云南大学学报（自然科学版），2007，29（3）：251~255　　CN 53-1045/N　ISSN 0258-7971
Journal of Yunnan University

基于 PVM 的并行编程技术及其对温度场的可视化*

周　亚，蒋慕蓉
（云南大学 信息学院；云南 昆明　650091）

摘要：首先介绍了基于 PVM 分布式平台的并行编程技术以及科学计算可视化的基本概况、数据场可视化的流程，接着在 Visual C++ 6.0 可视化编程环境下对温度场可视化应用进行了并行程序设计.

关键词：并行；可视化；PVM
中图分类号：TP 39　　文献标识码：A　　文章编号：0258-7971(2007)03-0251-05

1　预备知识

并行计算机可以是 1 台具有多个内部处理器的单计算机，也可以是由多台互联的计算机构成一个整体的高性能计算平台. 就目前来说，主要有 2 种基本的并行计算机类型：① 共享存储器多处理机；② 分布式存储器多计算机. 由多台完整的计算机通过互联网络连接，处理机间通过传递消息来实现计算通信而构成的一个多处理机系统属第 2 种类型，通常被称为消息传递多处理机（message-passing multiprocessor），或简称为多计算机（multi-computer）[1].

多计算机系统中常用的编程模型是消息传递模型，消息传递模型通过标准库函数使得驻留在不同节点上的进程可以通过网络传递消息相互通信. MPI 和 PVM 是最常用的 2 种消息传递标准库.

1.1　PVM 并行计算平台　PVM 是并行虚拟机（Parallel Virtual Machine）的缩写. PVM 系统由两大部分组成：守护进程和 PVM 接口例行程序库. 守护进程，缩写为 pvmd，它驻留在构成虚拟机的所有计算机上. 守护进程是在后台执行的一种程序，在需要时随时准备完成一种操作. PVM 接口例行程序库，它包含各种功能完备的原语，这些原语主要用于协调应用任务. 该程序库包括那些用于消息传递、创建进程、协调任务以及修改虚拟机等用户可调用例行程序[2,3].

PVM 计算模型是基于由若干个任务组成的应用，所有任务通过标准接口例行程序库来访问 PVM 资源. PVM 虚拟机结构如图 1 所示.

从图 1 中我们可以看出守护进程 pvmd 构成了虚拟机的核心. 调用 PVM 库函数的用户应用程序称为 PVM 任务，它们是完成并行应用实际工作的主体，当它们被加载到虚拟机中后，由 PVM 管理并被实际分到某台具体机器中运行，位于不同机器中的任务真正并行地执行. PVM 任务之间，PVM 任务和 PVMD 之间以及 PVMD 之间都相互联系，存在着消息通讯（包括 PVM 系统内控制消息和 PVM 任务之间传递的数据）.

1.2　科学计算可视化　科学计算可视化是 20 世纪 80 年代后期提出的一个新的研究领域. 科学计算可视化（Visualization in Scientific Computing）这一术语是在 1987 年由美国国家科学基金会（NSF）召开的“科学计算可视化研讨会”中提出的[2]. 它涉及计算机图形学、图像处理、计算机辅助设计、计算机视觉及人机交互等多个领域. 随着各应用领域中数值模拟及测量技术的发展，数据场在规模上急剧扩大，仅仅依靠计算机本身存储空间的扩大、计算能力的提高，尚不足以解决这个问题. 将并行处理技术引入可视化算法，利用并行体系结构实现对科学计算过程和计算结果的可视化，成为近年来科学计算可视化领域的一个非常活跃的研究方向.

* 收稿日期：2006-06-01
基金项目：云南省教育厅基础研究基金资助项目（5Y05670）；云南大学理（工）校级科研基金资助项目（2002T006XX）.
作者简介：周　亚（1981-　），男，湖北人，硕士生，主要从事网格计算、科学计算可视化方面的研究.
通讯作者：蒋慕蓉（1963-），女，湖南人，副教授，主要从事分布式计算、图像处理等方面的研究.

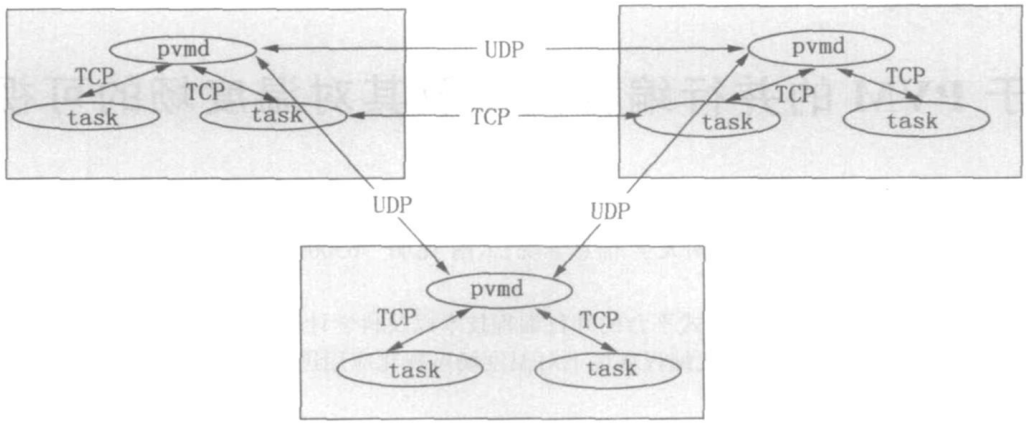

图 1 **PVM** 虚拟机运行结构示意图

Fig. 1 The working structure of PVM

1.3 数据场可视化 在科学计算中,研究对象的特性往往是用方程组来描述的. 通常情况下,问题的解析解不易得到,我们常常借助数值计算的方法来求得方程组的数值解. 为此,我们需要对问题空间进行离散化处理,再利用计算机数值求解的方法求出各离散单元的函数值. 数据场可视化即是指在这些离散单元函数值的基础上通过映射、图形图像的绘制而将整个数据场显示出来的一系列过程. 其基本流程如图 2 所示[2].

图 2 数据场可视化流程

Fig. 2 The procedure of rendering of data sets

2 Windows 下 PVM 并行编程环境的安装与配置

PVM 在 Windows 下的版本可以让用户在基于 Windows 平台的多计算机系统中开发并行程序. PVM 需要在一种编译环境中运行, Visual C^{++} 6.0 是其编译环境的一种极好选择,我们是只要进行必要的设置,就可以直接在 Visual C^{++} 6.0 环境中编写、编译和运行 PVM 程序. PVM 软件开发包 pvm3.4.3 和远程调用服务 rshdnt- eval 软件包的安装可参看文献[4,5],还需对 Visual C^{++} 6.0 环境进行必要的设置方可对 PVM 程序进行编译与运行[6]. 图 3 为 PVM 虚拟机配置成功的控制台显示结果.

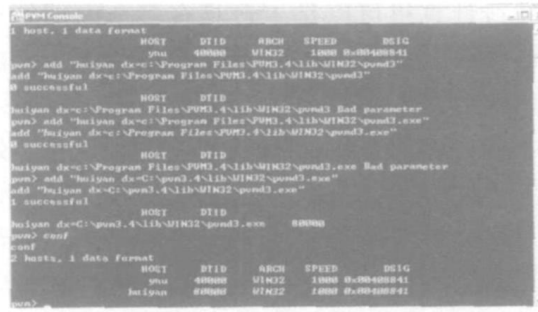

图 3 由 2 台计算机构成的 PVM 并行虚拟机配置成功显示

Fig. 3 The display of PVM constructed by two PCs which configured successfully

3 温度场可视化

为了验证在基于 Windows 平台的局域网中配置的 PVM 并行平台下开发并行程序的可行性及其并行可视化效率,本文设计了一个简单的科学计算与可视化相耦合的温度场并行可视化实例.

先将问题描述如下:

工程实际中要解决物体内部的传热问题一般最终归结为求物体内温度场的分布,若我们可以将温度场可视化出来,这将为科研人员工程分析找出规律提供极大方便.

二维热传导方程为 $\frac{\partial u}{\partial t} = a^2 \nabla^2 u = a^2\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right)$[3]; 若我们考虑稳恒温度场,热传导过程中物体的温度趋于某种平衡状态,这时温度 u 已与时间 t 无关,即 $\frac{\partial u}{\partial t} = 0$,此时方程简化为拉普

拉斯方程(Laplace Equation): $\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0$.

边界条件假定为: $u|_\Gamma = f(x, y) = x + y$.

3.1 计算模型　该算法的基本思想是采用区域分割的方法对温度场实现并行可视化处理, 每个子进程实现对单个区域温度场的可视化, 将可视化结果收集到 master host, 合并计算结果即实现了温度场的并行可视化.

首先, 我们可以通过差分法对拉普拉斯方程同步迭代计算出离散温度场中各离散单元结点的函数值. 拉普拉斯方程同步迭代的差分格式为

$$U_{i,j} = \frac{U_{i+1,j} + U_{i,j+1} + U_{i-1,j} + U_{i,j-1}}{4}.$$

为简化计算过程, 假设温度场子区域的几何形状为方形, 分别进行网格划分如图 4 所示.

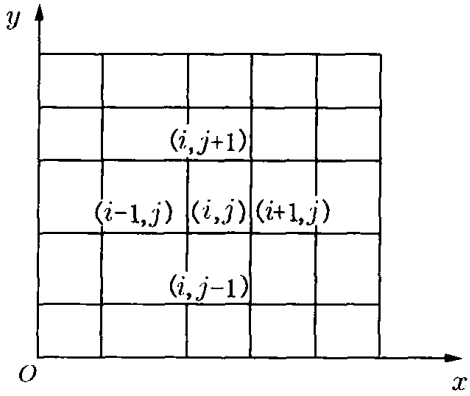

图 **4**　差分格式网格划分

Fig. 4 Lattice dividing on difference pattern

计算完温度场各离散单元结点的温度值后, 需对各子区域内温度场进行可视化映射. 其基本思想为将场值函数映射为颜色函数, 其映射函数的一般形式为

$$H_{i,j} = H(r(u_{ij}), g(u_{ij}), b(u_{ij})),$$

其中 $0 \leqslant r, g, b \leqslant 255$.

为了简化, 我们可以在温度场的范围内将各数据值按大小划分为不同的子类, 然后根据子类的数目预设同样数目的颜色值, 通过温度值到颜色值的匹配, 从而给各数据结点赋上相应的颜色值.

最后将各子进程的颜色值函数返回给主进程, 以实现对温度场可视化的绘制. 其算法流程如图 5 所示.

3.2 编程环境说明

硬件环境: 采用局域网中的 2 台 PC 机相连接组成;

软件环境: Windows 2k 操作系统, Winsock RSHD \ NT 远程调用软件, PVM3. 4. 3, VC++ 6. 0, OpenGL图形库.

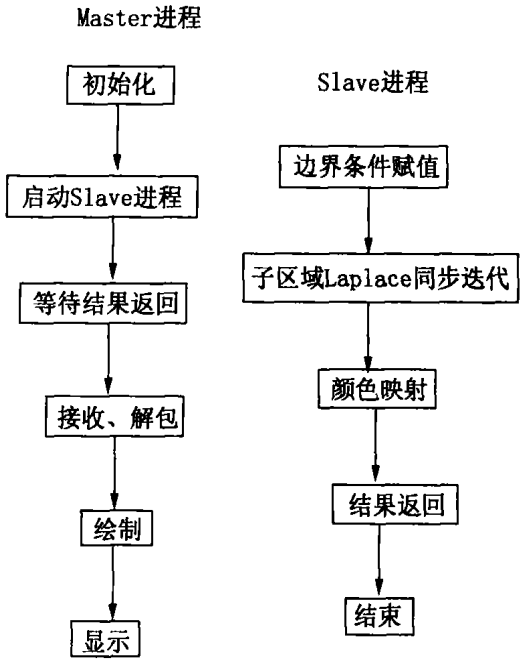

图 **5**　温度场可视化算法流程图

Fig. 5 The procedure of the visualization of the temperature set

3.3 实验步骤

(1) 主进程 Master 程序

Master 程序基于 VC++ 对话框框架, 在 COpenGL::OnPaint() 中显示图像, 主要步骤为:

第 1 步: 初始化 PVM 环境, 启动 2 个从进程;

```
mytid= pvm_mytid();// 将父进程注册进 PVM 系统中;
int num= pvm_spawn(SLAVE,(char 3 3) 0,0,"",2,tids);// 派生 2 个子任务;
```

第 2 步: 从 2 个从进程中接收数据, 并存放到 1 个二维数组中;

```
for(i= 0; i< 2; i++){
  pvm_recv(-1,msgtag= 5).
  pvm_upkint(&who,1,1);
  pvm_upkfloat(&RGB[who],1,1);//RGB[ ] 为为温度场颜色值数组;
}
```

第 3 步: 显示图像;

```
void COpenGL::DrawColorArea(H){
  //边界赋值;
```

```
glBegin (GL_POLYGON); //绘制多个彩色多边形,每个多边形代表一行数据场网格
//颜色的深浅代表温度值的大小
glColor3fv(c0);
glVertex2fv(p0);
    …
glEnd();
}
```

(2) 从进程 Slave 程序

Slave 程序设计为 Win32 Console 类型, 在 On-Computing() 函数中迭代计算拉普拉斯差分方程, 主要步骤为:

第 1 步: 进入 PVM;

```
pvm_mytid (); // 进入 PVM,获得从进程的 tid;
pvm parent (); // 获得主程序的 tid;
```

第 2 步: 迭代计算子区域内拉普拉斯差分方程, 设定迭代次数为一个常数 s;

```
while(step< s){
for(i= 1; i< m; i++)
  for(j= 1;j< n;j++)
   u[i][j]= 0.25*(u[i-1][j]+ u[i][j-1]+ u[i+1][j]+ u[i][j+1]);
}
```

第 3 步: 对数据场进行可视化映射;

```
switch(int(ceil(u[i][j])))
{
case U[0]: c[0]= c[i][j];
case U[1]: c[1]= c[i][j];
 …
}
```

第 4 步: 将颜色值返回给 Master 进程;

```
pvm_initsend (PvmDataDefault);
pvm_pkint (&me, 1, 1);
pvm_send (pvm_parent (), msgtag= 5).
```

3.4 结果显示及分析 上述实例在 PVM 平台下运行, 产生如图 6 所示试验结果. 下面对结果进行分析:

(1) 由于数据点的选取比较稀疏, 从而导致可视化图像的效果比较模糊, 颜色的变化不够均匀, 但还是大约可以看出温度场的分布情况;

(2) 区域分割后假设第 1 区域内的新边界条件为 $u[5]=0$, 第 2 区域内的新边界条件为 $u[0]=5$; 这种假设在线性边界条件下有一定的合理性, 但同样会引入一定的误差. 对于非线性边界条件, 其边界条件的假设将有一定的复杂性;

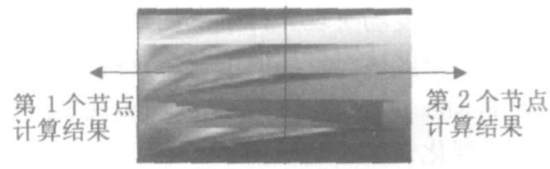

图 **6** 温度场可视化结果显示

Fig. 6 The result display of the visualization of the temperature set

(3) 通过对数据场划分而产生的并行可视化, 在一定程度上提高了可视化计算的效率. 但若能在子区域内实现拉普拉斯方程的并行算法, 将大大提高可视化的效率;

(4) 现就并行算法与串行算法在时间复杂度的对比作分析如下: 假设串行算法的时间复杂度为 $t_s(n)$, 并行平台的启动时间为 $t_{startup}$, 数据传输时间为 $t_{data}(n)$, 则并行算法的时间复杂度为

$$t_o(n)=t_s(n)/2+t_{startup}+t_{data}(n).$$

其加速比系数为

$$s(n)=\frac{t_s(n)}{t_p(n)}=\frac{t_s(n)}{t_s(n)/2+t_{startup}+t_{data}(n)},$$

只有当 $t_{startup}+t_{data}(n)<t_s(n)/2$ 时, 加速比系数 $s(n)>1$, 也即对于大规模数据场并行算法才能体现出较串行算法的更大的优越性. 为验证结论, 我们试着增大数据量 n, 当 n 递增时, 并行算法的加速比 $s(n)$ 是 n 的增函数, 其极限为 2, 此时并行算法较串行算法的效率将近提高了 1 倍;

(5) 若改变 PVM 平台中处理器数 p, 此时加速比

$$s(p)=\frac{t_s(p)}{t_p(p)}=\frac{t_s(p)}{t_s(n)/p+t_{startup}+t_{data}(n)},$$

其理想加速比为 $s(p)=p$.

由此可知, 通过增加处理器数 p, 可以更大程度地提高并行算法的效率. 对于算法复杂度较高的科学计算问题, 若该应用具有很高的可并行性且串行算法不能在有效的时间内获得问题解时, 我们可以

利用并行算法, 从而可能在有效时间内获得问题解.

4　结语及下一步工作展望

通过本实例的并行程序设计, 我们可以看出在 Windows 环境下基于 PVM 的并行编程很易实现科学计算以及可视化等方面的并行开发.

以上工作也为下一步的工作打下了一定的基础. 我们可以将现有科学计算以及可视化并行算法移植到网格环境中, 将科学计算可视化作为 OGSA 网格体系中的一种网格服务(Grid Service)[7], 从而广大了潜在的同类应用提供服务. 在这方面国内外已经有大量的相关研究, 如美国德克萨斯大学奥斯汀分校的计算可视化中心在 Globus 的基础上以网格服务的形式向用户提供并行可视化服务[8], 浙江大学 CAD & CG 国家重点实验室正在进行的面向网格可视化平台 GVis[9] 等都在网格可视化服务研究方面取得了很大的进展.

参考文献:

[1] Barry Wilkinson Michael Allen. 并行程序设计[M]. 陆鑫达, 等译. 北京: 机械工业出版社, 2002.

[2] 唐泽圣. 三维数据场可视化[M]. 北京: 清华大学出版社, 1999.

[3] 南京工学院数学教研组. 数学物理方程与特殊函数[M]. 北京: 高等教育出版社, 1982.

[4] PVM 编程指南[EB/OL]. [2005- 06- 30]. http://www.longen.org/L- R/detaill- r/PVMProgamming.html.

[5] Winsock RSHD- NT manual[EB/OL]. [2003- 05- 10]. http://www.denlcomp.com/rshdnt.html.

[6] 杨光亿. 基于 PVM 平台的并行编程技术及其在图像处理中的应用[J]. 计算机工程与科学, 2005, 27(9): 44-46.

[7] FOSTER I, KESSELMAN C. The physiology of the grid: an open grid services architecture for distributed systems integration[J/OL]. [2001- 03- 10]. http://www.globus.org/papers.html.

[8] Center for Computational Visualization of University of Texas at Austin. Grid enabled visualization[C]. NPACI All-hands Meeting, San Diego, 2003.

[9] 石教英, 赵友兵, 仇应俊, 等. 面向网格的可视化系统研究[J]. 计算机研究与发展, 2004, 41(12): 2231-2235.

PVM based parallel programming technology and its application on visualization of temperature set

ZHOU Ya, JIANG Mu-rong

(Department of Computer Science and Engineering, Yunnan University, Kunming 650091, China)

Abstract: It is firstly introduced a method of developing parallel programs based on PVM and then a generalization of Visualization in Scientific Computing and the procedure of rendering of data sets is given. Subsequently, the author designs a parallel program on visualization of temperature set in the VC++ 6.0 visualization integration environment.

Key words: Parallel programming; visualization; PVM

* *

(上接第 250 页)

Abstract: In allusion to the insufficient of the flexibility of the CSnd scalable characteristic of SIP protocol as a signal control protocol, synthetically using the CSCL principle technology, database technology and the multimedia communication technology, it was studied and designed a SIP-based CSCL model system. It was discussed the design of conversation model, the systemic structure of the CSCL system, the design of its resource layer, SIP server and SIP terminal.

Key words: computer-supported collaborative learning; SIP protocol; SIP server; SIP terminal