一种基于PETSc的热传导方程大规模并行求解策略

程汤培 王 群
(中国地质大学(北京)信息工程学院 北京 100083)

摘 要 提出了一种大规模热传导方程并行求解的策略,采用了分布式内存和压缩矩阵技术解决超大规模稀疏矩阵的存储及其计算,整合了多种Krylov子空间方法和预条件子技术来并行求解大规模线性方程组,基于面向对象设计实现了具体应用与算法的低耦合。在Linux机群系统上进行了性能测试,程序具有良好的加速比和计算性能。

关键词 热传导方程,偏微分方程组,有限差分法,并行算法
中图法分类号 TP301.6 **文献标识码** A

Parallel-computing Strategy for Large-scale Heat Equation Based on PETSc

CHENG Tang-pei WANG Qun
(School of Information Engineering, China University of Geosciences (Beijing), Beijing 100083, China)

Abstract A parallel-computing strategy was presented to solve the large-scale heat equations. The distributed memory and compressed matrices technology was adopted for both the process of storage and evaluation of large-scale sparse matrices. All kinds of Krylov subspace methods and preconditioners were introduced to assemble and solve the linear systems of equations. The code implementation of this strategy was written in high-level abstractions based on object-oriented technology which promotes code reuse, flexibility and helps to decouple issues of parallelism from algorithm choices. The experiments carried on Linux clusters demonstrate that this strategy has achieved desirable speedup and efficiency.

Keywords Heat equations, Partial differential equations, Finite difference methods, Parallel algorithm

1 引言

在工程应用中,热传导方程是非常重要的一类数学物理方程。构造高效并行求解热传导偏微分方程的方法一直是科学研究中的热点问题,并且已有了大量的研究。文献[1]研究了热传导方程有限差分区域分解算法,其方法是将求解区域分解成若干子区域,在子区域间的内界点上使用古典显式格式求解,在每个子区域内使用隐式格式求解。文献[2]采用了与文献[1]类似的思想,通过引进内界点将求解区域分裂成若干子区域,在子区域间内界点上采用较小时间步长的分组Saul'yev非对称格式计算,提高了经典的显式格式计算的稳定性。文献[3]中提出了一种改进的基于时间分解的实时并行算法,该算法基于欧拉系统对含时间微分的非稳态项进行分解,分别在不同尺度的粗细网格上迭代求解,实现了细网格上的并行计算。然而现有的研究大多局限于对并行迭代算法本身或者差分格式加以改进来提高算法的稳定性,很少涉及超大规模问题热传导方程求解中数据的存储和计算问题。当计算问题规模相当大时,数据的强相关性和全局通信等问题将成为制约高性能计算的突出瓶颈问题。因此,大规模热传导偏微分方程的高效求解是目前大型科学与工程计算中迫切需要解决的具有挑战性的问题。

本文提出的热传导偏微分方程求解策略采用了分布式内存技术来解决超大规模稀疏矩阵的存储以及计算问题,整合了各种高效并行迭代求解线性、非线性问题的Krylov子空间方法和预条件子技术,基于面向对象思想实现了程序与算法选择、数据对象与存储格式的松耦合。

2 PETSc简介

2.1 PETSc定义

PETSc(Portable, Extensible Toolkit for Scientific Computation)是由美国Argonne国家实验室Satish Balay等人开发的可移植、可扩展科学计算工具箱,主要用于高性能求解偏微分方程组及其相关问题[4]。PETSc是一系列软件和库的集合,它为研发人员求解偏微方程主要提供了3个核心组件——TS(时间步进积分器)、SNES(非线性解法器)和SLES(线性解法器),而这些组件本身又是基于高性能的线性代数库(BLAS, LAPACK)和MPI消息传递环境等实现的。

2.2 PETSc抽象层次

PETSc实现的抽象层次如图1所示,其最底层基于BLAS, LAPACK高性能的线性代数库以及消息传递库MPI, MPI-IO。在此基础上,PETSc提供了性能监测接口来监测包括程序运行时间、浮点计算能力、内存占用以及MPI消息数

到稿日期:2008-12-22 返修日期:2009-03-02
程汤培(1983-),男,博士生,主要研究方向为分布式并行计算等,E-mail:tangpeicheng@gmail.com;王 群(1955-),男,教授,博士生导师,主要研究方向为地球科学及地球工程中高性能计算及可视化问题、地质空间信息共享协同环境及公共服务、地质空间数据挖掘等。

等性能指标，以帮助用户进一步分析和优化并行程序[5]。在PETSc中的算法实现采用了高度抽象的原则，向量、矩阵等基本数据对象完全采用抽象数据类型，对用户屏蔽了数据对象的区域分解和存储等细节。这种方式使得程序具有良好的可扩展性和可移植性，降低了数据结构、算法选择与并行应用的耦合度[6]。所有数据对象的划分、初始化和存取等基本操作都由DA(分布式数组)管理以及相应的PETSc库函数来实现。另外，PETSc不仅为求解中小规模线性方程组提供了高效的直接方法，还为大规模稀疏线性方程组的迭代求解提供了多种Krylov子空间方法和多种预条件子，并且具有错误诊断、计算性能分析、实时信息打印和图形可视化输出等功能。

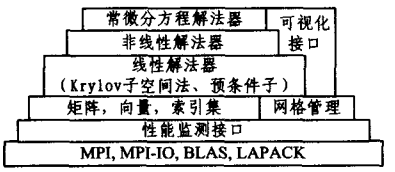

图1 PETSc实现的抽象层次

3 热传导方程及其差分格式

在有限差分法求解过程中，连续的时间和空间被划分成为一系列离散的点。在这些点上，连续的偏导数由有限差分公式来取代，将所求得的未知点联合起来，这些有限差分公式构成了一个线性方程组，然后对这个线性方程组进行联立求解，这样获得的解就是物理量在各个离散点上的近似解。

3.1 二维空间区域划分

考虑定义在二维规则区域 $\Omega=(0,M)\times(0,N)$ 上的热传导方程

$$\begin{cases}\dfrac{\partial^2 u}{\partial x^2}+\dfrac{\partial^2 u}{\partial y^2}=\dfrac{\partial u}{\partial t},(x,y)\in\Omega,t>0\\ u=u^0,t=0\\ u=g,(x,y)\in\partial\Omega,t\geqslant 0\end{cases}\tag{1}$$

其中，$u=u(x,y,t)$ 为未知函数；$u^0=u^0(x,y)$ 与 $g=g(x,y,t)$ 为已知函数，分别定义在区域 Ω 的内部和边界上。

在 x,y 方向上分别取步长 $h_x=\dfrac{M}{IM}$，$h_y=\dfrac{N}{JN}$，采用五点格式的二阶中心差商近似导数后，方程(1)离散为

$$\begin{cases}\dfrac{u_{i+1,j}-2u_{i,j}+u_{i-1,j}}{h_x^2}+\dfrac{u_{i,j+1}-2u_{i,j}+u_{i,j-1}}{h_y^2}=\dfrac{\partial u}{\partial t}\\ u_{i,j}|_{t=0}=u_{i,j}^0,1\leqslant i\leqslant IM-1,1\leqslant j\leqslant JN-1\\ u_{i,j}=g_{i,j},i=0\text{ or }i=IM\text{ or }j=0\text{ or }j=JN\end{cases}$$

即

$$\begin{cases}-\dfrac{2(h_x^2+h_y^2)}{h_x^2h_y^2}u_{i,j}+\dfrac{u_{i+1,j}+u_{i-1,j}}{h_x^2}+\dfrac{u_{i,j+1}+u_{i,j-1}}{h_y^2}=\dfrac{\partial u}{\partial t}\\ u_{i,j}|_{t=0}=u_{i,j}^0,1\leqslant i\leqslant IM-1,1\leqslant j\leqslant JN-1\\ u_{i,j}=g_{i,j},i=0\text{ or }i=IM\text{ or }j=0\text{ or }j=JN\end{cases}$$

令 $\alpha=h_x^2$，$\beta=h_y^2$，则

$$\begin{cases}-\dfrac{2(\alpha+\beta)}{\alpha\beta}u_{i,j}+\dfrac{u_{i+1,j}+u_{i-1,j}}{\alpha}+\dfrac{u_{i,j+1}+u_{i,j-1}}{\beta}=\dfrac{\partial u}{\partial t}\\ u_{i,j}|_{t=0}=u_{i,j}^0,1\leqslant i\leqslant IM-1,1\leqslant j\leqslant JN-1\\ u_{i,j}=g_{i,j},i=0\text{ or }i=IM\text{ or }j=0\text{ or }j=JN\end{cases}$$

令 $AU_i=-\dfrac{2(\alpha+\beta)}{\alpha\beta}u_{i,j}+\dfrac{1}{\beta}u_{i,j+1}+\dfrac{1}{\beta}u_{i,j-1}$，则

$$AU_i+\frac{1}{\alpha}u_{i+1,j}+\frac{1}{\alpha}u_{i-1,j}=\frac{\partial u}{\partial t}$$

$$\begin{bmatrix}A&\frac{1}{\alpha}I&&&&&\\ \frac{1}{\alpha}I&A&\frac{1}{\alpha}I&&&&\\ &\ddots&\ddots&\ddots&&&\\ &&\ddots&\ddots&\ddots&&\\ &&&\frac{1}{\alpha}I&A&\frac{1}{\alpha}I&\\ &&&&\frac{1}{\alpha}I&A\end{bmatrix}\begin{bmatrix}U_1\\ \\ \ddots\\ \\ U_{IM-1}\end{bmatrix}=\frac{\partial u}{\partial t}\tag{2}$$

其中

$$A=\begin{bmatrix}-\frac{2(\alpha+\beta)}{\alpha\beta}&\frac{1}{\beta}&&&&&\\ \frac{1}{\beta}&-\frac{2(\alpha+\beta)}{\alpha\beta}&\frac{1}{\beta}&&&&\\ &\ddots&\ddots&\ddots&&&\\ &&\ddots&\ddots&\ddots&&\\ &&&\frac{1}{\beta}&-\frac{2(\alpha+\beta)}{\alpha\beta}&\frac{1}{\beta}&\\ &&&&\frac{1}{\beta}&-\frac{2(\alpha+\beta)}{\alpha\beta}\end{bmatrix}$$

$$U_i=\begin{bmatrix}u_{i,1}\\ \ddots\\ u_{i,JN-1}\end{bmatrix}$$

3.2 三维空间区域划分

考虑定义在三维规则区域 $\Omega=(0,M)\times(0,N)\times(0,L)$ 上的热传导方程

$$\begin{cases}\dfrac{\partial^2 u}{\partial x^2}+\dfrac{\partial^2 u}{\partial y^2}+\dfrac{\partial^2 u}{\partial z^2}=\dfrac{\partial u}{\partial t},(x,y,z)\in\Omega,t>0\\ u=u^0,t=0\\ u=g,(x,y,z)\in\partial\Omega,t\geqslant 0\end{cases}\tag{3}$$

其中，$u=u(x,y,z,t)$ 为未知函数；$u^0=u^0(x,y,z)$ 与 $g=g(x,y,z,t)$ 为已知函数，分别定义在区域 Ω 的内部和边界上。

在 x,y,z 方向上分别取步长 $h_x=\dfrac{M}{IM}$，$h_y=\dfrac{N}{JN}$，$h_z=\dfrac{L}{KL}$，采用七点格式的二阶中心差商近似导数后，方程(3)离散为

$$\begin{cases}\dfrac{u_{i+1,j,k}-2u_{i,j,k}+u_{i-1,j,k}}{h_x^2}+\dfrac{u_{i,j+1,k}-2u_{i,j,k}+u_{i,j-1,k}}{h_y^2}+\\ \dfrac{u_{i,j,k+1}-2u_{i,j,k}+u_{i,j,k-1}}{h_z^2}=\dfrac{\partial u}{\partial t}\\ u_{i,j,k}|_{t=0}=u_{i,j,k}^0,1\leqslant i\leqslant IM-1,1\leqslant j\leqslant JN-1,\\ 1\leqslant k\leqslant KL-1\\ u_{i,j,k}=g_{i,j,k},i=0\text{ or }i=IM\text{ or }j=0\text{ or }j=JN\\ \text{or }k=0\text{ or }k=KL\end{cases}\tag{4}$$

为便于计算，将 x,y 和 z 方向上取成等步长 $h_x=h_y=h_z$，则方程(4)简化为

$$\begin{cases}-6u_{i,j,k}+u_{i+1,j,k}+u_{i-1,j,k}+u_{i,j+1,k}+u_{i,j-1,k}+u_{i,j,k+1}+\\ u_{i,j,k-1}=\dfrac{\partial u}{\partial t}\\ u_{i,j,k}|_{t=0}=u_{i,j,k}^0,1\leqslant i\leqslant IM-1,1\leqslant j\leqslant JN-1,\\ 1\leqslant k\leqslant KL-1\\ u_{i,j,k}=g_{i,j,k},i=0\text{ or }i=IM\text{ or }j=0\text{ or }j=JN\\ \text{or }k=0\text{ or }k=KL\end{cases}$$

令 $AU_i=-6u_{i,j,k}+u_{i,j+1,k}+u_{i,j-1,k}+u_{i,j,k+1}+u_{i,j,k-1}$，则

$AU_i+u_{i+1,j,k}+u_{i-1,j,k}=h_x^2\frac{\partial u}{\partial t}$

$$\Rightarrow\begin{bmatrix}A & I_1 & & & & \\ I_1 & A & I_1 & & & \\ & \ddots & \ddots & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & I_1 & A & I_1 \\ & & & & I_1 & A\end{bmatrix}\begin{bmatrix}U_1 & & \\ & \ddots & \\ & & U_{IM-1}\end{bmatrix}=h_x^2\frac{\partial u}{\partial t}\qquad(5)$$

其中，I_1 为 $(JN-1)(KL-1)$ 维的方阵，$U_i=\begin{bmatrix}u_{i,1,1}\\ u_{i,1,2}\\ \ddots\\ u_{i,1,KL-1}\\ \ddots\\ u_{i,JN-1,KL-1}\end{bmatrix}$。

令 $BU_{ij}=-6u_{i,j,k}+u_{i,j,k+1}+u_{i,j,k-1}$，则

$AU_i=BU_{ij}+u_{i,j+1,k}+u_{i,j-1,k}$

$$\Rightarrow A=\begin{bmatrix}B & I_2 & & & \\ I_2 & B & I_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & \ddots \\ & & & I_2 & B & I_2 \\ & & & & I_2 & B\end{bmatrix}\qquad(6)$$

其中，I_2 为 $KL-1$ 维的方阵，$U_{ij}=\begin{bmatrix}u_{i,j,1}\\ \\ \ddots\\ \\ u_{i,j,KL-1}\end{bmatrix}$，

$$B=\begin{bmatrix}-6 & 1 & & & & \\ 1 & -6 & 1 & & & \\ & \ddots & \ddots & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & 1 & -6 & 1 \\ & & & & 1 & -6\end{bmatrix}。$$

3.3 时间离散

3.3.1 显示格式

对于二维热传导方程，选取时间步长为 Δt，记 $u_{i,j}^n=u_{i,j}|_t=n\Delta t$，$g_{i,j}^n=g_{i,j}|_t=n\Delta t$，由一阶向前 Euler 格式离散方程得

$$\begin{cases}\dfrac{u_{i,j}^{n+1}-u_{i,j}^n}{\Delta t}=\dfrac{u_{i+1,j}^n-2u_{i,j}^n+u_{i-1,j}^n}{h_x^2}+\dfrac{u_{i,j+1}^n-2u_{i,j}^n+u_{i,j-1}^n}{h_y^2},\\ 1\leqslant i\leqslant IM-1,1\leqslant j\leqslant JN-1,n\geqslant 0\\ u_{i,j}^{n+1}=g_{i,j}^{n+1},i=0\text{ or }i=IM\text{ or }j=0\text{ or }j=JN,n\geqslant 0\end{cases}$$

该格式的稳定性条件为 $\frac{\Delta t}{h_x^2}+\frac{\Delta t}{h_y^2}<\frac{1}{2}$。

3.3.2 隐式、半隐式格式

由于采用显示格式对时间步长有着比较严格的限制，而隐式、半隐式格式是无条件稳定的，因此实际应用中更为广泛使用的是隐式格式。由一阶向后 Euler 格式离散方程得

$$\begin{cases}\dfrac{u_{i,j}^{n+1}-u_{i,j}^n}{\Delta t}=\dfrac{u_{i+1,j}^{n+1}-2u_{i,j}^{n+1}+u_{i-1,j}^{n+1}}{h_x^2}+\dfrac{u_{i,j+1}^{n+1}-2u_{i,j}^{n+1}+u_{i,j-1}^{n+1}}{h_y^2},\\ 1\leqslant i\leqslant IM-1,1\leqslant j\leqslant JN-1,n\geqslant 0\\ u_{i,j}^{n+1}=g_{i,j}^{n+1},i=0\text{ or }i=IM\text{ or }j=0\text{ or }j=JN,n\geqslant 0\end{cases}$$

采用隐式格式计算时，每个时间步需要解一个关于 $u_{i,j}^{n+1}$ 的线性代数方程组，可以采用某种迭代法并结合预条件技术求解。对于这种情况，最好选用一个现有的并行解法器包，例如 PETSc 的线性解法器；另一种较好的选择是使用 PETSc 的时间步进器 TS。

4 热传导方程并行求解

随着热传导问题规模的不断扩大，单机系统由于处理器、内存以及数据通道系统上的限制已遇到了无法避免的性能瓶颈。下面以三维热传导问题为例来说明热传导方程大规模并行求解方法。

4.1 时间的差分

如前所述，PETSc 的时间步进器可以处理关于时间的微分问题，即可解决形如 $Alhs(t)\frac{\partial u}{\partial t}=Arhs(t)$ 的问题。因而，对于式(3)，方程右侧可以通过指定时间离散格式以及系数矩阵求解函数等相关参数，利用时间步进器来实现自动求解时间上的微分。相对于向前差分，向后差分法因为在求解时要求对模型中所包含的所有有效计算单元的数据进行同步求解，所以计算量和内存需求量都会比较大。而在数值计算中，向后差分是无条件稳定的，因此对于时间的差分热传导方程求解过程中通常采用的是向后 Euler 差分法。

4.2 空间的差分

由式(5)、式(6)可知，方程左侧可以采用七点格式有限差分来实现对空间上的离散，最后形成一个带状对角稀疏矩阵。当网格划分达到百万、千万数量级的时候，单机系统已经无法开辟足够大的内存空间来存储经过离散后形成的稀疏矩阵。因此将数据进行区域划分并分别存储到各个处理机当中，建立全局索引来实现对数据的统一编址与管理。采用了一种分布式数组的方式来存储大容量数据，对用户屏蔽了数据对象的区域分解和分布式存储等细节。如图 2 所示，通过生成离散的网格点来代替方程连续的定义域，剖分网格和相关的数据，分配给所有参与计算的处理机，使得每个处理机都拥有网格的一个子集并且定义了需要与其进行通信的相邻网格单元。在每个处理机上，数值算法中的数据排列和数据访问具体形式是获得高效科学模拟的关键，应保证数据块的邻接处尽量少，以减少相邻网格单元的数据通信。

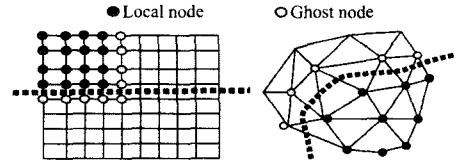

图 2 数据的存储与区域划分

由于偏微分方程离散后形成的系数矩阵包含了大量的零元素，因此有必要采用特殊的存储结构以避免存储零元素。稀疏矩阵的存储格式包含两方面的信息：非零元素及其在矩阵中的位置信息。采用了压缩矩阵的方式，比如 Aij 格式，仅存储大规模稀疏矩阵中的非零元素及其全局索引，以减少内

存开销。

当问题的计算规模较大时，稀疏矩阵的填充是影响程序性能的关键因素[5]。以三维均匀网格的有限差分后形成的系数矩阵的填充为例，对于每个处理机来说，分别定义本地网格行的全局上下界标号为 Istart 和 Iend。从 Istart 到 Iend，采用循环逐行插值的方式对矩阵赋值，I_i 为当前所在行的索引值。稀疏矩阵的填充流程可以描述为：

Step 1　将式(6)带入式(5)并展开矩阵 B 后，其主对角线上的元素均为-6，而 I_i 的值即主对角元素所在行的全局索引值，因此首先每行对角线位置即(I_i, I_i)上都需赋值为-6。

Step 2　然后考虑第 I_i 行所在的 A 矩阵在 $Arhs(t)$ 中的位置。矩阵 $Arhs(t)$ 按照行划分被分割成了 m 个区域，第一个区域的非零结构为 A, I_1，最后一个区域的非零结构为 I_1，A，中间区域的非零结构为 I_1, A, I_1。

Step 2.1　如果第 I_i 行所在的 A 矩阵不在 $Arhs(t)$ 的第一个区域即标号为 0 的区域，那么相对于第 I_i 行主对角元素之前的 nl 个元素的位置即$(I_i, I_i - nl)$上需赋值为 1；

Step 2.2　如果第 I_i 行所在的 A 矩阵不在 $Arhs(t)$ 的最后一个即标号为 $m-1$ 的区域，那么相对于第 I_i 行主对角元素之后的 nl 个元素的位置即$(I_i, I_i + nl)$上需赋值为 1。

Step 3　再考虑第 I_i 行所在的 B 矩阵相对于 A 矩阵中的位置。矩阵 A 按照行划分被分割成了 n 个区域，第一个区域的非零结构为 B, I_2，最后一个区域的非零结构为 I_2, B，中间区域的非零结构为 I_2, B, I_2。

Step 3.1　如果第 I_i 行所在的 B 矩阵不在 A 的第一个区域即标号为 0 的区域，那么相对于第 I_i 行主对角元素之前的 l 个元素的位置即$(I_i, I_i - l)$上需赋值为 1；

Step 3.2　如果第 I_i 行所在的 B 矩阵不在 A 的最后一个区域即标号为 $n-1$ 的区域，那么相对于第 I_i 行主对角元素之后的 l 个元素的位置即$(I_i, I_i + l)$上需赋值为 1。

Step 4　最后考虑第 I_i 行在矩阵 B 中的位置。

Step 4.1　如果第 I_i 行不在 B 的第一行即标号为 0 的行，那么相对于第 I_i 行主对角元素之前 1 个元素的位置即$(I_i, I_i - 1)$上需赋值为 1；

Step 4.2　如果第 I_i 行不在 B 的最后一行即标号为 $l-1$ 的行，那么相对于第 I_i 行主对角元素之后 1 个元素的位置即$(I_i, I_i + 1)$上需赋值为 1。

4.3　迭代求解

利用有限差分法分别对时间和空间进行离散后，热传导方程的求解可归结为求解一个大型的线性代数方程 $Au = f$。而求解稀疏线性方程组的常用方法包括两种：稀疏直接法和稀疏迭代法。由于直接法在求解过程不可避免需要引入填充元或者必须为减少填充元的数目而付出代价，尤其对于大规模的应用问题，实际所需的大量存储空间已成为获得高性能的瓶颈[7]，因此三维热传导方程的求解采用了开销较小的稀疏迭代法。程序集成了一些现有的高效并行迭代算法，比如广义最小残差法(GMRES)、Richardson 方法、共扼梯度法(CG 和 Bi-CG)等。

由于迭代过程中舍入误差的积累，对于某些条件不好的问题，迭代法可能不收敛，这点可以通过计算前给定好的预条件子以改善系数矩阵的谱性质来避免。所集成的预条件子包括雅可比矩阵方法(Jacobi)、块状雅可比矩阵方法(Block Jacobi)、超松弛迭代方法(SOR/ SSOR)、不完全 LU 分解(ILU)、可加性 Schwartz 方法(Overlapping Additive Schwarz)、多重网格预条件子(Multigrid)等。

三维热传导并行迭代求解完全基于面向对象技术实现，使得程序具有更好的灵活性和扩展性。我们构建了线性方程组求解类，并将 Krylov 子空间方法的名称以及预条件子的名称作为该类的构造函数的输入参数，实现了程序与算法选择的松耦合；构建了抽象矩阵类，并将矩阵维数、大小和结构信息作为该类的构造函数的输入参数，实现了对象数据与存储格式的松耦合。当需要改变 Krylov 子空间方法和预条件子以获得程序的最优计算性能时，可实现相关算法的运行时动态加载；当矩阵结构需要发生改变时，在运行时修改矩阵的相关输入参数，从而增强了程序的灵活性和可维护性。

三维热传导方程的求解流程可以描述为：

Step 1　生成离散的网格点来代替方程连续的定义域、剖分网格和相关的数据；

Step 2　给定问题的离散初始值和边界值；

Step 3　将整个计算网格区域划分为若干子区域，并将每个子区域的计算任务分配给相应处理机，使得每个处理机都拥有整个网格区域的一个子集并且定义了需要与其进行通信的相邻网格单元；

Step 4　计算当前时间步并进入时间步长的循环；

Step 5　每个处理机对所分配到的子区域的内部网格建立各自的数值模型并执行相应的迭代计算；

Step 6　每个处理机与拥有相邻网格单元的处理机进行通信，以计算处于边界上的网格数据；

Step 7　通过计算残差来检查数值解是否满足精度要求。如果满足，则转到 Step 4 并将当前运算结果作为下一次时间步循环的初始值；否则转到 Step 5，继续执行迭代。

5　性能评测

运行的硬件环境是由 4 个节点以及百兆以太网所构成的 Linux 集群，每个节点的配置为 Xeon 3.0(2M cache)双核 CPU，2GB 容量的 DDR2 主存。以三维热传导方程的并行求解为例进行计算，空间上的离散使用了正方体网格划分，时间上的离散使用了向后欧拉法，采用了在迭代步数达到 30 步重新开始计算的 GMRES[8] 作为并行迭代求解方法，选取 10^{-5} 作为残差精度来进行收敛性条件的判断，采用 BJACOBI 作为预条件子，0 级 ILU 作为子预条件子。

实验一：取 100 * 100 * 100 的网格划分，设置时间步长以保证显示时间格式下的计算稳定性，时间最大迭代次数为 100，分别使用不同处理机数来求解。表 1 分别给出了显示向前差分、隐式向后差分时间格式下不同处理机数的计算时间以及加速比。

表 1　不同处理机数下的计算时间以及加速比

Number of Processors	Forward Difference		Backward Difference	
	Escaping Time (s)	Speedup	Escaping Time (s)	Speedup
1	318.29	1	1287.62	1
2	180.64	1.76	729.90	1.76
4	97.88	3.25	387.27	3.31
8	83.96	3.79	288.18	4.45

表 1 中的结果显示，对于 4 个节点所组成的并行系统，在

处理机数为2～4时的并行程序保持较为理想的加速比，而当处理机数为8时加速比值较差。分析其原因，在于该并行程序的性能主要受到内存系统性能的限制，而非CPU的数量或CPU性能。对比表1中显式方法与隐式方法的加速比，随着处理机数的增加，隐式方法要好于显式方法。其原因一方面在于当每个处理机所分配的问题区域减小的同时，每个区域的对角部分也相应减小，因此求解块状雅克比预条件子所引入的计算量将减少。另一方面由于ILU不完全分解方法仅对位于对角线上的带状区域进行计算，随着对角线上的带状区域变得越来越窄，其计算量也将减少。对比表1中显式方法和隐式方法下的执行时间，前者要远低于后者。这是因为显式算法在进行迭代时仅用到邻近网格点的离散信息更新解向量，无需使用全局的线性或非线性解，而隐式方法每一步迭代需在全局范围内传播信息。因此，当时间步长可满足显式格式收敛性条件时，显式方法的程序执行速率将明显优于隐式方法。

实验二：取网格划分为100 * 100 * 8～100 * 100 * 128，表2中分别给出了由4个节点组成的并行系统在不同问题规模下的相对加速比和计算效率。

表2　不同问题规模下的相对加速比和计算效率

Grids	Escaping Time (s)	Speedup	Efficiency
100 * 100 * 8	45.17	2.66	66.58%
100 * 100 * 16	79.09	2.92	73.03%
100 * 100 * 32	146.72	2.90	72.37%
100 * 100 * 64	286.50	2.99	74.84%
100 * 100 * 128	591.95	2.95	73.73%

从表2可看出，当问题规模为100 * 100 * 8的网格时，并行程序的计算效率仅为66.58%，而问题规模扩大后效率基本稳定在73%～74%。分析其原因，一方面在于随着问题规模的扩大，计算初始化以及输入、输出所占的时间比重将减小，并行程序将获得更高的计算效率。另一方面，当问题规模达到一定程度，单机在计算取数据过程中存在大量的内外存交换开销，而并行系统大多数数据处理操作都是并行地在本地内存中执行，通信开销和计算开销均比较小。

结束语　热传导方程在地下水流动数值模拟、油藏数值模拟等工程计算中有着广泛应用，其并行实现是加快问题求解速度、提高问题求解规模的重要手段。然而，由于高精度模拟中所涉及的数据存储和计算等所带来的复杂性常常使得这类问题难于高效求解。本文提出的热传导偏微分方程求解策略采用了分布式内存技术来解决超大规模稀疏矩阵的存储以及计算问题，整合了各种高效并行迭代解法。实验表明，基于该策略所实现的面向大规模热传导问题并行求解程序具有良好的并行性能以及可扩展性。

参考文献

[1] Dawson C N, Du Qiang, Dupont T F. A Finite Difference Domain Decomposition Algorithm for Numerical solution of the Heat Equation[J]. Mathematics of Computation, 1991, 195(57):63-71

[2] Zhang Bao-lin, Wan Zheng-su. New techniques in designing finite-difference domain decomposition algorithm for the heat equation[J]. Computers & Mathematics with Applications, 2003, 45(10/11):1695-1705

[3] Lions J-L, Maday Y, Turinici G. A "parareal" in time discretization of PDE's[J]. Comptes Rendus de l'Académie des Sciences - Series I-Mathematics, 2001, 332(7):661-668

[4] Balay S, Buschelman K, Eijkhout V, et al. PETSc Users Manual [EB/OL]. http://www-unix.mcs.anl.gov/petsc/petsc-as/documentation/index.html#Manual

[5] Knepley M. PETSc Tutorial [EB/OL]. http://www-unix.mcs.anl.gov/petsc/petsc-as/documentation/tutorials/index.html

[6] Hovland P D, LMcInnes C. Parallel simulation of compressible flow using automatic differentiation and PETSc[J]. Parallel Computing, 2001, 27(4):503-519

[7] Kelley C T. Iterative Methods for Linear and Nonlinear Equations [M]. Philadelphia: SIAM Press, 1995

[8] Saad Y, Schultz M H. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems[J]. SIAM J. Sci. Stat. Comput., 1986, 7:856-869

(上接第108页)

思维整体处理 $\Delta c_2^{NF}[i]$。

(3)对于 $\Delta\Psi_7[i]$，关注 $\Delta\Psi_7[i]$是否等于 $\Delta\Phi_7[i]$，关注进位扩散对其的影响。

(4)在控制差分路径时，关注第32 bit。鉴于第32bit的特殊性，差分[+32]和[-32]在差分分析中认为是相同的。

结束语　本文是基于王小云教授所写的MD5破译的文章，从理论分析和程序验证两方面共同对MD5算法进行研解析。文中以第8步的差分特征为目标，详细解析了F函数的特性、模差分和带符号的异或差分的特点，深入分析了MD5第8步的各bit差分的产生过程，说明了满足其差分特征的条件，对王的文章的进行了部分修正，最后总结了控制差分路径的几个关键点。本文对MD5和其他Hash函数的分析和破译有着重要的作用。

参考文献

[1] Rivest R L. The MD4 message digest algorithm [C]// Advances in Cryptology-Crypto'90. LNCS 537. Springer-Veralg, 1991: 303-311

[2] Rivest R L. The MD5 message-digest algorithm[C]// Request for Comments (RFC 1320). 1992

[3] Zheng Y, Pieprzyk J, Seberry. HAVAL-A one-way hashing algorithm with variable length of output[C]// Advances in Cryptology, Auscrypto'92, LNCS 718. 1992: 83-104

[4] RIPE. Integrity primitives for secure information systems[R]. Final report of RACE integrity primitives evaluation (RIPE-RACE 1040), LNCS 1007, 1995

[5] FIPS 180-1. Secure hash standard[s]. NIST, US Department of Commerce, Springer-Verlag: Washington D C, 1996

[6] FIPS180-2. Secure hash standard[s]. http://csrc.nist.gov/publications

[7] Wang Xiaoyun, Lai Xuejia, Feng Dengguo. Cryptanalysis of the Hash Functions MD4 and RIPEMD[C]// EUROCRYPT 2005, LNCS 3494. 2005: 1-18

[8] Wang Xiaoyun, Yu Hongbo. How to break MD5 and other Hash Functions[C]// EUROCRYPT 2005, LNCS 3494. 2005: 19-35

[9] Wang Xiaoyun, Yu Hongbo. Efficient collision search attacks on SHA-0[C]// Crypto 2005, LNCS 3621. 2005: 1-16

[10] Wang Xiaoyun, Yu Hongbo. Finding collisions in the Full SHA-1 [C]// Crypto 2005, LNCS 3621. 2005: 17-36

[11] 王小云，冯登国，于秀源. 中国科学[J]. 信息科学，2005，35(3)：1-12

[12] Liang Jie, Lai Xuejia. Improved collision attack on Hash Function MD5[J]. Journal of Computer Science & Technology, 2007, 22(1):79-87