

MPI 并行调试与优化策略在三维绕流气体运动 论数值模拟中的应用

徐金秀¹ 李志辉^{2,3} 尹万旺¹

(江南计算技术研究所 无锡 214083)¹ (国家计算流体力学实验室 北京 100191)²
(中国空气动力研究与发展中心超高速研究所 绵阳 621000)³

摘 要 从求解三维绕流问题的 Boltzmann 模型方程的数值模拟程序出发,通过研究区域分解并行计算策略,引入输入/输出、通信与 CACHE 等优化策略,对数值模拟程序进行 MPI 并行化移植与高性能计算调试。以高空稀薄过渡流区飞行器绕流状态为算例,进行了 MPI 大规模并行计算测试,证实了所发展的 MPI 并行化区域分解策略及程序优化途径的正确性。研究表明开展的并行化实现能明显地缩短模式计算时间,并取得较好的效果。

关键词 并行调试,区域分解,Boltzmann 模型方程,并行计算,性能优化

中图法分类号 TP301 **文献标识码** A

Applications of MPI Parallel Debugging and Optimization Strategy in the Gas-Kinetic Numerical Algorithm for 3-D Flows

XU Jin-xiu¹ LI Zhi-hui^{2,3} YIN Wan-wang¹

(Jiangnan Institute of Computing Technology, Wuxi 214083, China)¹

(National Laboratory for Computational Fluid Dynamics, Beijing 100191, China)²

(China Aerodynamics Research and Development Center, Mianyang 621000, China)³

Abstract Based on the numerical simulated program of Boltzmann model equation in three dimensional flows, this paper studied the parallel strategy of domain decomposition and supplied the optimization methods in I/O, communication, memory access, etc, which are applied in the debugging and optimization of the parallel MPI program. Experimental results of large-scale parallel computing of flows around 3-dimensional sphere and spacecraft in rarefied field at high space show that the parallel strategy and the optimization methods are correct and efficient, and the parallel implementation scheme is very useful and can short the computing time explicitly.

Keywords Parallel debugging, Domain decomposition, Boltzmann model equation, Parallel computing, Performance optimization

1 引言

随着计算机技术的发展,高性能计算已经成为国家安全、经济和社会发展的一种潮流与发展趋势,具有越来越重要的意义。成千上万的并行处理器被组装起来开展高性能大规模并行计算,而且工程实际中需要解决的问题规模越来越大,这对研究开发能长时间稳定运行的并行计算环境提出了越来越严峻的挑战。当计算节点数量过于庞大时,就不可避免地会遭遇到网络延迟加剧和并行处理环境等方面的瓶颈,特别是超级计算机处理器节点数量超过一定范围,达到数千、数万个时,作业计算过程中某处理器出现问题而中断运行的几率会大大增加,系统的可靠性会大打折扣,且维护的难度明显增加;并行调试也成了发展大规模并行计算环境需要考虑的重要因素,用好高性能计算机,需要用户与并行计算机应用、维护人员共同努力,以提高高性能计算机应用水平。并行计算

机的计算模式主要分为共享存储和分布存储两大类,针对不同机器的特点需要设计相应的并行计算程序调试环境^[1,2];同时并行程序是在多个节点上运行,即使在单节点上也是以多进程或多线程的方式运行,这种计算模式突破了传统的串行计算概念,串行程序调试中的一些技术和手段将不再适用于大规模并行调试。以上原因都要求并行调试需要引入新的方法,对计算程序进行适应性改造,以更好地解决成千上万个处理器高性能并行计算与稳定运行问题。本文以求解 Boltzmann 模型方程的三维绕流气体运动论数值模拟及所发展的并行算法为例,通过研究设计区域分解大规模并行化策略,引入输入/输出、通信与 CACHE 优化技术,进行 MPI 高性能并行调试方案设计 & 性能分析与计算讨论。

2 区域分解并行化策略及 MPI 并行实现

三维绕流气体运动论数值模拟基于从稀薄流到连续流的

到稿日期:2011-06-26 返修日期:2011-10-19 本文受国家自然科学基金(91016027)资助。

徐金秀(1963—),女,硕士,高级工程师,主要研究方向为并行算法及应用,E-mail:jinxu3186@sina.com;李志辉(1968—),男,博士后,研究员,主要研究方向为 Boltzmann 模型方程和 DSMC 方法应用等;尹万旺(1980—),男,硕士,工程师,主要研究方向为并行算法及应用。

气体运动论统一算法,是由中国空气动力研究与发展中心李志辉研究团队从求解描述各流域气体流动输运现象统一的 Boltzmann 模型方程出发,发展起来的一种新算法,其数学模型和计算方法在文献[3-6]中进行了详细论述。本文以此为例,着重讨论该算法 MPI 大规模并行调试策略与方案实现及程序性能优化。

对于求解 Boltzmann 模型方程的三维绕流问题的数值计算,需要用到六维数组来存取各个离散速度坐标点(iv, jv, kv)、各个计算网格处(i, j, k)的气体分子速度分布函数,这将占用大量计算机内存和计算时间,其中 $iv=1, \dots, nvit, jv=1, \dots, nvjt, kv=1, \dots, nvkt, i=1, \dots, im, j=1, \dots, jm, k=1, \dots, km$ 。为了有效模拟复杂飞行器高超声速绕流,离散速度坐标点与位置空间计算网格数一般都会达到数十万、数百万量级以上,为此需要大量计算节点分担其计算内存及参与并行计算,这对由成千上万个计算节点组成的计算平台能否实施长时间大规模并行稳定计算提出了挑战。

由于求解 Boltzmann 模型方程的气体运动论统一算法将速度空间的计算与物理空间的计算解耦,这使得各个离散速度坐标点之间的计算具有非常好的并行独立性,特别适合于大规模的并行计算。为了从根本上解决气体运动论数值算法所需要的计算机内存与计算速度问题,同时充分利用现有大规模计算机并行技术资源,本文依托我国自行研制的分布式共享框架结构的“神威”并行超级计算机系统,在 FORTRAN+MPI 并行编译系统提供的并行程序设计环境下,对已经发展起来的气体运动论统一算法 HPF 并行化程序^[4,6,7]进行 FORTRAN90+MPI 并行移植及并行化模式研究。

模式中预处理和后处理过程的执行时间都较短,因此对模式的并行化集中在主程序部分。模式计算的速度空间在水平方向和垂直方向都没有数据相关性,因此我们的并行化策略选定在离散速度空间 3 个方向进行区域分解,即对数据进行分解,图 1 所示为离散速度空间 3 个方向区域分解示意图。所谓区域分解就是将模式的整个计算区域分为若干个相对独立的区域^[8],划分时遵循“owner-computing”原则,每个区域内部的计算均独立于其它区域,但需要在边界上与相邻区域进行必要的数据交换。

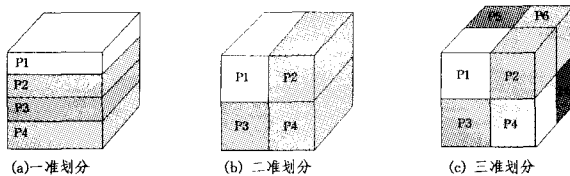


图 1 离散速度空间数据并行模式区域分解策略

2.1 一维区域分解方案

一维区域分解是沿垂直于 X, Y 或 Z 轴的某一个方向将三维网格空间切分为若干个 $N \times NVJT \times NVKT$ (沿垂直于 X 方向) 或 $NVIT \times N \times NVKT$ (沿 Y 方向) 或 $NVIT \times NVJT \times N$ (沿 Z 方向) 的三维模式网格切片。考虑到模式在划分方向的数据相关性所导致的数据通信问题及负载均衡,在划分时每个进程上的计算量只应相差一个切片。

2.2 二维区域分解方案

二维区域分解是沿垂直于 X 轴和 Y 轴方向分别对 $NVIT \times NVJT \times NVKT$ 的三维模式网格空间进行分割,将模式网格空间分割为若干个 $M \times N \times NVKT$ 柱状网格体。

在 Z 轴向下的俯视图上,每个区域为 $M \times N$ 的网格棋盘式划分结果。二维区域分解与一维区域分解类似,不同之处在于二维分割区域在两个方向上都存在相邻区域,在两个方向上都要进行数据交换。

2.3 三维区域分解方案

三维区域分解是沿垂直于 X 轴、 Y 轴和 Z 轴方向分别对 $NVIT \times NVJT \times NVKT$ 的三维模式网格空间进行分割,将模式网格空间分割为若干个 $M \times N \times K$ 的小立方体网格体,即每个区域为 $M \times N \times K$ 大小的小立方体划分结果。三维区域分解与一、二维区域分解类似,不同之处在于三维分割区域在 3 个方向上都存在相邻区域,在 3 个方向上都要进行数据交换。这里 M, N, K 表示沿 X, Y, Z 轴划分后每个方向的网格大小,如 $M = NVIT / XDIM$ ($XDIM$ 为 X 方向进程数)。

2.4 3 种区域分解方案的比较

一维区域分解的优点在于:实现起来较为简单,对模式串行版本的修改可以达到最少;进程间的通信次数和通信的总数据量可以达到最小。

一维区域分解的缺点在于:可扩展性较差,因为模式的计算区域为 $NVIT$ 个(假设 X 方向) $1 \times nvjt \times nvkt$ 的网格切片,相对于现代的超级计算机就显得数目较少,当程序所占用的进程数增大到一定数量时,并行效率就会变得较低。

二维、三维区域分解的优点在于:与一维区域分解方案相比,其可扩展性较好,可以利用更大数目的进程数,即充分利用现代超级计算提供的大规模计算能力。缺点是对串行版本修改较大,调试较困难,工作量大。

通过对 3 种区域分解方案利弊和可行性的深入分析与权衡比较可知,三维区域分解方案在该模式的并行化中较为可行,故我们的 MPI 并行化调试是基于沿离散速度空间 3 个方向的区域分解方案进行的。每个方向的数据并行分布计算如下:

```
nproc = XDIM * YDIM * ZDIM
myid_x = mod(myid, XDIM)
myid_y = mod((myid / XDIM), YDIM)
myid_z = myid / (XDIM * YDIM)
IST = 1
MM = MOD(NVIT, XDIM)
IF (myid_x .LT. MM) THEN
    IEND = NVIT / XDIM + 1
    IPT = myid_x * IEND
ELSE
    IEND = NVIT / XDIM
    IPT = myid_x * IEND + MM
ENDIF
JST = 1
MM = MOD(NVJT, YDIM)
IF (myid_y .LT. MM) THEN
    JEND = NVJT / YDIM + 1
    JPT = myid_y * JEND
ELSE
    JEND = NVJT / YDIM
    JPT = myid_y * JEND + MM
ENDIF
KST = 1
MM = MOD(NVKT, ZDIM)
IF (myid_z .LT. MM) THEN
    KEND = NVKT / ZDIM + 1
```

```

KPT=myid_z * KEND
ELSE
KEND=NVKT/ZDIM
KPT=myid_z * KEND+MM
ENDIF

```

这里, *NVIT*、*NVJT*、*NVKT* 分别为离散速度空间 3 个方向的网格点数; *im*、*jm*、*km* 分别为位置空间 *x*、*y*、*z* 3 个方向的网格点数; *XDIM*、*YDIM*、*ZDIM* 分别为对应 3 个方向的进程数, *nproc* 为程序运行时总进程数, *IEND*、*JEND*、*KEND* 分别为每个进程划分到的计算网格点数, 于是图 1 中每个小立方体网格数据大小为 $IEND \times JEND \times KEND$ 。

注: 这里需要强调的是节点数和进程数是一个概念, 若每个计算节点配置一个计算核, 则我们所说的进程数和节点数是一个概念; 若每个计算节点配置 *n* 个计算核(即多核), 则进程数=计算节点数 * *n*。

3 MPI 并行调试与优化

基于以上区域分解并行化策略和离散速度空间并行分布模式的分析, 对该模式采用 SPMD 类型的 MPI 并行编程模型, 该编程模型易于编写且具有良好的可扩展性。并行程序中主要根据数据所在的计算节点的逻辑号(进程号)进行判断, 以决定该进程应该执行怎样的计算操作。并行程序与串行程序相比, 主要的差异除了计算与输入/输出外, 还包括数据集到进程的指派、进程间的数据通信和进程间的归约、同步等操作。

3.1 输入/输出

选定逻辑编号为 0 的进程作为主进程, 由它来完成输入/输出任务。程序开始执行时由 0 进程读入初始数据文件, 进行必要的程序数据初始化, 然后由 0 进程将初始化后的数据广播到其它结点上。

在时间步迭代满足一些输出条件的时候, 由 0 进程收集其它进程上的数据, 有些变量经过必要的处理, 形成完整的输出数据集后再写到输出文件中。在并行程序执行过程中, 除了 0 进程和指定的输出进程(物理空间中心几何点所在的进程)外, 其它进程一般都不进行文件操作。这样一方面保持了输入/输出数据的完整性和一致性, 另一方面大大减少了程序执行中的输入/输出开销, 对提高并行程序执行性能大为有利。但由于迭代时间较长, 迭代过程中必须有写断点文件操作, 又由于运算的数据量较大, 每个进程的断点文件由每个进程自己写。

3.2 通信分析

由于是在离散速度空间进行数据分解, 该模式在迭代计算中没有数据相关, 但宏观变量的分析需要收集全局数据, 因此通信以全局通信为主。为了解决通信带来的额外开销, 需要采取下一节介绍的技术途径减少通信时间, 提高 MPI 全局通信效率, 通信与计算并行实施。

3.3 优化策略

3.3.1 “写”文件优化

Boltzmann 模型方程的求解是一个长时间的迭代求解过程, 一般需要数万步甚至 10 多万步。考虑到机器的稳定性, “写”断点数据是保证迭代继续下去的非常重要且必不可少的步骤。由于“写”文件在操作系统底层是一个开销很大的操作, 操作系统建议一般尽量少写或一次尽量多写一些数据。

如果写保留恢复数据采用三重循环调用“写”文件操作, 则会使底层“写”开销很大, 效率很低。我们采用一次“写”完成, 极大地提高了写断点文件的效率。具体如下, 对于:

```

DO KV=KST,KEND
DO JV=JST,JEND
DO IV=IST,IEND
write(198) (((FDV(i,j,k,iv,jv,kv),i=1,imt),j=1,jmt),k=
1,kmt)
ENDDO
ENDDO
ENDDO

```

修改为:

```
WRITE(198) FDV
```

这样一次“写”操作就完成了 FDV 数组的写入。

3.3.2 数组降维与循环次序调整的 CACHE 优化

三维绕流气体运动论数值模拟主要运算部分用到 2 个 6 维数组, 绝大部分计算都在 6 重循环内进行。CACHE 大小是有限的, 数据太大或数据访问地址不连续都可能导致 CACHE 命中率低, 而 CACHE 命中率对性能影响至关重要。FORTRAN 数组存储特点是从第一维开始连续存储, 为了提高 CACHE 命中率, 循环语句最内层须从第一维开始, 但很多程序段的最内层循环却是从第二维或第三维开始的, 数据的不连续性导致了 CACHE 访问的缺失而影响性能, 故须调整循环次序。具体举例如下:

```

REAL::X(is,ie,js,je,ks,ke)
do kv=1,nvk
do iv=1,nvi
do jv=1,nvj
FDV(iv,jv,kv)=(X(iv,jv+1,kv)-2 * X(iv,jv,kv)+X
(iv,jv-1,kv))/2
end do
end do
end do

```

上述最内层循环从第二维变量 *jv* 开始, 应调整为从第一维 *iv* 开始, 但由于原程序中每一个点的计算涉及到 *jv*, *jv*+1, *jv*-1 三行数据, 考虑到数据的相关性, 需将这三行数据一次传入, 具体示例如下:

```

do kv=1,nvk
do jv=1,nvj
CALL F1NEW(FDV(is,jv,kv),X(is,jv+1,kv),X(is,jv,kv),X
(is,jv-1,kv))
end do
end do
SUBROUTINE F1NEW(FDV,X1,X2,X3)
real::FDV(is,ie),X1(is,ie),X2(is,ie),X3(is,ie),
integer::i
do i=is,ie
FDV(i)=(X1(i)-2 * X2(i)+X3(i))/2
end do
END SUBROUTINE F1NEW

```

一维数组在 CACHE 中的读取是连续的, 对符合这种特点的程序段, 这种优化措施效率很明显。

3.3.3 用逻辑判断代替浮点乘

该模式中有一个主要调用函数如下:

```
FUNCTION FMINMD(X,Y)
IMPLICIT NONE
real::X,Y,FMINMD
IF (X * Y.GT. 0.0 ) THEN
.....
ELSE
FMINMD=0.0
ENDIF
RETURN
END FUNCTION FMINMD
```

因 X、Y 是很小的浮点数,这两个数相乘时常常超过机器的有效数值表达范围而进入操作系统的“非规格化数”模拟,从而导致运算速度成倍下降,改为逻辑判断后性能有了 2 倍多的提高。即:

```
IF (X * Y.GT. 0.0 ) THEN
修改为:
IF((X.GT. 0.0.and. Y.GT. 0.0).OR.(X.LT. 0.0.and. Y.LT. 0.0)) THEN
```

通过以上一些优化措施,程序运行效率的提高 5~6 倍了。

3.4 优化效率

测试案例: $kbody=3, cpus=2000, imt=31, jmt=21, kmt=31$, 表 1 是一次迭代期间不同程序段优化前后的执行时间。

表 1 各种优化方法的测试时间对比

优化项	优化前(秒)	优化后(秒)
写文件	150.1	10.0
计算优化	6.54	6.12
循环次序调整	6.58	3.85
MPI 全局归约	10.1	3.2
非规格化数	165.0~177.0	40~41

4 MPI 并行测试与计算结果

基于上述离散速度空间区域分解并行化方法与优化策略对求解 Boltzmann 模型方程的三维绕流气体运动论数值模拟程序进行 MPI 大规模并行调试,经过编译器进行词法、语法和语义分析及程序优化与适应性改造,在“神威蓝光”高性能并行计算机上进行大规模并行测试。拟定同一规模的统一算法 MPI 并行计算程序,分别在 8075、16150、24225、32300、40375、48450、80750 个 CPU 进程上进行了性能、加速比测试,实测加速比如表 2 所列。表中第一行给出了所用 CPU 进程数,第二行给出了不同进程数下完成统一算法一个完整迭代步所需要的平均计算时间(秒/步),第三行给出了相应进程数基于 8075 的并行计算加速比。图 2 绘出了基于 8075 个 CPU 进程数计算得到的并行加速比随 CPU 数目的变化关系。可看出,统一算法 MPI 大规模并行程序从数千到数万 CPU 进程数并行计算,算法加速比随处理机数目的增加基本上呈拟线性分布,说明算法具有很高的并行加速性能与并行效率,证实了本文发展的 MPI 并行化区域分解模式与输入/输出、通信与 CACHE 优化策略的高效准确性。

表 2 MPI 并行计算实测加速比

进程数	8075	16150	24225	32300	40375	48450	80750
时间	186.26	97.94	62.65	56.86	45.32	39.32	36.38
加速比	1.0	1.90	2.97	3.28	4.11	4.74	5.08

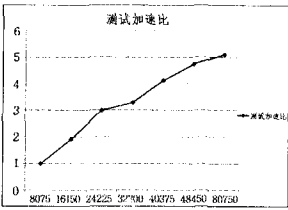
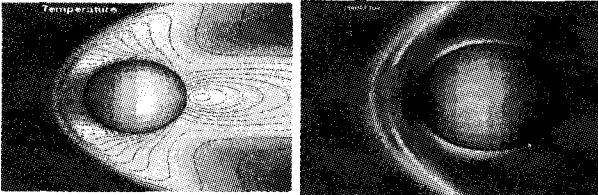
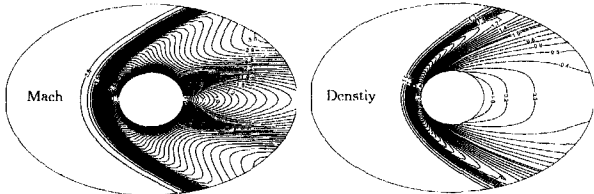


图 2 求解 Boltzmann 模型方程统一算法基于 8075~80750CPU 进程数并行计算加速比



(a) 温度 (b) 热流

图 3 轴对称面绕流($Kn_\infty=0.03, M_\infty=3.83$)等值线云图计算结果



(a) 马赫数 (b) 密度

图 4 轴对称面绕流($Kn_\infty=0.03, M_\infty=3.83$)流场等值线计算结果

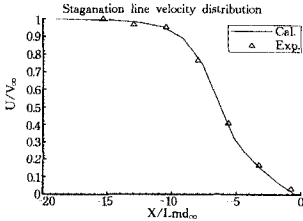


图 5 飞行器绕流($Kn_\infty=0.03, M_\infty=3.83$)驻点线速度 MPI 并行计算与实验结果比较

为了考验求解 Boltzmann 模型方程三维绕流气体运动论数值模拟 MPI 并行程序用于求解飞行器跨流域绕流问题计算结果的准确可靠性,拟定 80 公里高空稀薄过渡流区一绕流状态,取来流克努森数 $Kn_\infty=0.03$ 、来流马赫数 $M_\infty=3.83$,该状态所用计算空间规模是:离散速度空间坐标点 $40 \times 25 \times 25$ 、位置空间网格划分 $61 \times 25 \times 51$,进行 MPI 并行计算 10000 步,达到计算收敛的残差为 1×10^{-6} 。图 3(a)、(b)分别绘出该绕流状态轴对称面内的温度与热流等值线云图。可以看出超声速绕流飞行器在物体前部驻点区出现高温区,且飞行器表面温度因热交换而增加很大,因脱体激波在飞行器前面一定区域产生较高的热流剧烈变化区。图 4(a)、(b)绘出该绕流流场轴对称面内的马赫数与密度等值线分布,揭示了超声速绕流飞行器前部一定区域出现脱体激波,并在物体后部出现再压缩波及背风真空区绕流现象。图 3 与图 4 直观揭示了超声速飞行器绕流面貌与流动变化特点,所得并行计算结果与理论预测分析吻合较好。图 5 绘出了本文 MPI 并行计算得到的飞行器绕流驻点线速度 U/V_∞ 随脱体距离 X/λ_∞ 的变化关系与实验结果(符号 Δ 表示)的比较情况。可看出,

(下转封三)

SIRT 算法即包含在 ATOM 软件中。同时,ATOM 软件还包含串行 WBP $\times$ ART 和并行 ART $\times$ SIRT 重构算法。

结束语 本文主要分析了 SIRT 算法移植到 GPU 平台上的过程,并通过 CUDA 中的合并访问、常量存储器、共享存储器等优化技术对并行算法进行了优化。并行迭代重构算法在 Tesla C1060 GPU 平台上相对于运行在 Intel i7 920 CPU 平台上的串行算法性能上最高提高了 47.86 倍,极大地提高了重构的性能,且没有任何重构质量上的下降。在串行平台上重构 2048 $\times$ 2048 像素的图像时需要数天的时间,而在我们开发的并行算法平台上仅需要不到 3 个小时的时间。同时,本文简单介绍了国内首款免费的电子断层三维重构软件-ATOM 1.0,其使用简单、性能高、成本低、移植性强。

本文所完成的工作还有进一步优化的空间,还有待做更深入的研究和进一步的完善:SIRT 算法是按切片进行重构的,片与片之间的数据也没有任何依赖性,因此可以在 GPU 集群上采用 MPI 和 CUDA 相结合的技术,由 MPI 完成片与片之间的并行,由每个节点上的 GPU 完成片内的并行,通过 GPU 集群可以使重构速度更快,争取在数分钟内完成大尺寸图像的重构。ATOM 1.0 软件的功能还需要进一步的完善,如添加更多的并行重构算法(如并行加权背投影重构算法)等。

参考文献

- [1] 张凯,张艳,胡仲军,等. 电子显微镜三维重构技术发展与前沿[J]. 生物物理学报,2010,26(7):533-559
- [2] Derosier D J, Kak A. Reconstruction of Three Dimensional Structures from Electron Micrographs[J]. Nature, 1968, 217:

(上接第 303 页)

两者吻合很好。通过对图 4、图 5 超声速绕流状态($Kn_{\infty}=0.03, M_{\infty}=3.83$)并行计算结果、有关实验数据的理论分析与比较确认,证实了本文并行计算结果正确可靠。

结束语 本文通过研究区域分解并行计算模式,引入输入/输出、通信与 CACHE 优化策略,对求解 Boltzmann 模型方程的三维绕流气体运动论数值模拟程序进行 MPI 并行化移植与高性能计算调试,以高空稀薄过渡流区飞行器绕流状态为算例,进行了 MPI 大规模并行计算测试与并行计算研究,证实了本文所发展的 MPI 并行化区域分解策略及程序优化途径的正确性。研究表明本文开展的并行化实现能明显地缩短模式计算时间,取得了较好的效果。同时看出,每个进程上的负载取决于区域分解时分配的网格大小。当结点数较少时,由于每个进程所分配的分割区域较大,这时负载不平衡及通信带来的影响相对而言不明显;而当结点数较多时,每个进程所分配的分割区域越来越小,这时负载不平衡及通信带来的影响是较明显的。由于选取的模式网格空间的大小一定,随着结点数的增加,每个进程所负担的计算量不断减少,通信所占用的时间就显得较为重要。

本文所发展的 MPI 并行调试方法与程序优化工具在求解 Boltzmann 模型方程的三维绕流气体运动论数值模拟程序

130-134

- [3] Gordon R, Bender R, Herman G T. Algebraic Reconstruction Techniques (ART) for three Dimensional Electron Microscopy and X-ray Photography[J]. Journal of Theoretical Biology, 1970,29(3):471-481
- [4] Gilbert P. Iterative Methods for the Three-Dimensional Reconstruction of an Object from Projection[J]. Journal of Theoretical Biology,1972,36(1):105-107
- [5] Herman G T. Fundamentals of Computerized Tomography: Image Reconstruction from Projection(2nd edition)[M]. Springer, 2009
- [6] Andersen A H, Kak A C. Simultaneous Algebraic Reconstruction Technique (SART): A Superior Implementation of the ART Algorithm[J]. Ultrason Imaging,1984,6(1):81-94
- [7] 向良忠,邢达,谷怀民,等. 改进的同步迭代算法在光声血管成像中的应用[J]. 物理学报,2007,56(7):3911-3916
- [8] Yan X, Sinkovits R S, Baker TS. AUTO3DEM—an Automated and High Throughput Program for Image Reconstruction of Icosahedral Particles[J]. Journal of Structural Biology, 2007, 157(1):73-82
- [9] Zheng S Q, Keszthelyi B, Branlund E, et al. UCSF Tomography: an Integrated Software Suite for Real-Time Electron Microscopic Tomographic Data Collection, Alignment, and Reconstruction[J]. Journal of Structural Biology,2007,157(1):138-147
- [10] Lindholm E, Nickolls J, Oberman S, et al. NVIDIA Tesla: A Unified Graphics and Computing Architecture[J]. IEEE Micro, 2008,28(2):39-55
- [11] 储琪,张凯,万晓华,等. ATOM 1.0:基于 GPU 的电子断层重构软件[J]. 生物物理学报,2011,27(3):231-241

的大规模并行计算的应用中,已取得了一定的成果,以后将进一步努力,不断优化并行算法程序和 CACHE、内存使用,提高计算速度,以满足不断发展的模式研究和实际计算需求。

参考文献

- [1] Wang F, et al. Completely Debugging Indeterminate MPI/PVM Programs [J]. Journal of Software,2001,12(3)
- [2] 王巍,方滨兴,张宏莉. 并行调试中的若干关键问题 [J]. 计算机科学,2003,30(3):132
- [3] 李志辉,张涵信. 稀薄流到连续流的气体运动论统一数值算法初步研究[J]. 空气动力学学,2000,18(3):255-263
- [4] 李志辉. 从稀薄流到连续流的气体运动论统一数值算法研究[D]. 绵阳:中国空气动力研究与发展中心,2001
- [6] 李志辉,张涵信. 基于模型方程的气体运动论统一算法研究[J]. 力学进展,2005,35(4):560-572
- [5] 李志辉,张涵信. 稀薄流到连续流的气体运动论统一算法研究[J]. 空气动力学学,2003,21(3):256-264
- [7] 李志辉,张涵信,符松. 基于 Boltzmann 模型方程的气体运动论 HPF 并行算法[J]. 计算物理,2003,20(1):1-8
- [8] 陆林生,董超群,李志辉. 多相空间数值模拟并行化研究[J]. 计算机科学,2003,30(3):129-131,137