基于任务图的一种并行程序设计方法(Ⅰ)
——任务图的设计*

张德富 吴巧泉

(南京大学计算机科学系,南京 210093)

摘要 本文提出一种基于任务图的并行程序设计方法.首先分析欲解的问题,产生数据流程图,并以此设计出表示并行算法的任务图,然后根据任务图选择合适的系统拓扑结构,最后完成并行程序的设计.该方法思路清晰,富有条理,产生的并行程序质量较高.

关键词 任务图,并行程度设计方法,并行算法.

近 20 多年来,并行计算机系统有了显著的发展,其速度和其它性能都比串行计算机高出许多倍.现在并行处理技术的研究重点在于如何在并行计算机系统上设计高效清晰的并行程序.现在的并行程序设计都是就各个系统、各个问题,由程序员精心编制,没有一个规范的工程化方法,编制程序复杂、困难,设计出的程序可读性差,难于调试.为此本文提出了一种基于任务图的并行程序设计方法,以便编制并行程序有一定的方法可依.基于任务图并行程序设计方法的总思想是逐步求精.它来源于结构化串行程序设计,通过分析问题、逐步求精,得到任务图,再把任务图映射到并行计算机结构上,其总的设计流程,如图 1 所示.我们曾按此流程设计了一些并行程序,表明该方法思路清晰,条理性强,产生的并行程序质量较高.

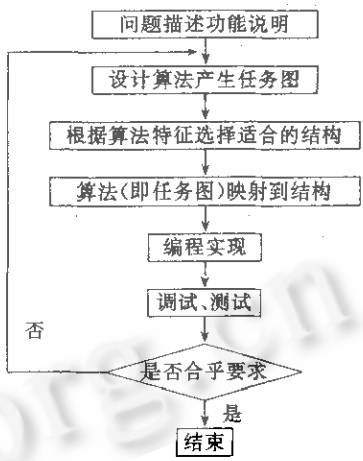

图1 基于任务图的并行程序调计流程

1 任务图的设计

1.1 任务图

定义 1. 任务图是一个有向图,结点代表任务,结点值代表该任务的计算量,有向线代表

* 本文 1993-05-24 收到,1994-02-01 定稿

本研究得到国家 863 计划的资助.作者**张德富**,1937 年生,教授,主要研究领域为计算机软件,并行处理技术和分布式计算.**吴巧泉**,1968 年生,工程师,主要研究领域为并行处理技术和电力自动化.

本文通讯联系人:张德富,南京 210093,南京大学计算机科学系

任务之间的通信,有向边值代表通信量.

任何一个并行算法都可以用任务图来描述.

我们的设计目标是设计出的任务图,一是结构不要太复杂,否则会不清晰,难以划分、映射和调试;二是任务粒度不要太大,否则易丧失并行性,负载也不易平衡,但任务粒度也不能太小,否则会增加通信量和通信复杂性,合适的粒度是使任务图的平均并行度接近处理机数[1].

设计任务图有两种方法:转换分析法和直接分析法.转换分析法先开发数据流程图,再依此设计任务图;直接分析法顾名思义直接设计任务图.但不论哪一种方法,都是采用分解、逐步求精的方法,即把一个复杂的大问题逐步分解为一些简单的相对独立的子问题.

1.2 转换设计法

转换设计法是一种设计策略,它能较好地扩展问题的并行性,其主要步骤:首先用数据流程图表示用户问题,然后把数据流程图转换成任务图,并作一些适当调整,使之优化.

从本质上说,程序系统就是对数据作出一系列的转换.所以画出了用户问题的数据流程图,也就是画出了整个问题的解决过程.

画数据流程图的方式有几种.大多数设计者喜欢从物理输入开始,顺着数据的流向画出一系列的转换,直至到达物理输出为止;而有些设计者则喜欢逆向画,从物理输出开始画到物理输入为止.

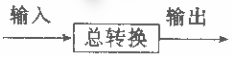

图2 最初的数据流程图

在具体画数据流程图时,本文采用自顶向下逐步求精的方式,从高度抽象逐步过渡到具体转换.首先把数据流程图画成一个带有抽象输入和抽象输出的总转换,如图2所示,然后把这个抽象的总转换分解成若干个比较具体的转换,最后将这些转换分解成更具体的转换.随着这个过程的进行,数据元素也由抽象的逐步转换成具体的,每一步的分解都是用更详细的数据流程图来取代前一步的数据流程图.

根据数据流程图可画出任务图,其规则如下:

(1)每个结点表示一个任务.

(2)针对每一根数据流程线,判断一下该线两个端点所表示的任务之间的关系,如果这两个任务可并行执行,则它们之间用实线表示,否则用虚线表示.

(3)调整,经过转换之后得到的任务图只是初步的,需要进一步的调整.首先把几个仅由虚线串连的任务组合成一个大任务;其次若二个任务间通信量很大,则尽量也把二者合成一个大任务;最后若任务数很多,则把几个粒度较小和功能相关的任务合为一个大任务.

(4)若得到的任务图并行度不高,则细化数据粒度,拓展数据并行性.

总之,尽量使任务图的并行性拓展彻底、功能明确、粒度适当、结构比较规则.

1.3 直接设计法

直接设计法不同于转换分析法,它是根据问题的功能描述,直接分解问题,拓展出问题的并行性.其设计过程如下:

首先把任务图画成只有一个任务的抽象图,并标志任务的输入、输出和功能,这也就是算法的输入、输出和功能.而后分解该任务,把该任务的功能分成几个并行或串行执行的子功能,每个子功能说明为一个新任务,如两任务之间是串行关系,则用有向虚线表示,如图3

(a)表示依次执行 1 号、2 号、……、n 号任务;两任务之间如果是并行的关系,则用有向实线表示,如图 3(b) 表示 1～n 号任务并行执行.各个任务的功能要描述清楚,各根通信线的内容也要详细说明.第一步分解后形成的任务图基本有如图 3 所示的 4 种情况.

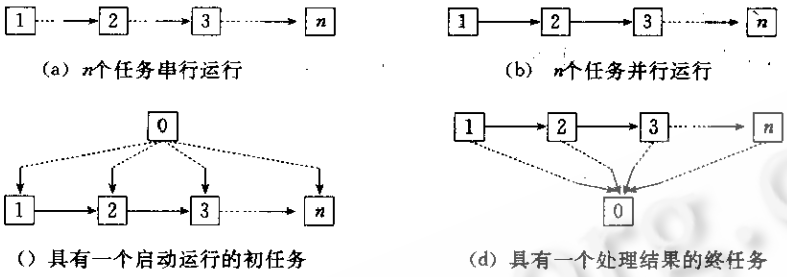

图3 第一步分解后形成的4种基本任务图

若第一步分解后产生的任务功能仍较复杂或功能不复杂但内含并行性,则继续分解,如图 4 所示,需要标明新的任务功能及新的通信线内容.

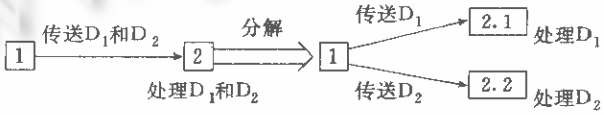

图4 任务分解图

在画任务图的过程中,要注意:一是任务功能要明确;二是初始分解时,通信线的内容要标上主要的数据,次要的数据可以暂时忽略,在细化过程中,自然会考虑到它;三是若通信内容初始不明,可暂时不标,待细化后,自然会明了.

按照以上的方法细化到一定程度后,即得到初步的任务图.

具体设计时,每一次分解都要画一张新任务图,并保存好原来的任务图,各张任务图以先后次序标上序号,标志任务图的细化过程.

由于问题的复杂性,任务图可能需要反复修改,往往细化到一定程度时,才发现原来的想法是错的或不合适,这时就得自下而上,重新考虑任务图的设计,有时候可能需要经过几次反复才能成功.

上述两种方法都是采取逐步求精的方法,自顶而下分解问题的复杂性,得到一张表示算法的任务图,但它们还是有一些不同点的,这主要是:

(1)转换分析法拓展并行性比较彻底,但形成任务图的功能比较含糊,而直接分析法则相反.

(2)转换分析法适合于分解问题的数据并行性,直接分析法适合于分解问题的功能并行性.在转换分析法中,只要把作为基本处理单位的数据粒度细化,画出对应于基本数据单元的数据流程图,就能看出是否存在数据并行性.

2 任务流程图

如同模块图对于顺序程序设计具体编程缺乏辅助一样,任务图也是如此,对于每个任务

图,还需要设计各个任务的流程图以指导具体编程.

任务图中的任务有两类:一类顺序性任务,任务内没有并行性,对于这一类任务,若结构复杂,可按结构化串行程序来设计.另一类是并行性任务,任务内有小粒度且少量的并行性,这一类粒度不大,可直接画流程图.

有时,任务图看来很合适,但到画任务流程图编程时才发觉情况并非如此,这时就要修改任务图,有时甚至需要修改整个任务图.一旦发觉不合适之处,就要审查整个任务图,任务图的仔细、良好计划对程序设计的质量起着十分重要的作用.

对于串行程序,任务流程图用串行结构、循环结构和选择结构就可表示;而对于并行程序,由于任务内还有并行性,任务流程图一般不只是这些,视具体语言而定.如OCCAM语言还提供PAR(并行)结构,ALT(备择)结构等.

任务流程图的每一种结构都应有一种表示图.以OCCAM语言为例,有串行、循环、选择并行和备择等5种结构,分别如图5—9所示,为了使程序设计者不产生混淆,对于前3种结构,我们仍采用串行程序中的结构图来描述.在图5—9中◇中的条件可以是复合条件;□中的指令可以是多条指令(即子程序).

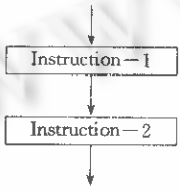

图5 标准的串行结构

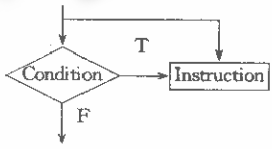

图6 标准的循环结构

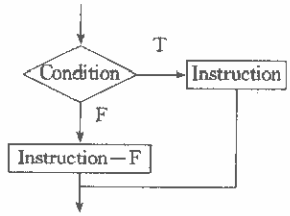

图7 标准的选择结构

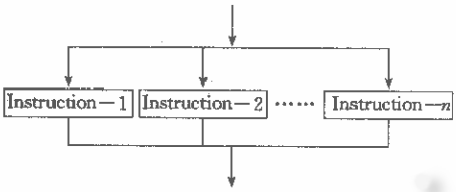

图8 标准的并行结构

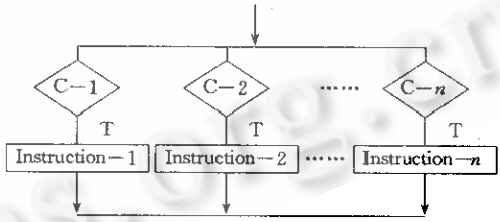

图9 标准的备择结构

3 重构与映射

任务图设计完成后,根据并行算法选择一个合适的拓扑结构使之匹配,然后把算法映射到结构上,最后编程、调试、结束.本文只讨论完成任务图的设计,余下问题将另文探讨.

4 实 例

以上探讨了基于任务图的并行程序设计方法,我们曾根据这个方法在多个Transputer并行计算机系统(NCD—1)上开发了若干并行程序.

• 环境

我们开发的程序是基于以IBM—PC为宿主机的多个Transputer并行计算机系统

NCD－1(图 10),Transputer 是英国 Inmos 公司研制的一种面向 OCCAM 语言的新型积木式单片计算机,我们采用的是 TsB94－9 板,上面有 9 个 Transputer(T414),其中有一个专门用作控制其余 8 个,称为 Master Transputer,其余的称为 Slave Transputer, Master Transputer 通过一个开关网络与其余的 Slave Transputer 相连, 开关网络是可编程的,故 Master Transputer 可根据需要设置 Slave 之间的互连网络,以得到所需的拓扑结构,但每个 Transputer 只有 4 条有向通信链路,故不是所有拓扑结构都能构作. 采用的语言有 OCCAM并行 C 和并行 Fortran.

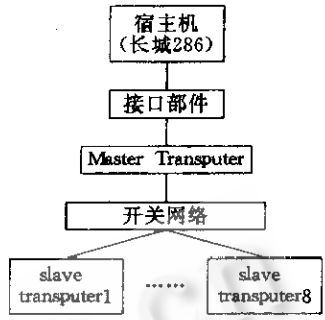

图10 并行计算机系统NCD－1

· 实例

在 NCD－1 系统上开发了几十个并行程序,下面以解二维 Possion 方程为例,说明本方法的思路.

$$\triangle U=2(3x+x^2+y^2) \qquad \text{于 } \Omega \text{ 内}$$

$$U|_P=x^2(x+y)$$

二维 Possion 方程第一边值问题的精确解是 $u=x^2(x+y^2)$,其中 $\Omega=\{(x,y)|0<x,\ y<2\}$, P 为 Ω 的边界. 采用步长为 h 的 5 点差分格式,记:$f(x,y)=2h^2(3x+x^2+y^2)$,得到上面的差分方程:

$$U_{i+1,j}+U_{i-1,j}+U_{i,j+1}+U_{i,j-1}-4U_{i,j}=f_{i,j} \qquad i,j=1,2\cdots\cdots,n-1;n=2/h$$

设 A 为以 h 来划分 Ω 而得的网格阵列,算法是对 A 操作,对于 1 个 Transputer,任务图为图 11(图中 ε 为对计算精度的要求)不需进一步细化. 对于 2 个 Transputer,就要进一步细化任务图. 图 11 中任务不存在功能并行性,因而可尝试拓展数据并行性,为此细化数据粒度,即把 A 分为两部分,对任务计算分析表明,A 的两部分能并行运行,即存在数据并行性,任务图细化如图 12 所示. 对于 4 个、8 个 Transputer 也作类似的进一步细化. 运算结果如表 1 (表中 S 为加速,E 为并行计算效率).

图11 解Possion方程初始任务图

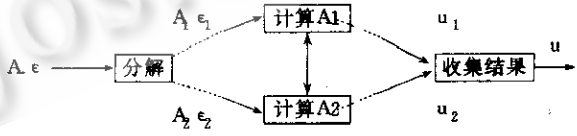

图12 解Possion方程第二级任务图

表 1 解 Possion 方程运算结果

(h=2/17 ε=|e－7)

算法	Possion1	Possion2	Possion4	Possion8
处理机数	1	2	4	8
T(秒)	95	49	26	15
S	1	1.93	3.65	6.33
E	1	0.96	0.91	0.79

6 结束语

本文提出的并行程序设计方法是初步的,不够完善,有待进一步发展.本方法中的算法是用任务图来表示的,用任务图表示算法有下列局限:

(1)动态并行算法难于表示,用任务图无法表示动态产生的进程.

(2)分段并行算法不能表示清楚,任务图只是表示运行的任务及其通信,而没有表示在某一时刻是哪些任务和哪些通信线在起作用,所以对于分段执行的并行算法就不能表示其阶段性.

(3)任务图中有些任务是互斥的(即算法每次运行,在这些任务中只可能有一个任务投入运行,至于是哪一个任务,则要根据算法的输入决定),这些任务应划分在一起,影射到同一个处理机,以提高处理机利用率.但任务图却没有表示出哪些任务是互斥的,对于通信也是如此.

对于通过设计数据流程图和功能分解来得到任务图,还只是尝试,这两种方法容易分解出问题的功能并行性.对于数据并行性,如用直接分析法开发有些不太自然.

尽管本文提出的方法有一些局限性,但总的说来,给设计者在可重构系统上设计并行程序提供了一套有条理的方法.它思路清晰,比较有效.

参考文献

1 Jiang H, Bhuyan L N, Ghosal D. Approximate analysis of multiprocessing task graphs. Proceedings of the 1990 International Conference on Parallel Processing, 1990(3): 228—235.

A METHOD OF PARALLEL PROGRAM DESIGN BASED ON TASK GRAPH(Ⅰ)——DESIGN OF TASK GRAPH

Zhang Defu Wu Qiaoquan

(*Department of Computer Science, Nanjing University, Nanjing* 210093)

Abstract This paper presents a method of parallel program design on task graph. At first, the authors can produce a dataflow chart by analyzing a given problem to get the task graph, i.e., a representation of the parallel algorithm for the problem, and then, choose the topologic structure according the task graph, at last parallel program is written in parallel programming languages. This method is lucid and most of the parallel programs designed by the method are of high quality.

Key words Task graph, method of parallel program design, parallel algorithm.