

基于 PVM 的并程序开发环境研究

熊泽时

(广州大学 华软软件学院, 广东 广州 510990)

摘要:开发并程序要比开发单机串行程序更难。PVM 开发环境是应用比较广的环境之一,适合于开发粗粒度的工程科学计算并程序,而这些工程计算问题一般是一些数值计算问题的集合。编写这些数值计算并程序有一定的难度和复杂度,并且现在没有很好支持开发 PVM 并程序的成熟开发环境。针对这个问题,构造一个基于 PVM 的并程序开发环境。开发环境包括一个并行算法库和一个嵌入到 Visual Studio 的可视化程序开发插件。通过开发平台进行并程序开发将更加简单、高效。

关键词:网络并行计算;PVM;并程序设计;算法库

中图分类号:TP302

文献标识码:A

文章编号:1673-629X(2010)05-0100-04

The Research of Parallel Programs Environment Based on PVM

XIONG Ze-shi

(South China Institute of Software Engineering, Guangzhou University, Guangzhou 510990, China)

Abstract: Developing parallel programs is more difficult than sequential programs. The PVM developing environment is popular in developing parallel programs. It is suitable for developing parallel program of engineering problem. These engineering problems regularly are the numerical computation problems. Aiming at the difficulties of the people always encounter when making parallel programs, a parallel program developing platform is presented. Platform includes a parallel algorithm library and an add-in module in the Visual Studio. It is easier and more efficient to program in the platform.

Key words: network parallel computing; PVM; parallel programming; algorithms library

0 引言

随着高性能应用需求的迅猛发展,单台高性能计算机已经不能胜任一些超大规模应用问题的解决。随着网络的发展和分布式计算的提出,通过多机互连的并行计算来解决大型的计算问题已成为许多人的首要选择^[1]。

伴随着网络和分布式并行技术的发展,出现了一些支撑并行计算的系统平台(如 PVM, MPI 等),这些平台本身都具备一系列的 API 接口,提供给开发人员调用,用以编写针对各自工程领域的并程序和解决一部分实际问题。但是它们的共同特点是使用难度大,对开发人员的要求较高。开发人员需要对这些平台系统非常熟悉,才能用其进行并程序的开发,加大了并行开发的难度^[2]。

为简化并程序开发的难度,构建了一个网络并程序开发平台。平台是在 PVM 消息通信库的基础

上构建的,对 PVM 的底层编程进行了简化。由于基于 PVM 的网络并行计算问题涉及的主要是数值计算问题,因此平台也主要研究数值并行计算的问题。

1 Windows 下基于 PVM 的并程序设计

网络并行计算是利用高速网络将一组计算机按某种结构连接起来,实现高效并行处理的系统。它可以是同构,也可以是异构的,计算机数量一般为几个至几十个。配以相应的支持软件,用户就可以将这组计算机当作一台并行计算机系统来使用^[3]。

PVM 是属于基于消息传递机制的并行虚拟机。基于 PVM 的并程序设计一般采用 Master/Slave 模式^[3]。它的并行计算结构由一个主处理机 Master 和若干从处理机 Slave 组成,Master 维持全局数据结构并负责任务划分,负责用户界面,接受任务,启动计算和回收结果。而每个 Slave 负责完成子任务的计算,包括局部初始化,并行计算和处理机间的数据通信,并把结果返回到 Master。它们都是操作系统的进程。进程一旦登录 PVM,就成为受 PVM 控制的任务。每个处理

收稿日期:2009-09-07;修回日期:2009-12-15

基金项目:广西自然科学基金资助项目(0229009)

作者简介:熊泽时(1981-),男,硕士,研究方向为网络并行计算。

机负责计算分配给它的任务。主处理机和从处理机间的关系如图1所示^[4]。

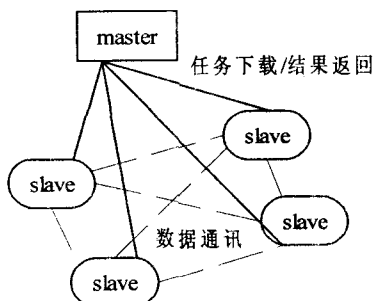


图1 并行计算结构中处理机间的关系示意图

Master依据问题规模和并行程度,启动若干个Slave,开始计算,并对Slave进行初始化。Slave从初始化信息中识别自己,接收保存任务数据,再进行计算。每次计算后,各Slave需要相互交换边界数据,以保证下一次计算使用新值。最后,Slave通知Master计算完成,并检验结果。Master回收结果并输出。

2 一般数值并行算法模型

2.1 并行程序的模块结构

并行程序的编写有别于串行程序,特别是要处理任务划分、任务分配以及任务通信同步等等。下面先讨论并行程序中的几个基本概念^[4]。

(1) 任务划分:将一个程序划分成若干个可以全部或部分并行执行的部分。任务划分可以根据控制流进行,也可根据数据流进行。一个任务可以是一个子程序(它又可包含其他子程序调用)、一段语句或一个循环中的一个或一组迭代。划分任务要力求并行性最大,任务间的相关性最小,任务的粒度适当。

(2) 任务调度:将各个任务分配到不同的处理机上去执行称之为调度。任务调度要和数据分布对准,即把任务分配到它需要的数据所在的处理机上去执行,以减少处理机间通讯。

(3) 任务同步:同步是协调各并行任务运行的一种方法,它保证相互作用的各任务所要求的条件得到满足。

(4) 任务通讯:一个任务将一批数据传送给另一个任务称之为通讯。一般来说,发送数据的任务计算这些数据的值,接收数据的任务使用这些数据的值。为了保证正确性,通讯必须在同步点进行,在分布主存并行系统上通讯开销是一个值得重视的问题。通讯的模式有一对一、一对多和多对多。

采用Master/Slave模式的并行程序有其共同的程序流程和模块。根据对一般的数值并行程序的分析,我们把Master程序分成四个模块:自定义变量模块,

数据发送模块,数据接收模块,主控制模块。各模块相互关联,但功能分离,每个模块实现相应的功能。Slave程序跟Master程序要进行同步通信,因此Slave中的模块要跟Master中的模块相对应,同样分成四个模块:自定义变量模块,数据接收模块,数据处理模块,数据发送模块。每个模块间有顺序和依赖的关系。在Master程序中,数据发送前必须要进行定义和初始化,因此自定义变量模块要在数据发送模块前,这两个模块间就有一个顺序关系。同样的,在Slave程序中,在进行数据处理前,必须能够接收到Master发送过来的计算数据。模块间的关系如图2所示。

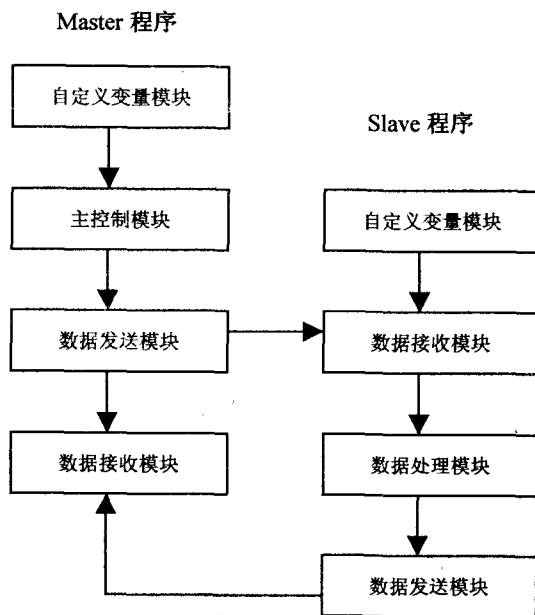


图2 并行程序流程和模块间的关系图

2.2 并行算子

基于PVM的网络并行计算解决的往往是工程中的大型科学计算问题。而这些大型计算一般情况下是一些数值计算问题的集合,如定积分计算、矩阵运算、线性方程组求解等。将大型计算问题分解成一个个的子问题,称这种方法为分治策略。

在网络并行计算平台中,把一个逻辑上不能再分的原子问题称为一个并行算子。例如上面提到的计算定积分,两矩阵相乘等都是并行算子。算子之间可以有以下几种关系:

(1) 包含关系。一个算子是另外一个算子的一部分。

(2) 依赖关系。一个算子的运算依赖于另外一个算子。

(3) 空关系。算子之间不存在直接的关系。

2.3 任务分配

一般的任务分配方法有两种:静态任务分配和动

态任务分配^[5]。静态分配方法是在程序运行前就已经决定好任务的划分。二次均分法是最常用的静态任务分配方法。这种方法是将所有的总任务首先按照参与计算的机器数量进行均分,然后将不能整除的任务数再一次分配给各台处理机,这样每个结点最多只相差一个子任务。

以下是按照二次均分法进行任务分配的源代码:

```
loop = TotalTask / ProcNum;
//将总任务数按处理机个数均分
for(i=0; i < ProcNum; i++)
loops[i] = loop;
//为每台 Slave 机器分配相等的任务数,即处理点数
for(i=0; i < TotalTask - TotalTask / ProcNum * ProcNum; i++)
loops[i]++;
//将不能整除的各点再一次分配给处理机,保证各台处理机工作量大致一样
pvm_initsend(PvmDataDefault);
//初始化发送缓冲区
pvm_pkint(&ProcNum, 1, 1);
//打包变量(处理机的个数)
pvm_pkint(tids, ProcNum, 1);
//打包变量(为 Slave 机器所分配的 id 数组)
pvm_pkint(loops, ProcNum, 1);
//打包变量(各台 Slave 机器所需承担的任务数数组)
pvm_mcast(tids, ProcNum, msg_spawn);
//向 Slave 机器发送以上打包的数据
```

这种方法的思想和实现代码都比较简单,适用于一般并行计算的环境。算法库中是用静态的二次均分方法实现的。

3 数值计算并行算法库

通过对并行程序流程的分析,可以知道每种数值计算并行程序都有一定的编写模式,并且有相同的程序流程和模块。因此,把相同类型的并行算子封装成一个标准算法组件^[6]。算法组件将并行程序代码封装起来,只留出足够的接口让外界调用即可。

基于消息传递的 PVM 并行程序要通过显式的通信来进行,Master 和 Slave 之间要同步通信,这就决定了组件实现也必须模块化。组件的实现可以考虑两种方案:一种是以函数库的形式添加到 PVM 的开发环境中,另一种是通过后台代码生成并插入的形式添加到并行程序中。根据对这两种方案的分析和实验,最后决定将组件以函数库的形式添加到 PVM 的开发环境中,构造一个并行算法库,这样开发并行程序更灵活,组件的扩展性更好。

下面介绍一个定积分的计算的组件。定积分的计算只要由四个参数来确定:上限、下限、步长和多项式函数^[7]。需要构造两个主要的 Master 中的函数集和 Slave 中的函数集。按照上面对并行算子的模块化分析,每个模块的功能需要相应的函数来完成。这其中还需要解决并行程序中任务通信的同步问题。

在 Master 程序中,调用 M_Integration() 函数即可完成定积分的计算,函数原型如下所示:

```
double M_Integration(double begin, double end, int step, char *s);
```

其中 begin 为上限, end 为下限, step 为步数, s 为函数字符串数组。

在上文模块结构分析时将 Master 程序分成了 4 个模块,每个模块完成特定的功能。因此, M_Integration() 函数需要再调用其他的功能函数。但是在应用上 M_Integration() 函数对用户是透明的。

通过函数的重载封装了通信变量的接收和发送。在定积分计算中需要发送初始化信息以及计算数据。初始化信息包括进程号 tids 数组和进程数量等,这些代码在程序框架生成的时候完成,不在 M_Integration() 函数中处理。定积分计算函数只要发送要处理的计算数据即可。这里包括上下限和步长,以及任务分配情况。

同样在 Slave 程序中,可以调用相应的函数 S_Integration()。函数原型如下所示:

```
double S_Integration(double ibegin, double iend, int istep, char *s);
```

其中 ibegin、iend 和 istep 为子任务的上限、下限和步数, s 为函数字符串数组。

S_Integration() 函数调用相应的数据接收函数和数据处理函数,计算结束后再调用数据发送函数将结果回送到 Master。数据接收函数和发送函数同样用重载将 PVM 通信原语封装起来。

S_Integration() 函数还包括数据处理部分,这里涉及词法分析以及任务分配的处理。词法分析通过对函数字符串数组进行处理;而任务分配的处理,在系统内部实现的是静态的二次均分方法。

4 并行程序框架的生成

编写基于 PVM 的并行程序需要做一些繁琐的配置,如每次编写程序都要连接 PVM 的库文件和头文件等,以及将生成的 Slave 可执行文件复制到 PVM 的 bin 目录下。Visual C++ 是一个强大的可视化开发平台,它有一个自定义的向导功能,提供给用户设置一个

自定义的程序框架^[8]。因此,总结了数值并行程序的特点,并根据这些特点给出了一个生成并行程序框架的向导,帮助程序员快速建立并行程序的基本框架。

Win32 环境下的 PVM 并行程序主要是采用 Master/Slave 模式的,向导根据这种模式生成框架。向导有两个:Master 程序生成向导和 Slave 程序生成向导。

根据向导的步进对话框选择相应的内容(通过向导对话框资源),就可以生成相应的程序框架。向导只有一个步进对话框,在向导的新建对话框中输入文件名以及文件的定位后,有一个计算类型的步进对话框,包括计算定积分、矩阵并行计算、线性方程组的求解以及一个空文档(如图3所示)。

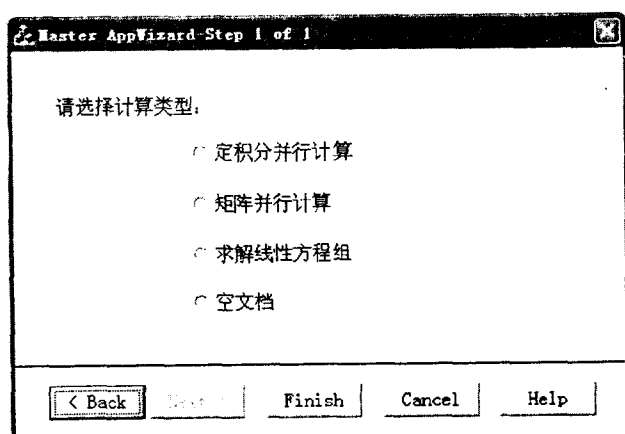


图3 并行程序框架生成向导

向导会根据用户的定义自动生成 Master 和 Slave 的头文件以及源程序文件^[9]。文件的内容则根据选择的计算类型不同而不一样,例如一个计算定积分的并行程序框架内是一个计算定积分的并行程序,并且备有详细的代码注释,只要修改函数多项式、上下限以及步长就可以完成一个计算定积分的并行程序。其他类型也是一个完整的并行计算程序。一般情况下,空文档是主要的应用。空文档中主要包括以下内容:

(1)包含 PVM 平台提供的编程接口的头文件以及需要的所有库文件连接。

(2)主控制模块中的基本初始化代码,以及调用相应的原语将必要的初始化信息广播出去。

(3)Master 和 Slave 文件中终止调用进程和终止进程的通信原语代码。

空文档是一个什么也不做的并行程序,用户需要调用算法库来进行程序设计。

5 可视化开发环境

平台的可视化部分作为一个 Visual Studio 插件,以 Add-in 的形式嵌入到 Developer Studio 当中^[9]。这样,开发人员不但可以在熟悉的 VC 环境下继续开发

并行软件,并且开发人员还可以继续使用 Visual Studio 环境中自带的断点调试,查看内存等功能。

通过平台的对话框可以进行可视化程序设计。开发人员可以通过对话框添加发送变量和接收变量,平台底层完成变量定义和通信代码的生成,并且插入到源程序的适当位置。向导也可以通过对话框添加一个或多个并行算子(如图4所示)。同样平台底层自动完成并行代码的生成和插入。功能的实现主要有代码的生成和代码插入两个部分。

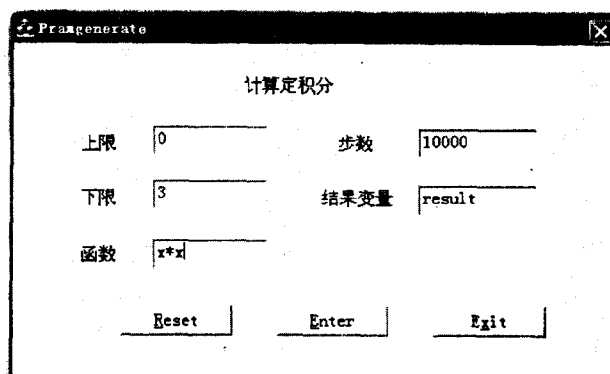


图4 添加并行算子对话框

6 结束语

并行程序的开发与单机串行程序开发相比,具有一定的特殊性和复杂性。针对编写这些数值计算并行程序有一定的难度,构造了一个开发并行数值计算的开发平台,并且编写了一个数值并行算法库,算法库中包括了一些常用的数值并行算法,并且还可以根据需要对算法库进行扩展。通过开发平台来进行并行程序设计大大降低了并行开发的难度,并且程序开发更加高效。

参考文献:

- [1] Wilkinson B, Allen M. 并行程序设计[M]. 陆鑫达,等译. 北京:机械工业出版社,2001.
- [2] 李代平,罗寿文,张信一,等. 一个网络并行计算新平台[J]. 计算机工程与设计,2005,26(1):24-26.
- [3] 孙家昶,张林波,迟学斌,等. 网络并行计算与分布式编程环境[M]. 北京:科学出版社,1996.
- [4] 张信一,李代平,章文. 基于 Win32 平台上的 PVM 并行程序设计[J]. 计算机应用研究,2004(5):102-104.
- [5] 熊泽时,李代平. 一种有效的动态任务分配方法[J]. 计算机技术与发展 2007,17(4):175-177.
- [6] 陆嘉,温冬婵,王鼎兴,等. 一个基于机群系统的面向对象并行程序开发环境的研究与实现[J]. 计算机研究与发展,1999(7):836-841.
- [7] 孙世新,卢光辉,张艳,等. 并行算法及其应用[M]. 北

(下转第107页)

```
{“Compare”,2,“int,int”,“int”}
```

这个元素说明 main 函数需要调用一个 Compare 模板,它需要传递两个整型参数。这个程序的生成过程如下:

(1)在输出文件中插入基础模板。

(2)在 main 函数前插入经过修剪的 Compare 模板的定义;在 main 函数开始处定义两个整型参数变量及一个返回整型变量。

(3)在 main 函数中依据控制语句顺序插入这两个变量的输入模板、Compare 模板调用及输出模板,并将定义的两个整型量作为参数传递给 Compare 模板。

3 系统测试结果

输入

有其他的行^[4]

输出

Main. c:

```
#include“GetLine. h”
```

```
#include <stdio. h>
```

```
void main()
```

```
{
```

```
int len;
```

```
char line[MAXLINE];
```

```
len = GetLine(line, MAXLINE)
```

```
}
```

GetLine. h:

```
/*
```

```
* GetLine: read a line into s, return length */
```

```
#define MAXLINE 1000
```

```
int GetLine(char para0[],int para1)
```

```
{
```

```
int c, i;
```

```
for (i=0; i < para1-1 && (c=getchar())!= EOF && c!= '\n'; ++i)
```

```
para0[i] = c;
```

```
if (c == '\n')
```

```
{
```

```
para0[i] = c;
```

```
++i;
```

```
}
```

```
para0[i] = '\0';
```

```
return i;
```

```
}
```

目前测试的模块模板库规模比较小,在有效的测试用例中,绝大部分比较规范的输入语句都能得到和语句意思相符合的程序。

4 结束语

文中实现了自动代码的生成系统,但仍然存在很大的可改进之处:

一是目前只能使用模板库中已有模板来生成程序,需要添加大量的模板来完善;同时当前的模板代码是固定的,如果语义解释的内容有变动,模板代码也需要进行大量的改变。如果能实现通过代码生成模板,这将大大减少模板的开发时间。

二是 LCGS 应用还只是针对一些简单的模型,不够完善,可以扩大代码生成范围,比如可以生成整个应用程序框架,实现真正意义上模型自动向代码转换,达到自动部署和运行的目的。

参考文献:

- [1] Burginsky F, Finnie M, Yu P. Automatic Code Generation from Design Patterns[J]. IBM Systems Journal, 1996, 35 (2):151-171.
- [2] 李家治,陈永明. 机器理解汉语——实验 I[J]. 心理学报, 1982(1):32-43.
- [3] 马庆株. 汉语语义语法范畴问题[M]. 北京:北京语言文化大学出版社,1998.
- [4] 陈祖荣. 浅谈动词的分类问题[J]. 四川师范学院学报:哲学社会科学版,1995(1):56-58.
- [5] 马庆株. 汉语动词和动词性结构(一编)[M]. 北京:北京大学出版社,2005.
- [6] 钱进,王希杰. 词义性别原型与构词模式[J]. 江西社会科学,2004(9):149-153.
- [7] Sakurai T, Shibagaki T, Shinbori T. An Automatic Programming System Based on Modular Integrated - concept Architecture[C]//Proc. of IECON'90. California, USA: Industrial Electronics Society, 1990:1303-1308.
- [8] Kernighan B W, Ritchie D M. The C Programming Language [M]. [s.l.]:Prentice - Hall, 1988.

(上接第 103 页)

京:机械工业出版社,2005.

- [8] Shepherd, G. Visual C++ .Net 技术内幕[M]. 第 6 版. 北京:清华大学出版社,2004.

- [9] 张信一,李代平,罗伟刚. 并行程序开发平台的可视化实现[J]. 计算机应用研究,2004(11):266-269.