基于知识的数据自动分布模型*

杨 莉

(国家智能计算机研究开发中心,北京 100080)

葛建新

(浙江大学数学系,杭州 310027)

何志均

(浙江大学计算机系,杭州 310027)

摘要 基于松散耦合的多处理机系统(DMS)的发展,要求有相应的并行软件工具的支持,而在目前的大多并行软件系统中,用户必须显式给出顺序执行程序的划分方法以及在DMS中不同处理器上的分配算法.针对这一问题,本文首先介绍了一种基于知识的数据自动分布模型(KBDM),以期对程序的划分和分布问题提供自动的支持,然后讨论了KBDM设计的主要思想,并给出该模型主要部件的详细描述.

关键词 并行性,数据分布,基于知识的程序重构,模式匹配.

近年来,DMS(也称为具有分布式存储的多处理机系统)在科学计算领域被广泛地推广使用.然而,与DMS相连的程序流程却使用户面临着很多本质上难以克服的问题.关键之处在于,用户必须手工组织在DMS上运行的相互间独立进程的正确同步,并保证进程运行过程中所需的数据项在需要时可在本地存储器(LM)中得到[1].

目前,DMS要求其上运行的程序具有明显的并行可编程性,这既可通过并行程序语言,如OCCAM[2]来得到,也可通过在传统顺序程序语言中引入用于数据传递的操作原语来达到此目的.然而上述两种方法都需要用户极其繁琐地处理上面所提及的关键问题,这又使得DMS上的编程成为既耗时又容易出错的一项工作.针对这一问题,人们通过向DMS的使用者提供虚拟共享存储器来进行解决,目前这方面的工作既可以通过硬件来实现,也可以通过合适的软件工具来实现.

软件研究集中在并行任务自动化方面的工具上,这些工具的目标是希望能自动地把顺序执行的程序转变为很明显的并行程序.现在用于并行性的编译技术已基本成熟,出现了一些较成熟的原型系统,如KAIL,SUPERB,MIMDizer.在这些系统中,并行性是依据在DMS的处理器之间合理地划分程序数据,并以下列约定在处理机上分配工作负载来进行开发的[3]:DMS中每个处理器在执行所分配的任务时所需的数据在它的LM中可获得.

---

* 本文1993-09-27收到,1994-07-27定稿

本研究得到国家自然科学基金的支持.作者杨莉,女,1965年生,副研究员,主要研究领域为人工智能,专家系统,神经网络,并行计算机体系结构等.葛建新,1965年生,博士后,主要研究领域为人工智能,CAD,图形学,CIMS,用户界面等.何志均,1923年生,教授,博士导师,主要研究领域为人工智能,专家系统,CIMS等.

本文通讯联系人:杨莉,北京100080,国家智能计算机研究开发中心

因而,我们可知并行过程是由数据分布决定的.另外,上述系统都要求用户提供程序数据在DMS中处理器之间分布的明确方法,这样一来,编译器的任务从本质上说只包括按下述方法进行的程序重构:每个处理器执行所有任务所需的程序数据已经映射给它,这包括在必要时插入一些信息交换操作原语以获取非本地数据.

由此也可看到,数据分布的指定是决定性的,并且是并行处理过程中最为关键的一步,它需要有关程序的结构以及已重构程序中通讯费用方面的详细知识,因为任何错误的程序数据分布都可能导致过多的机间通讯,从而使得目标程序的性能不可接受.

本文我们提出了一个基于知识的数据自动分布模型KBDM,KBDM通过知识库中的启发式规则和其本身带有的模式匹配器(推理机)自动地决定一个合适的数据分布.另外,KBDM也可在交互式系统框架下给用户提供一些详细的程序执行信息以支持数据分布的一个手工选择.本文详述KBDM使用的基本原理,它的主要设计思想和体系结构中各功能模块的详细说明.

1　自动并行性的基本概念

本节我们通过KBDM软件工具的一些关键特点的介绍,来讨论用于DMS的自动并行性的基本概念.

假设对KBDM的输入是用FORTRAN 77编写的程序Q、一个进程集合P,设在目标机的每个处理机上至多有一个进程在执行,这里每个进程可以和各自不同的LM相连,每个进程在它的LM里可直接访问本地数据,在LM里不存在的数据必须通过机间通讯才能得到.

通过KBDM,Q被转变为明显的并行程序Q^*.Q^*是FORTRAN语言的扩展,它包括信息交换操作原语SEND,RECEIVE和VECTOR构造子.Q^*和Q在语义上等价,且被P中进程处理.这里特别提一下KBDM在优化目标代码方面所做的一些努力.KBDM把程序Q中的语言分为一般陈述句、循环语句、函数和过程,KBDM通过在可能情况下从循环中抽取通讯信息,并把多个通讯语句打包组成块通讯以降低整个通讯费用.这种优化可以通过在知识库中存储有关程序代码中精确的数据相关性方面的启发式知识来实现,这里有关数据以何种方式被分布的精确知识也已存储在知识库(KB)中.

由于数组在程序中举足轻重的作用,下面讨论程序Q中任一数组A的划分.在并行性处理时,对A的划分是通过把A分割成不同的子数组A_i,然后映射A_i到P中的处理器上$(1\leqslant i\leqslant n)$.

令A为Q中任意一个数组,我们把A看成它的数组元素组成的集合.在KBDM中,对A的划分是把A分成很多不相交的子集,且这些子集的并集等于A,在2.2节我们将进一步讨论KBDM中划分和映射方法.

我们已知用于划分二维数组的一些标准方法,如按行、列和块的方法.按上述方法可以把任一数组分割成不同的子数组.值得注意,在KBDM中,不要求块是同体积的,特殊情况为平凡划分,即数组被划分成仅仅一个子数组,即与原数组相同.在KBDM的知识库中认为所有标量变量集是平凡划分的数组.

一个子数组 A_i 是数组 A 划分后的一个元素，A_i 能被映射到 P 中一个或多个进程上. 如果 A_i 被映射到 P 的每个进程，我们说 A_i 被拷贝，在KBDM 中，认为所有标量变量集以这种方式被拷贝.

一个进程 p 的局部变量集是由映射到 p 的子数组集中所有的元素组成，包括所有标量变量. 我们说进程 p 拥有自己的局部变量，"拥有"有两个重要的含义. 首先，p 拥有的每个变量 V 在 p 的局部地址空间内；其次，p 对于出现在源程序里的变量 V 必须执行赋值操作.

程序数据被划分和映射的方法也决定并行程序的进程结构，特别是所需的机间通讯，因而它也决定并行程序的整体性能. 影响划分的因素常包括：应用范围，DO 循环的限制，数组使用模式以及一个循环里陈述句间相关性的结构等.

2 KBDM 模型

2.1 程序类别

KBDM 模型的目标是使程序数据划分和映射任务自动化，它用来处理用 FORTRAN 77 编写的任意程序. 因为结构化程度较高的程序并行性较好，且并行性策略对于松散同步程序可能产生更好的结果，因而在 KBDM 中我们主要考虑处理松散同步程序(LSP)[4].

• 一个 LSP 被分解为通讯和计算的交替过程. 在计算阶段，DMS 中所有处理器在各自不同的 LM 上执行相同的程序计算过程. 一但某个处理器完成一次计算过程，它使用 SEND 操作原语把最新计算结果发送给向它发出 RECEIVE 操作原语的所有处理机，并用 RECEIVE 操作原语接受它本身用于继续计算所需的数据，然后又开始下一个计算过程.

由上可知，在处理 LSP 时，仅仅同步执行的处理机之间需要交换数据. 在通信和计算过程明显独立的情况下，松散同步问题是最适合于自动并行性划分和执行的.

2.2 数据分布

由第 1 节可知，数组是程序中最基本、最重要的数据类型. 因为程序中所有标量变量组成的集合可以被看作平凡划分的数组，故本文着重讨论数组的划分和分布问题.

数据自动分布的任务是在 KB 中寻找、发现一个合适的知识 K，并用 K 划分和映射程序数组元素到目标机(DMS)的处理器集合上. 在 KBDM 中，为了正确、有效地进行数组元素的分布，我们在 KB 中设置了很多元级知识，以对数据分布进行合理的限制.

下面首先给出用于数据分布的重要概念[5].

分布的概念既包含了划分，又包含了映射. 假设：

• P 是 DMS 中物理处理器的集合，我们将 P 解释为一个处理器的矩阵；

• M 是程序里所有数组的集合，对任一 $A\in M$，A 被解释为它的元素集合.

结论：数据分布是一个从数组元素到处理器集合上的映射函数.

定义 2.1. 数据分布函数 δ 定义为：$\varepsilon\to\rho(P)$，这里 ε 指示所有数组元素的集合，对任一 $e\in\varepsilon$，$\delta(e)=P_e$，$P_e\neq[]$，这里 P_e 为处理器的集合，且对任一 $P_i\in P_e$，P_i 拥有 e("拥有"的概念见第 1 节). 对每一 $A\in M$，δ^A 定义为 δ 到 A 中元素的限制，即 $\delta|_A$.

因此，δ^A 指示 A 中元素到 P 中处理器的映射. 下面我们看看在何种情况下，δ^A 由 A 到 P 的映射函数变为 A 到 P 的一个分布.

令 $\lambda:P\to\rho(\varepsilon)$($P$ 为处理器集),对任一 $p\in P$,$e\in\lambda(p)$,当且仅当 $p\in\delta(e)$,且令 λ^A 是 δ^A 的反函数,这里 λ^A 为 λ 到 $\rho(A)$ 中元素的限制. 因而 p"拥有"的变量集合是 $\lambda(p)$.

定义 2.2. 对某一 p,$p'\in P$,如果 $\lambda^A(p)\cap\lambda^A(p')\neq[]$ 时,$\lambda^A(p)=\lambda^A(p')$ 成立,则我们称 δ^A 是一个合适的分布.

如果对任一 $A\in M$,δ^A 是一合适的分布,那么 δ 为一合适的分布.

定义 2.3. 对所有 $A\in M$,且对任一 $p\in P$,如果 $\lambda^A(p)$ 是 A 的矩形子数组(这里 $\lambda^A(p)=A_i$,A_i 是 A 的子集,且对任一 $e\in A_i$,$\delta(e)=p$),那么我们称 δ 是一标准分布.

非标准分布必须以一种特别的方式进行处理,我们将另文讨论.

注意,如果数组 A 被合适地分布,且存在处理器 p,$p'\in P$,$p\neq p'$,p 和 p' 拥有 A 中相同的元素,我们称数组 A 被拷贝,被拷贝数组元素的所有赋值在拥有它们的所有进程中执行.

KBDM 中数据分布工作既要决定数组划分成子数组的一个合适的方法,又要决定这些子数组映射到一个或多个处理器上的方法,即对于程序中的数组元素,它必须通过 KB 中的启发式知识和 KBDM 的模式匹配器寻找、发现一个合适的分布函数.

在程序里经常性地要确定数组的类型,因而对相同类型的数组赋予相同的分布函数是很合理的,这部分的详细讨论请见第 3 节,下面我们用例子来说明正确选择划分的重要性.

矩阵乘:考虑一个矩阵乘运算:$C=A\times B$,这里 A、B、C 是 $N\times N$ 矩阵,且我们想把这个乘法工作分布到 N 个处理器上进行. 开始时使用通常的矩阵乘算法:划分 A 用行,划分 B 用列,使得数组元素 $A(I,1:N)$ 被赋到处理器 I 上,$1\leqslant I\leqslant N$,且元素 $B(1:N,J)$ 赋到处理器 J 上,$1\leqslant J\leqslant N$. 第一步,C 按 A 的方式进行划分,用上述划分和映射方案,仅仅 B 的列必须和所有处理器进行通讯. 这里相关矩阵乘算法的结构可以流水线方式执行.

开始,A 的第 I 行和 B 的第 I 列被调入第 I 台处理器,$1\leqslant I\leqslant N$,且进行相乘,在此之后,B 的 I 列被传输到第 $I+1$ 台处理器($I+1$ 对 N 取模),并取代以前存储在第 $I+1$ 台处理器上 B 的第 $I+1$ 列,且和 A 的第 $I+1$ 行相乘,这一过程继续进行直到执行完 N 步结束. 因此,这一算法的整体通讯费用是矩阵 B 的 N 次"转换",这里每次"转换"等于在一个超立方体上 N 个近邻通讯的同时执行.

现假定:矩阵 A,C 用列划分,矩阵 B 用行划分. 首先 A 的 I 行从所有其它处理器上收集并集中到处理器 I 上,且在每个后继乘法步(处理器 I 上,A 的 I 行和 B 的 J 列相乘)将存在处理器 I 和每个其它处理器的通讯,以收集 B 的第 J 列到处理器 I 上. 从上面的分析也可知,有经验的编程人员可通过手工来改进程序以提高效率,但性能仍比第一种方法更糟.

2.3　一般的方法

KBDM 的数据分布方法是基于以下两点:先进的程序分析方法和基于知识的人工智能(AI)技术. 这里值得一提的是,KBDM 中所使用的模式匹配机制与显式的知识表示和知识重构技术的结合,因为理论与实践的结果证明,一个优化的数据分布不能单靠分析来发现,人们往往依靠大脑中的经验知识来做出很多重要的决定.

· 程序分析　当代并行性开发要进行大量的程序分析,在这一过程里可获得关于程序结构的信息和数据划分过程中所需要的数据分配与使用的模式(特别是在循环里).

· 基于知识的技术　启发式规则被广泛地用于减少搜索空间,并用于确定结论应被满足的先决条件. 对程序代码而言,模式匹配机制用于识别几种不同类的数据分配和使用的模

式，特别地，用于检测一定类型的 VECTOR 构造子. 在 KBDM 中，启发式知识和模式匹配信息被存放在知识库中.

• 相互作用 KBDM 是一个交互式系统，它使用户能够提供系统无法得到的信息；提示程序代码里特别重要的区域. 例如，对变量，用户可以提供它可能值的范围或提供循环界线，用户也可能被系统要求去指明一个典型输入值的集合或去帮助任务的重新划分，与此同时，KBDM 也允许用户明确指示一个或多个数组的分布.

3 KBDM 的结构和功能

KBDM 的体系结构如图 1 所示，由分析和标准化模块 AN、权探测器 WF、粗粒度启发式知识和模式匹配器 HKMM、代码分析器 CA 和一致化部件 unify 5 大模块组成. 它的输入是一个 FORTRAN 77 程序和一个目标机的详细说明. 在 KBDM 中对数据的分布相应地可分成 5 个阶段，这一过程的逻辑结构如图 2 所示. 下面我们逐个讨论每个阶段.

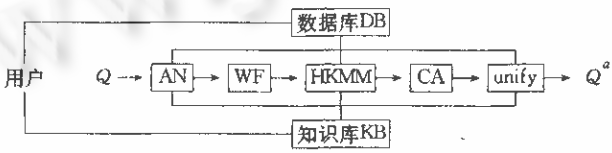

图1 KBDM的总体结构

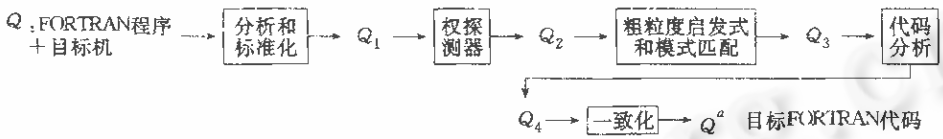

图2 KBDM数据分布的逻辑结构图

1. 分析和标准化模块 AN

首先进行输入程序 Q 的详细分析，然后 Q 经历很多最优化和标准化变形，特别地 Q 接受传统编译器的分析（语法和语义分析），以及先进的程序分析（数据和控制流分析以及相关性分析）. AN 所使用的变形方法包括变量预定替换，下标标准化和隐含循环到明显 DO 循环的转换. AN 过程后所得到的程序分析信息被存放于程序数据库（DB）中，以便后继阶段参考. AN 执行的结果得到原始程序的一个标准化表示 Q_1，并用 DB 中的信息进行注释.

2. 权探测器 WF

KBDM 中权探测器 WF 的作用是在程序里检测出最重要的代码区域. 在 WF 中我们仅关心程序代码的执行时间，并把执行时间和一个权值关联起来，这个权反映程序执行可能需要的时间量. 因为程序中的某些区域，在以后执行中要花费很大代价，那么找到它们的合适分布就显得非常重要. 经验表明，这些区域可能只占原代码的 5—10%，但导致 KBDM 工具整体工作负载的降低.

对输入程序 Q_1，WF 必须决定个体程序段的相对和绝对重要性，这些个体程序段包括单一陈述句或陈述句块、循环、过程和子程序. 对上面每类程序段，WF 产生一个值对

——权,表示相应程序段的基本执行时间和它被预期执行的次数,这些权对今后程序数据的划分和执行将起到重要的指导作用.

WF 的输出是程序 Q_2,且 Q_2 的每个程序段被赋予一个权值. 注意:权探测器 WF 是在顺序程序上执行,它发现顺序程序执行中的加强区域,这可借助于静态性能分析和动态技术的结合来实现.

3. 粗粒度启发式知识和模式匹配器 HKMM

KBDM 下一阶段工作有两个重要任务去实现. 首先对程序 Q_2,它试图在整个数据划分和分析上抽取尽可能多的限制. 这个任务的主要一点是在程序的数组集合里检测、发现相互之间密切配合的数组子集.

为了讨论方便,我们定义数组间的密切配合(LC)如下.

定义 3.1. 令 Ar 是数组 A 的第 r 维,Bs 是数组 B 的第 s 维,$A\neq B$,那么 Ar 和 Bs 是 LC 当且仅当无论何时 A 和 B 出现在一个 DO 循环的相同语句里,Ar 和 Bs 的下标表示是同一的. 如果 A、B 都是 d 维数组,且对任一 $i=1,\cdots,d$,A_i 和 B_i 是 LC,那么我们说数组 A,B 是 LC.

由定义 3.1 可知,LC 是一等价关系.

在 KBDM 中,划分程序代码里的数组为几种类型,使得相同类型中的数组之间是 LC. 这样一来,我们可把程序代码的数组分为几类,且每类数组以相同的方法被使用.

KBDM 中发现数组类的方法是基于启发式知识和模式匹配器的推理. HKMM 执行对数组的一个简单句法分析,确定循环界线并对程序代码中数组的各种分配和使用进行比较. 在 KBDM 的 KB 中存放的启发式知识和元级知识,指示程序里数组分析和使用的一些规律性,这些规律性给出在程序里说明的一部分数组和一些数组类相联,这些数组类中的数组有相同的维数和说明,且以相同的方法被使用. 在程序中不同用途的数组之间的这些既简单但非常关键的关系我们用下例进行说明:

```
DO i=1, nnx
    DO j=1, nny
    rh(i,j)=sigx(i,j)
    un(i,j)=sigy(i,j)
    vn(i,j)=sigth(i,j)
    en(i,j)=tauxy(i,j)
    tn(i,j)=qx(i,j)
    pn(i,j)=qy(i,j)
    hn(i,j)=en(i,j)+pn(i,j)/rh(i,j)
       DO 2998 k=1, ncs
       fn(i,j,k)=rx(i,j,k)
       END DO
    END DO
END DO
```

上面例子中,fn 和 rx 是 LC 且属于相同等价类,这一结论对上面的其它数组也成立. 同时也应注意到,fn 的第一维在循环中和所有其它数组的第一维是 LC. 上述这些关系在 HKMM 中被确定下来.

这里要说明的一点是,HKMM 也识别和处理为了效率的原因已重构的数组.

HKMM 执行的第二个任务是对程序代码中几种公用数组的分配和使用方法进行检

测，并在 KBDM 的 DB 中作适当的记录. 对程序代码，在不同数据分布下有关它们性能的相关信息或者限制它们分布的特定规则被存储在系统的 KB 中，它包括化简分解操作、BLAS 操作等. 注意到，上述执行过程仅仅在有充分大权值的代码域中被匹配和推理. 为数组的分配和使用模式提供的性能信息可被考虑为一个代价函数，它指示在推理、匹配过程中相关变量的数目、目标机的结构以及数组在不同数据分布下执行的代价.

HKMM 执行完毕产生程序代码的新版本 Q_3，并同时获得所得到数据分布函数使用上的限制、描述已经匹配模式的注释. 在 HKMM 运行过程中，假设对匹配模式的每一个费用函数在 KBDM 的 KB 或 DB 中已经存在.

4. 代码分析器 CA

HKMM 用来在源代码中发现数组的分配和使用模式，相应的，代码分析器 CA 的任务是根据目标机的相应特点，在程序代码中对于最为重要的数组找到分布.

CA 集中它的注意力在由权探测器 WF 识别出的 DO 循环上，这些 DO 循环在整个代码执行时间上是最为关键的，但它们未被 HKMM 的模式匹配器所识别. 对这些 DO 循环，CA 对于实际使用的数组进行一些可能的分布，且得到每个数组的通讯模式，所有这些的完成借助于确定所使用的哪些数组需要进行通讯，然后使用数据相关性信息去寻找、发现这些通讯是否能从循环中抽取出来并确定这些通讯能否进行打包来实现. 在此状况下，系统能够给出目标机的一些性能特点，以便预测在给定数据分布下的执行时间，这又是一个依靠问题大小和目标机性能的代价函数，这个代价函数预测在特定的数据分布下 DO 循环的执行时间.

注意到在一定的环境下，人们可以靠分析来得到程序代码的最优分布，但目前这方面研究、应用的领域还比较狭窄.

在给定的区域内，DO 循环和它们的权值可用来评估在所给定的分布下程序代码的整体性能. HKMM 所提供的信息在 CA 阶段被使用. CA 段结束处，对应计算上每个重要的程序代码区域的相应代价函数被确定下来，这个代价函数表示程序代码在所包含数组的特定分布下的执行时间. 这些代价函数和与此相关的数据分布被包含在目标程序 Q_4 中.

5. 一致化部件 unify

unify 起到一个门阀的作用：因为在一个程序里可能有几个重要的代码区域，并且这些代码区域对一些数组来说并不一定导致相同的数据分布，这时 unify 必须确定是否要进行数据的重新划分. 因为数据的重新划分是很费时的，人们一般不敢冒然使用.

一致化部件 unify 必须使用由 HKMM 所提供的信息来决定没有被 CA 所处理的数组元素的分布函数. KBDM 执行的最后阶段，是在 FORTRAN 源代码里插入合适的划分注释，以产生最终输出程序 Q^a——一个扩充的 FORTRAN 超集程序版本，Q^a 具有明确的数据划分和映射知识.

4 结 论

建造一个并行化工具是一个难度很大的工作，尤其是对于支持并行性工作，并基于人类知识的数据自动分布工具的建造更是如此. 本文通过 KBDM 体系结构和系统各功能部件的详细介绍，论述了用于 DMS 自动并行性的理论和实现技术，并对关键问题用实例加以说明. 从文章中也可以看到，我们在已有数据分布方法和成熟的人工智能技术基础上所提出的

KBDM 设计思想,无疑是对并行软件工具研究的一个新的探索.

参考文献

1 Fox G, Johnson M, Lyzenga G *et al*. Solving problems on concurrent processors. Prentice Hall International Inc., 1988.

2 INMOS Ltd. Occam 2 reference manual. Prentice Hall Int., Series in Computer Science, 1988.

3 Gerndt H M, Zima H P. Optimizing communication in SUPERB. Proc. Conpar 90—VAPP IV, LNCS 457, 1990. 300—311.

4 Compass Inc. SUPRENUM Fortran reference manual. Compass Inc., Wakefield MA, 1989.

5 Balasundaram V, Fox G, Kennedy K *et al*. An interactive environment for data partitioning and distribution. Proc. DMCC5, Mar 1990. 1160—1170.

KNOWLEDGE—BASED DATA AUTOMATIC DISTRIBUTION MODEL

Yang Li

(*National Research Center for Intelligent Computing Systems, Beijing* 100080)

Ge Jianxin

(*Department of Mathematics, Zhejiang University, Hangzhou* 310027)

He Zhijun

(*Department of Computer Science, Zhejiang University, Hangzhou* 310027)

Abstract　The development of distributed—memory multiprocessing system (DMS) needs the support of the corresponding parallelization software tool. In current parallelization systems, the user must explicitly specify how the data domain of the sequential program is to be partitioned and mapped to the processors of DMS. In this paper, the authors present a knowledge—based data automatic distribution model (KBDM) that provides automatic support for this task, and then discuss the principal ideas underlying their model, the major components of KBDM and the main features of the distribution strategy employed.

Key words　Parallelization, data distribution, knowledge—based program reconstruction, pattern matching.