基于数组元素视图的并行化和特征*)

Parallelization and Parallelism Characteristic Based on Array Element Graph

曾国荪
(同济大学计算机科学及工程系　上海200092)
陆鑫达
(上海交通大学计算机科学及工程系　上海200030)

Abstract In order to realize heterogeneous computing, one must firstly find out the parallelism characteristic for a segment of program. But the existing theory of parallel processing is not sophisticated, and prevents parallel computing from becoming the mainstream in computer applications. Therefore, it is necessary to achieve breakthrough in schemes of research. In this paper, a novel approach to data dependence analysis, based array element graph, is proposed. By means of graph theory and algebra abstract, the authors infer and prove what parallelism characteristic types of single nested loop program have The paper also establishes a foundation for further research in such area.

Keywords Heterogeneous computing, Loop program, Array element graph, Parallelism characteristic

1 引言

近年高性能并行计算研究领域已转到异构计算[1,2],因为一个复杂的计算问题,如 Grand challenge 问题[3],往往存在多种计算需求,任何单一的计算模式,如:SISD,SIMD,MIMD,数据流等,都不能使求解过程顺利进行,只有按需分配计算模式,才能取得计算性能最优。异构计算的关键是并行性特征的提取。我们已经提出了基于程序结构和基于程序运行性能分析的两种方法[4~6],但还不完善。本文希望通过数组元素视图作深入的研究。

并行处理是异构计算研究的基础,数据相关性分析则是并行处理研究的出发点。多年来,并行化研究的对象集中在循环程序,因为循环程序耗时,且隐藏着丰富的并行性,对其并行化最有意义。透视一段循环程序,不外乎由语句、数组变量、数组元素等组成。目前,存在"基于语句"和"基于数组变量"的两种并行化方法,较为成熟。例如:以语句为结点的方法,构造相关性图,如果存在相关回路,则该循环程序不可并行执行,反之可并行执行。以数组变量为对象的分析方法,建立在解析公式的基础上,具有牢固的数学理论根基,已被广泛采用,如:各种循环变换技术(交换、反序、斜扭)都有相应的转换公式,最为著名的是数论丢番图(Diophantos)方程整数解判据的应用。但是,我们认为:基于数组变量的分析方法,研究对象的粒度仍然较粗,可能忽略某些性质,因为数组变量是所有同类性质的数组元素的总称,数组元素是数组变量取某一循环控制变量值时的实例。可见,数组元素是循环程序并行化研究的最小单位。从现代物理学研究发展史我们可以得到启示:物质是由分子、原子、中子、量子组成,了解物质基本性质时,可通过分子、原子,但探究物质内部结构和规律时,只能靠量子,甚至更小的微粒。所以,采用数组元素视图,是唯一有能力探索和识别出循环程序中纷繁紊乱相关性的方法,有望开发出循环体中最大并行性,以及冲击具有控制结构的并行化技术。

2 数组元素数据相关性视图

定义1 循环程序数组元素相关性视图是一个三元组 $G=(N,E,L)$,其中 N 为结点集;$E\subset N\times N$ 为边集,表示数组元素之间的关系,用一有向线条连接。$L\in\{0,1,2,\cdots,\}$,为图的层次编号,代表数组元素定值的时刻,即繁衍诞生的年轮辈分[7,8]。

可见,上述元素视图是时序层次图,最为关心的结点是引用、定值,再生数组元素。假设某循环体中,有一赋值语句:C(3)=A(1)*B(2),称 C(3)有一次定值性出现,A(1)和 B(2)各有一次引用性出现。同时称 A(1)和 B(2)结合生成了 C(3),A(1)和 B(2)是 C(3)的父母,C(3)是 A(1)和 B(2)是子女。如果其后还有一条赋

*)本课题得到国家自然科学基金资助(69773014)。曾国荪　博士、副教授,主要研究领域是并行处理,异构计算。陆鑫达　教授,博导,主要研究领域是并行处理,系统结构,指令级优化编译。

值语句:C(3)=D(4)+E(5),称第二个 C(3)是第一个 C(3)的再生。同一个元素的子女与再生具有固定的时序:子女居于再生之前。也就是说,一个元素总是被先引用计算出自己的子女,然后才会被新值代替形成自己的再生。由上分析可知,数组元素之间的关系为:c(child)链表示父子,即引用关系;r(re-born)链表示再生关系;m(married)链表示结合关系;s(same)链表示同体关系;y(younger)链表示同辈年幼关系。

在沿着串行运算的语义追踪循环计值的过程中,产生的各个元素构成一个序列,类似人类繁衍的家谱结构,第一个定值的元素,看作第一辈(或第一层)的第一个人;接踵而来的,不以任何定值元素为父母的定值元素均与他同辈,一个元素,他的父母中的最低辈分若为 n,则他最高辈分为 n+1,任何一个元素不得超过其前身及前身子女的辈分,在同辈元素之间,依据时序确定兄弟关系。还未在循环中定值(包括循环根本不给他定值)就先被引用的元素,引用的是他的初始值,把他看作原始人,统记作第零辈,排第一辈元素上方。在原始人跟他的第一次定值出现之间,以及同一人两次相继的定值性出现之间,用 c 链和 r 链双向链接。同一人的几个父母之间,用 m 链相接。一个人若同时为几个人的父母,为了使其子女能够形成各自父母的结合链,则要将此人复制,用 s 链相接。这样一来,在元素的出现队列中,存在着由各种链条形成的繁衍层次。上述的繁衍规则构成了数组元素视图分析方法的公理系统。

举例来说,设有循环程序:

```
for(index=2;index<=1000;index++)
    A(index)=(A(index-1)+A(index+1))/2;
```

那么其元素视图如图1所示。若将上述程序中的 index++修改成(index++)++,则元素视图变成图2所示。

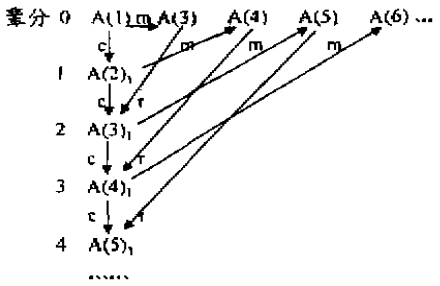

图1 循环的时序层次

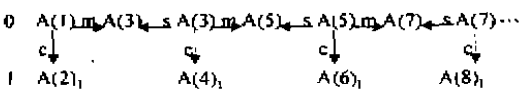

图2 循环修改后的时序层次

3 并行化判据、等价变换和数学抽象

3.1 循环并行化判据

数组元素时序层次图提供了任意两个数组变量之间的所有可能的依赖类型,而且以图的形式直观地呈现出来,这为深入研究循环并行化提供了手段。由于多重循环的内层循环一定是一重循环,因此我们以一重循环的一般形式(如图3)来讨论,其结论可扩展到多重循环。

```
FOR(i=I_L,i<I_U,i++)
{
    A_1,0(i_1,0)=f_1(A_1,1(i_1,1),A_1,2(i_1,2),…,A_1,m1(i_1,m1))
    A_2,0(i_2,0)=f_2(A_2,1(i_2,1),A_2,2(i_2,2),…,A_2,m2(i_2,m2))
    ……
    A_m,0(i_m,0)=f_m(A_m,1(i_m,1),A_m,2(i_m,2),…,A_m,nm(i_m,nm))
}
```

图3 一重循环的一般形式

这里的下标表达式 $i_{s,j}$($1\leqslant i\leqslant m$,$1\leqslant j\leqslant n_s$)为 i 的线性函数:$c_{s,j}*i+d_{s,j}$;循环控制变量的始值是 I_L 和终值是 I_U,以及下标系数均取整数,$A_{s,t}$($1\leqslant s\leqslant m$,$1\leqslant j\leqslant n_s$)可重名,$f_s$ 是仅以数组元素为自变量的函数,不含语句函数和外部函数在其中,那么,下面定理成立。

定理1 对于图3所示的循环程序,任给一个数组元素时序层次图,如果每条语句的所有定值结点均连片在同一辈分之上,称该时序层次图达到标准形,则该循环程序可并行执行。

证明:因为每一语句的所有定值结点均位于同一辈分上,说明它们之间无关联,不存在任何父子和再生关系,且可在同一时刻计算其值,即每一语句可并行执行,则整个循环程序可并行执行。 □

例如:图1对应循环不能并行执行,但图2对应的程序能够并行执行。

3.2 时序层次图的等价变换

定理1指出:只要元素视图具有标准形,则循环可并行执行。可是,一个原始时序层次图一般不具有标准形,并行化是否不得而知。然而,并行化的实质是一种改组语句运行次序的程序变换,它的前提是变换前后程序语义等价。对于数组元素时序层次图来说,图变换要求保留任意两条语句的任意两个数组变量之间的所有可能的依赖类型。由此,我们给出下面图变换规则:(1)孤立的奇异结点可随意调动;(2)没有其它依赖关系的同层上的兄弟结点可互换位置;(3)由 c 链相接起来的两次计值不能对调;(4)由 r 链相接起来的两次计值不能对调;(5)由 c-r 关联的两结点,任意定值的子女,可与该定值的再生对调,但需暂存其原值,令其后的子女去引用此暂存的值以代替原值本身。

第(5)条规则涉及到三个结点:父、子、再生,构成

一种典型的依赖关系，称之为 c-r 依赖对，它是时序层次图进行变换时主要搜索和操作的对象。

```
FOR(i=1;i<=1M;i++)
{  A(i)=B(i)*C(i);
   C(i)=B(i-1);
   B(i)=A(i+1)*D;
}
```

图4　一个循环程序

假设一个循环程序如图4所示，它的时序层次如图5所示。图5的原始状态并非标准形，除 $C(1)_2$ 之外，第二语句的结点全部在第二辈分上，欲使 $C(1)_2$ 挪入此片，它得与 $B(1)_3$，$A(2)_1$，$B(2)_3$，$A(3)_1$…调换次序。因为只有 $A(1)_1$ 与它构成 c-r 型依赖对，所以上述变动可随意进行。剩下第一语句和第三语句的结点间隔于第一辈分，让我们来分离两者。若令第一语句左靠，则需对调 $B(1)_1$ 和 $A(2)_1$，r 链所指第一语句的左部量是 A 需暂存；若第一条语句右靠，则需对调 $A(1)_1$ 和 $B(1)_3$；r 链所指第三语句的左部量是 B 需暂存。前者的目标语序为：(1,3,2)，后者为：(3,1,2)，目标执行程序为：

```
!#A=A;
!A(1:N)=B(1:N)*C(1:N);
!B(1:N)#A(2:N+1)*D;
!C(1:N)B(0:N-1);
!I=N+1;
```

说明：上段程序中，例如 !A(1:N)=B(1:N)*C(1:N) 表示 A(1) 至 A(N) 共 N 个元素并行计值。#A=A 表示暂存 A(1)，以备后引用。

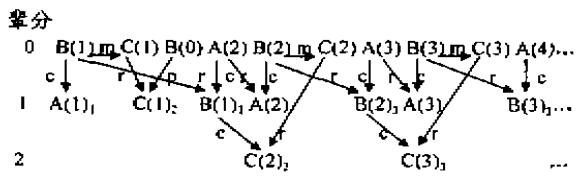

图5　一个时序层次图

3.3　时序层次图的数学抽象

对时序层次图实施等价变换，通常可推导出许多乃至无穷多个时序层次图，其图形表示直接和直观，但过于繁琐且不可穷尽，不宜计算机自动变换和处理。因此，必须提炼出时序层次图的本质特征。

任意一个集合上的任意一种等价关系，都给出该集合上的一个划分，划分的结果是原集合幂集的这么一个子集，其成员互不相交，所有成员的并恰为原集，这样的成员被称为等价类。全体时序层次图的集合也不例外，任给一个时序层次图，它在等价变换下的全部轨迹构成一个等价类。前一节中，我们论述过一个时序层次图中，结点间可能存在父子关系、再生关系、结合关系、兄弟关系，分别表示为 c 链、r 链、m 链、y 链。兄弟关系是串行语义的本质体现，然而并行执行正是对它的忽略。父子关系和再生关系构成并行处理的本质约束，二者必取其一。至此，我们归纳出时序层次图等价类的三大本质特征：结点集 N，父子关系集 C，再生关系集 R，即三元组 (N,C,R)。例如图1中，C={(A(1),A(2)$_1$),(A(3),A(2)$_1$),(A(2)$_1$,A(3)$_1$),(A(4),A(3)$_1$),(A(3)$_1$,A(4)$_1$),(A(5),A(4)$_1$)…}，R={(A(3),A(3)$_1$),(A(4),A(4)$_1$),(A(5),A(5)$_1$),(A(6),A(6)$_1$)…}。

对于每个结点 node，我们也提炼出四元组表达式，即数组名 name，下标值 subscript，语句号 sn，辈分数 gn，node=(name,subscript,sn,gn)，特别对于零辈分上的原始结点，不隶属任何语句，它们的语句号取无定义值0。因此，对于图1所示例子，N={(A,1,0,0),(A,3,0,0),(A,4,0,0)(A,5,0,0)(A,6,0,0),…,(A,2,1,1),(A,3,1,2),(A,4,1,3),(A,5,1,4),…}。结点的四元组形式，很容易用计算机程序设计语言中的结构（或记录）来实现。我们分别用 node.name，node.subscript，node.sn，node.gn 表示结点 node 的数组名，下标名，语句号，辈分数。那么，可得到下面一些形式化的结论，

时序层次图的计算机生成算法

```
输入：图3所示的循环程序 L
输出：N,C,R
算法：
N={原始入结点集};C=φ;R=φ;
FOR(i=IL;i<=IU;i++)
{
  依次扫描循环体每条语句 k: Ak,0(ik,0)=fk(Ak,1(ik,1),…,
      Ak,2(ik,2),…,Ak,mk(ik,mk))
      令 gnk=1+max(Ak,1.gn1,Ak,2.gn2,…,Ak,mk.gnmk)
      则 N=N∪{(Ak,0,Ik,k,gnk)};
      IF (Ak,1(ik,1),Ak,0(ik,0))∉C 或 (Ak,2(ik,2),Ak,0
            (ik,0))∉C 或…(Ak,mk(ik,mk),Ak,0(ik,0))∉C
            C=C∪(Ak,1(ik,1),Ak,0(ik,0))或…C=C∪
            (Ak,mk(ik,mk),Ak,0(ik,0));
      IF ∃ (Ak,0,Ik,j,gnj)∈N
            R=R∪{(Ak,0,Ik,k,gnk),(Ak,0,Ik,j,gnj)};
}
```

可见，N 集中的所有结点构成一系列，结点中的辈分信息反映了长幼关系。即便两结点辈分相同，排在前面的先执行，说明前者是兄、后者是弟，所以不需要引入专门的表示来抽象兄弟关系。当得到了 N,C,R 集后，计算机可以自动绘制出时序层次图。

推论1　对于图3所示循环 L，时序层次图结点集 N，循环体中语句总数为 SN，且 SN<I_U-I_L，那么 L 并行执行的条件是：∀ node∈N，node.gn≤SN。

证明：（反证法），如果 L 并行执行，但∃ node∈N 且 node.gn>SN，则至少有一条语句的定值性结点位于两个辈分上，这与定理1矛盾，得证。

3.4　循环包含关系

用上面讨论的时序层次图和时序层次图等价类为工具，可以进一步研究两个循环程序建立在内部数据相关依赖基础上的包含关系。

定义2 假设两个如图3所示的循环 L_s 和 L_m，对应时序层次图等价类分别为：$E_s=\langle N_s, C_s, R_s\rangle$，$E_m=\langle N_m, C_m, R_m\rangle$，若 $N_s\subset N_m$，$R_s\subset R_m+R_m^2+\cdots+R_m^n+\cdots=R_m^+$，$C_s\subset C_m+R_mC_m+R_m^2C_m+\cdots+R_m^nC_m+\cdots=R_m^*C_m$，则称循环 L_m 包含循环 L_s，记为 $L_s\subset L_m$。

显然，循环间的包含关系如同集合的包含关系，具有自反性，反对称性和传递性。

定理2 任给具有包含关系的循环 L_s 和 L_m，$L_s\subset L_m$，如果 L_m 的时序层次图能够等价变换达到标准形，则 L_s 亦然；如果 L_s 的时序层次图不能达到标准形，则 L_m 亦然。

证明：设 L_s 和 L_m 的时序层次图分别记为 H_s 和 H_m，对应的等价类抽象为 $E_s\langle N_s, C_s, R_s\rangle$ 和 $E_m=\langle E_m, C_m, R_m\rangle$，因为 $L_s\subset L_m$，根据定义有：$N_s\subset N_m$，$L_s\subset L_m$，$R_s\subset R_m^+$，$C_s\subset R_m^*C_m$，对于任意结点 node 来说，其四元组中的辈分记为 node.gn，据此，第 i 辈人的集合 G_i 抽象成：$G_i=\{\text{node}\mid \text{node}\in N\wedge \text{node.gn}=i\}$。$L_s$ 和 L_m 第 i 辈人的集合分别表示为 $G_i(L_s)=\{\text{node}\mid \text{node}\in N_s\wedge \text{node.gn}=i\}$，$G_i(L_m)=\{\text{node}\mid \text{node}\in N_m\wedge \text{node.gn}=i\}$。由于 $N_s\subset N_m$，显然 $G_i(L_s)\subset G_i(L_m)$，从而推知 L_s 的时序层次图的各个辈分分别是 L_m 相应辈分的子集。

下面在来证明两者构造上的一致性。根据 $R_s\subset R_m^+$，因此有：

$$(\forall\alpha,\beta\in N_s)R_s(\alpha,\beta)\Rightarrow(\alpha\in N_m)\wedge(\beta\in N_m)\wedge(\exists n>0, R_m^nC_m(\alpha,\beta))$$
$$\Rightarrow\exists\gamma_1,\gamma_2,\cdots,\gamma_{n-1}\in N_m, R_m(\alpha,\gamma_1)\wedge R_m(\gamma_1,\gamma_2)\wedge\cdots\wedge R_m(\gamma_{n-1},\beta)$$

一方面，站在 α 的立场上观察，L_s 时序层次图中任一结点的再生，必定是它在 L_m 时序层次图中的再生，或再生的再生……，另一方面，站在 β 的立场上看，L_s 的时序层次图中任一结点的前身，必定是它在 L_m 时序层次图中的前身，或前身的前身……。这就证明了 L_s 时序层次图中各根纯 r 链条，在 R_s 关系下与在 R_m 关系下的排序是一致的，差异仅在于可能少掉了几个结点而已。又因为 $C_s\subset(R_m^+)C_m$，则有：

$$(\forall\alpha,\beta\in N_s)C_s(\alpha,\beta)\Rightarrow(\alpha\in N_m)\wedge(\beta\in N_m)\wedge(\exists n\geqslant 0, R_m^nC_m(\alpha,\beta))$$
$$\Rightarrow\exists\gamma_1,\gamma_2,\cdots,\gamma_n\in N_m, R_m(\alpha,\gamma_1)\wedge R_m(\gamma_1,\gamma_2)\wedge\cdots\wedge R_m(\gamma_{n-1},\gamma_n)C_m(\gamma_n,\beta)$$

同理可证明 H_s 中各根 c-r 链条在 R_s-C_s 关系链下，与 H_m 中各根 c-r 链条在 R_m-C_m 关系链下排序是一致的，差异仍就在于少掉了几个结点而已。

至此已经证明了 H_s 不过是 H_m 去掉若干结点而来。L_s 不过是 L_m 跳过某些语句，或跳过某些语句的某次执行而来，H_s 中的结点可以逐个重叠于 H_m 对应的节点上。因此，对 H_m 图进行等价变换时，涉及到 H_s 图中结点的话，该等价变换可同步在 H_s 上进行。

综上，若 H_m 能在等价变换下达到标准形，则 H_s 更能达到标准形；反之，若 H_s 不能达到标准形，则 H_m 更不能达到标准形。 □

定理2指明了判断一个循环是否可以并行执行的又一办法。例如图3所示循环 L，若 L 能够并行执行，则循环体中任意语句组成的循环可以并行执行。另一方面，若循环体中某些语句组成循环不能并行执行，则 L 亦不能并行执行。

4 并行性特征分析

4.1 并行性特征定义

定义3 简化了的并行程序，即进程 P 的 BNF 范式定义如下：

$$P=0\mid P.d\mid\alpha.P\mid P+P\mid P\mid P\mid P\backslash\alpha$$

其中：0表示进程 P 终止或死锁；P.d 表示进程 P 对数据集 d 操作；α.P 是前缀操作项，表示先执行动作 α，后执行进程 P；P+P 是求和操作项，表示两进程至少有一个执行；P|P 是组合操作项，表示两进程并行执行；P$\backslash\alpha$ 是限制操作项，表示 α 操作不能在进程 P 中执行。

定义4 设有进程 P 和数据集 D，D 可划分为多个子集 $D=\{d_1,d_2,\cdots,d_n\}$，$d_i\neq d_j, i,j=1,2,\cdots,n$，如果 $P.D=P.d_1\mid P.d_2\mid\cdots\mid P.d_n$ 成立，则称 P 对 D 具有数据并行性。

定义5 设有进程 P 和数据集 D，P 可划分为 $P=\{p_1,p_2,\cdots,p_n\}$，$p_i\neq p_j$，D 可划分为 $D=\{d_1,d_2,\cdots,d_n\}$，$d_i\neq d_j, i,j=1,2,\cdots,n$，如果 $P.D=p_1.d_1\mid p_2.d_2\mid\cdots\mid p_n.d_n$ 成立，则称 P 对 D 具有任务并行性，或称 P 对 D 具有功能并行性。

定义6 对于程序 P，如果 $P=\alpha_1.\alpha_2\cdots\alpha_n$，则称 P 具有 SISD 结构，记为 P→SISD。

定义7 设有程序 P，数据集 $D=\{d_1,d_2,\cdots,d_n\}$，$d_i\neq d_j, i,j=1,2,\cdots,n$，转换规则集 $R=\{r_1,r_2,\cdots,r_m\}$：如果 $P\xrightarrow{r1}P'\xrightarrow{r2}\cdots\xrightarrow{m}P^s=P^*.d_1\mid P^*.d_2\mid\cdots\mid P^*.d_n$ 成立，则称 P 蕴含 SIMD 结构，记为 P→SIMD。

定义8 设有程序 P，数据集 $D=\{d_1,d_2,\cdots,d_n\}$，$d_i\neq d_j, i,j=1,2,\cdots,n$，转换规则集 $R=\{r_1,r_2,\cdots,r_m\}$，如果 $P\xrightarrow{r1}P'\xrightarrow{r2}\cdots\xrightarrow{m}P^s=\{P_1^*,P_2^*,\cdots,P_n^*\}=P_1^*.d_1\mid P_2^*.d_2\mid\cdots\mid P_n^*.d_n$ 成立，则称 P 蕴含 MIMD 结构，记为 P→MIMD。

4.2 并行性特征判据

定理3 对于图3所示循环 L，令 U_i 为循环控制变量取 i 值时生成的结点集，C_{0i} 表示第0辈结点集中，被

循环控制变量取 i 值时的那次循环迭代所引用的结点集，则当：$(\alpha\in U_i \wedge \beta\in U_i \wedge (C(\alpha,\beta)\vee R(\alpha,\beta)))\vee(\alpha\in G_{0i}\wedge\beta\in U_i\wedge R(\alpha,\beta))\Rightarrow(i=j)$成立，则 L→SIMD。

证明：任取两次循环 m，n，且 m≠n。根据已知条件，则 $U_m\cap U_n=\phi$。设$\forall\ \alpha\in U_m$，$\forall\ \beta\in U_n$，则 $C(\alpha,\beta)$，$C(\beta,\alpha)$和 $R(\alpha,\beta)$，$R(\beta,\alpha)$都不成立，因此第 m 和第 n 次循环迭代无数据相关依赖关系。由于 m，n 的任意性，所以循环 L 可整体并行执行。L 可变换成如下：

```
DOALL(i=I_L;i<I_U;i++)
{  ……
   A_k,c(i_k,0)=f_k(A_k,1(i_k,1),A_k,2(i_k,2),…,A_k,nk(i_k,nk))
   ……
}
```

对其中任何一条语句，令 $P=[A_{k,0}(i_{k,0})=f_k(A_{k,1}(i_{k,1}),A_{k,2}(i_{k,2}),\cdots,A_{k,nk}(i_{k,nk}))]$，$d_i=A_{i,0}(i_{k,0})$，$I_L\leqslant i<I_U$，则 $P\cdot d_{I_L}\mid P\cdot d_{I_{L+1}}\mid\cdots\cdots\mid P\cdot d_{I_U}$ 成立，所以 L→SIMD。

定理4 对于图3所示循环 L，令 U_x 为由第 x 条语句在循环空间上生成的结点集，G_{0x}表示执行第 x 条语句期间引用第0辈结点集中的结点子集，则当：$(\alpha\in U_x\wedge\beta\in U_y\wedge(C(\alpha,\beta)\vee R(\alpha,\beta)))\vee(\alpha\in G_{0x}\wedge\beta\in U_y\wedge R(\alpha,\beta))\Rightarrow(x=y)$成立，则 L→MIMD。

证明：任意选择循环体中两条语句 m，n，且 m≠n。根据已知条件，则 $U_m\cap U_n=\phi$。设$\forall\ \alpha\in U_m$，$\forall\ \beta\in U_n$，则 $C(\alpha,\beta)$，$C(\beta,\alpha)$和 $R(\alpha,\beta)$，$R(\beta,\alpha)$都不成立，因此第 m 条和第 n 条语句之间无数据相关依赖关系。由于 m，n 的任意性，所以循环体内所有语句不存在相关性。令：

$P_1=[$FOR(i=I_L;i<I_U;i++) $A_{1,0}(i_{1,0})=f_1(A_{1,1}(i_{1,1}),A_{1,2}(i_{1,2}),\cdots,A_{1,n1}(i_{1,n1}));]$

$P_2=[$FOR(i=I_L;I<I_U;i++) $A_{2,0}(i_{2,0})=f_2(A_{2,1}(i_{2,1}),A_{2,2}(i_{2,2}),\cdots,A_{2,n2}(i_{2,n2}));]$

……

$P_m=[$FOR(i=I_L;i<I_U;i++) $A_{m,0}(i_{m,0})=f_m(A_{m,1}(i_{m,1}),A_{m,2}(i_{m,2}),\cdots,A_{m,nm}(i_{m,nm}))]$

所以，L 可变换成 $P_1\mid P_2\mid\cdots\cdots\mid P_m$，显然 $P_1\neq P_2\neq\cdots\neq P_m$，所以 L→MIMD。 □

定理5 对于图3所示且不能并行执行的循环 L，如果存在某些控制变量 $I_1,I_2,\cdots,I_k$，并且记 L 在循环区间$[I_L,I_1)$，$[I_1,I_2)$，…，$[I_k,I_U)$上对应的子循环分别为 $L_1,L_2,\cdots,L_k$，若 L_1→SIMD，L_2→SIMD，…，L_k→SIMD，则 L→SIMD。

证明：因为 $L=L_1\cdot L_2\cdot\cdots\cdot L_k\cdot L_1\cdot L_2\cdot\cdots\cdot L_k$ 串行执行的语义即是 L 的语义，根据定义故定理显然。 □

定理5告诉我们，在循环程序整体不能并行执行的情况下，可通过循环区间分割开发局部并行性。例如循环程序：FOR(i=100;i>=0;i--) A(i-7)=A(-3*i+14)+A(i-7)；可分割区间为[100,6]，[5,0]。

进一步研究发现：对于如图3所示循环 L，设循环体中语句总数为 M，时序层次图结点集为 N，∀ node∈N，记 node.gn 为该结点的辈分，则 L 循环区间可分割的必要条件是：$\max(node.gn)<(I_U-I_L)*M$。

定理6 对于图3所示且不能并行执行的循环 L，如果将循环分布到循环体中每条语句上对应的循环分别为 $L_1,L_2,\cdots,L_m$，若 L_1→SIMD，L_2→SIMD，…，L_k→SIMD，则 L→SIMD。

证明：类似定理5的证明。 □

根据定理2可知，定理6的条件难于成立。通常情况下，$L_1,L_2,\cdots,L_m$ 不可能全部都能并行执行。但是，$L_1,L_2,\cdots,L_m$ 之中某些能够并行执行，开发出它们的并行性比循环 L 整体串行执行又前进了一步，我们称这种处理技术为循环分布。显然，开发循环分布并行性的必要条件是：存在某条语句产生的所有结点均在同一辈分上。

综上可知，开发循环最大并行性过程应该是：循环整体并行性，语句循环分布并行性和循环区间分割并行性。

以上对图3所示循环作了较全面的研究，但是如果循环体中包含分支控制语句，例如图6所示循环 L_C，情况又会怎么样？我们作一种特殊的处理，将控制依赖转换成数据依赖。具体的做法是：把 IF 语句的首部 IF(ei)改写成逻辑数据暂存语句 #LE(i)=ei，将 ELSE 关键字消失，变换后的循环记为 L_D，它和图3所示循环具有同样的结构，于是，我们得到下面的结论。

```
FOR(I=I_L;i<I_U;i++)
{  IF (ei)
      A_1,0(i_1,0)=f_1(A_1,1(i_1,1),A_1,2(i_1,2),…,A_1,n1(i_1,n1));
   ELSE
      A_2,0(i_2,0)=f_2(A_2,1(i_2,1),A_2,2(i_2,2),…,A_2,n2(i_2,n2));
}
```

图6 具有控制结构的循环程序

定理7 对于图6所示循环 L_C，控制依赖变为数据依赖之后的循环为 L_D。若 L_D 的时序层次图能够变换达到标准形，并且逻辑数据暂存语句产生结点的辈分总是先于 IF 和 ELSE 之后的语句，则 L_C 可并行执行。

证明：类似定理1的证明。 □

一般而言，图6所示循环 L_C 比它的变换体循环 L_D 并行执行的条件要严格，可见存在 $L_C\subset L_D$ 关系。

例如对如下循环程序：FOR(i=M;i<N;i++) IF(B(i)==0) B(i+1)=B(i+2)；其 L_D 有标准形，但 L_C 尚不能并行执行。

对于循环程序：FOR(i=1;i<100;i+=2) IF(A(i)<0) A(i)=A(i-1)+A(i+1))/2；满足定理7的条件，能够并行执行。

（下转第70页）

请求，将信息发布到管理决策者手中，并利用 PUSH 技术将日常报表、重大异常或告警及时推送给相应的人员，用户通过局内办公自动化系统连接到系统的 Web/PUSH 服务器，并用网关将它们与网管系统的其它设备隔离。该系统的软件结构如图3所示。

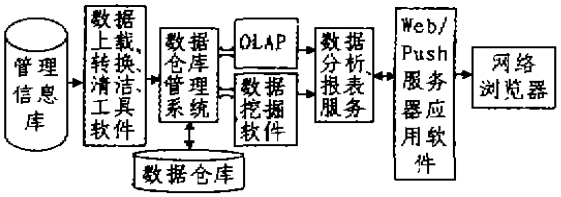

图3 本地网网管决策支持系统软件结构示意图

本应用系统中的软件体系是以数据仓库为核心，通过数据提取工具、数据转换工具、查询工具、报表工具、分析工具、数据挖掘工具等，满足用户对信息的各种需求。

在本系统中，我们主要考虑了对网络性能数据（话务量统计数据和网络配置数据）的利用。对管理信息库中的话务数据如忙时各中继群、目的码话务变化、每线话务量、电路可用率、电路溢出比、电路忙比等按日、月、年来进行分类、综合，并把这些信息以图表的形式 Push 给网络管理人员或供相关人员查询使用，网络管理人员从这些信息中可以分析网络性能、网络运行状况、网络负载变化趋势等信息。这些信息同时也为网络管理人员做出电路调度、阻塞控制、网络规划等提供决策依据。该系统已经用于实际工作中。

结束语 本文在分析了本地网网管系统现状的基础上，结合网管工作的实际需求，提出以本地网网管系统的管理信息库为数据源，建立基于数据仓库的网络管理决策支持系统的思路和实现方案，并初步实现了在本地网网管与监控系统基础上的应用系统并在实际工作中使用，取得初步成效。然而，研究工作仅仅是初步的，许多问题特别是相应的数据仓库模型和数据挖掘方法需要进一步探讨与研究。

参 考 文 献

1 [美] Hammergren T. 数据仓库技术. 中国水利水电出版社，1998

2 王珊，等. 数据仓库技术与联机分析处理. 科学出版社，1999

3 王广清，杨学良. 数据仓库技术及其在电信计费领域应用的探讨. 计算机工程与应用，1999，9

4 张伟民，等. 数据仓库技术在电信管理网中的应用 电子工程师，2000

（上接第19页）

定理8 对于图4所示循环 L_i，如果能够并行执行，则 L→MIMD。

证明：根据定义易证。 □

定理9 对于任意循环 L，如果¬(L→SIMD)∧¬(L→MIMD)成立，则 L→SISD。

证明：根据定义显然。 □

结束语 比较基于语句，基于数组变量和基于数组元素的三种方法，研究对象的粒度依次减小。基于语句的方法，判断循环整体是否可并行执行比较容易。基于数组变量的解析方法可以研究多重循环的数据相关性。基于数组元素视图方法，既有图直观的优点，也具有代数理论形式化的支持；既能研究循环整体并行化，又能判定循环是否存在部分并行性，还可讨论包含控制结构的循环程序的数据相关性问题，不愧为是一种值得深入研究的方法。

数组元素时序层次图随着循环体语句数增多以及循环步数的增多，结点数也将增多，导致处理数据和运算量特别大，给实际应用带来一定的困难。本文仅是对该方法进行了起步性研究，还存在一些问题，例如：时序层次图标准形自动转换算法怎样构造，如何在图上分割循环区间实现部分并行，以及如何有效构造二重、多重循环的时序层次图。这些问题是我们今后的研究工作。

参 考 文 献

1 Freund R F, Siegel H J. Heterogeneous processing. Computer, 1993, 26(6): 13～17

2 Ekmecic L, Tarraljа I. A survey of heterogeneous computing: concepts and systems. Proceeding of the IEEE, 1996, 84(8): 1127～1143

3 Khokhar A A, et al Heterogeneous computing: challenges and opportunities. Computer, 1993, 26(6): 18～27

4 Zeng Guosun, Lu Xinda, Wang Jincun Extracting Loop-Level Parallelism and Scheduling in Heterogeneous Computing Environment. In: The Third International Workshop on Advanced Parallel Processing Technology, Oct. 1999 228～230

5 Zeng Guosun, Lu Xinda, Wang Jincun. Structure-based automatic extraction of the program heterogeneity. HPC-Asia 2000, May 2000, to appear

6 曾国荪，陆鑫达. 异构计算中的负载共享. 软件学报，2000, 11(4): 551～556

7 Fan Zhihua. Discreteness of sequential hierarchy and its application. Scientia Sinica, 1988, 31(10): 1269～1279

8 曾国荪. 自动提取程序的异构特征实现异构计算. [博士学位论文]. 上海交通大学计算机科学与工程系，2000