次计数），j=1（Q单元字位计数）。

10) 如果 $bit_j(Q)\neq 1$，转12)；否则