基于新一代神威超算的计算流体力学 Palabos 软件的并行优化

柳安军，殷洪辉，王利，刘智翔，孔博，郭猛，陈成敏，杨美红

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.220100089

基于新一代神威超算的计算流体力学 Palabos 软件的并行优化

柳安军[1]　殷洪辉[2]　王　利[1]　刘智翔[3]　孔　博[4]　郭　猛[1,2]　陈成敏[2]　杨美红[1]
1 齐鲁工业大学(山东省科学院)山东省计算中心(国家超级计算济南中心)　济南 250014
2 济南超级计算技术研究院济南市高性能工业软件重点实验室　济南 251013
3 上海海洋大学信息学院　上海 201306
4 广东以色列理工学院　广东 汕头 526100
(liuaj@sdas.org)

摘　要　Palabos 软件是一款基于格子玻尔兹曼算法(Lattice Boltzmann Method,LBM)的计算流体力学软件,因其优异的计算能力被广泛用于多孔介质、自由界面、颗粒运动、血液流动等计算流体力学领域。Palabos 软件广泛的用户需求使其迫切需要在神威超算上进行移植优化和并行加速,服务于能源、化工行业。文中在新一代神威超算(SW26010pro)上对 Palabos 软件进行异构并行设计,针对 Palabos 的数据结构和模块化编程不利于神威众核编程的问题,通过直接取址,设置字段标记处理多态导致的分支、数据切片处理等优化思路;并结合新一代神威超算的特性,使用共享内存和寄存器通信的优化技术,实现众核加速 2～6 倍。同时实现 Palabos 软件在新一代神威超算上的复杂化工过程多尺度计算方向上两相流算法的百万核心规模的并行计算,以 6.4 万核心的并行计算规模为基准,百万核心的并行效率大于 40%。

关键词:众核化;模块化编程;Palabos;新一代神威超算;多相流
中图法分类号　TP391

Parallel Optimization of Computational Fluid Dynamics Application Palabos Based on Next Generation Sunway Supercomputer

LIU An-jun[1],YIN Hong-hui[2],WANG Li[1],LIU Zhi-xiang[3],KONG Bo[4],GUO Meng[1,2],CHEN Cheng-min[2] and YANG Mei-hong[1]
1 Shandong Computer Science Center(National Supercomputing Center in Jinan),Qilu University of Technology(Shandong Academy of Sciences),Jinan 250014,China
2 Jinan Key Laboratory of High Performance Industrial Software,Jinan Institute of Supercomputer Technology,Jinan 251013,China
3 College of Information Technology,Shanghai Ocean University,Shanghai 201306,China
4 Guangdong Technion Israel Institute of Technology,Shantou,Guangdong 526100,China

Abstract　Parallel lattice Boltzmann(Palabos)software is a widely used computational fluid dynamics software based on lattice Boltzmann method(LBM),which is widely used in the field of porous media,free interface,particle motion,blood flow and so on due to its excellent computing power. Palabos has a wide range of user needs,which makes it urgent to transplant,optimize and accelerate parallel on Sunway supercomputer to serve the energy and chemical industry. In this paper,the heterogeneous parallel design of Palabos software is carried out on the new generation Sunway supercomputer system(SW26010pro). The data structure and template programming of Palabos are not suitable for the heterogeneous parallel of Sunway supercomputer system. So we design the parallel optimization techniques called direct getting address,polymorphic tag processing and data slicing to deal with the Palabos data structure and template programming. Combined with the characteristics of the new generation of Sunway supercomputer system,the optimization technology of shared memory and register memory access(RMA) is also adopted. The acceleration efficiency of 64 computing processing elements(CPEs) is 2～6 speed up. The Palabos software is realized the parallel computing of one million core scale of two-phase flow algorithm in the field of complex multi-scale chemical process in the new generation

到稿日期:2022-01-10　返修日期:2022-05-09
基金项目:国家重点研发计划(2018YFB0704002);鳌山科技创新计划重大项目(2018ASKJ01);山东省重大科技创新工程项目(2019JZZY010302);山东省重点研发计划(国际科技合作)(2019GHZ018);山东省博士后人才创新支持计划(SDBX2020018);光合基金 B(202107021062)
This work was supported by the National Key Research and Development Program(2018YFB0704002),Aoshan Science and Technology Innovation Project(2018ASKJ01), Major Scientific and Technological Innovation Projects in Shandong Province(2019JZZY010302),Shandong Key Research and Development Program(International Cooperation Office)(2019GHZ018),Shandong Province Postdoctoral Innovative Talents Support Plan(SDBX2020018) and GH fund B(202107021062).
通信作者:杨美红(yangmh@sdas.org)

Sunway supercomputer system. The one million cores parallel efficiency is more than 40% compared with 64000 cores.
Keywords Many core, Modulation programming, Palabos, SW26010pro, Multiphase flow

计算流体力学的主流开源软件有 OpenFoam，Palabos，SU，FEATFLOW 等。国内外各课题组都基于具体研究场景选择或开发不同的计算流体力学程序。格子玻尔兹曼方法是一种基于介尺度的计算流体力学的方法，与传统计算流体力学方法相比，该方法同时具有介于微观分子动力学模型和宏观连续模型的介观模型特点，利用简化的运动学方程，将流场转化为格点，易于划分网格和计算域，特别适合描述复杂两相、颗粒流动问题。此外，LBM 只有相邻格点间的通信，非常适合模拟复杂几何形状和复杂流动的大规模并行计算。在神威上对 LBM 算法的优化是服务于工业计算的重要方向。针对神威超算平台的 LBM 算法的优化，目前已有研究工作涉及[1-4]。Lv 等[1]基于“神威·太湖之光”超级计算机系统，开发了一套高效扩展的 LBM 计算流体力学软件，设计了面向 19 点 stencil 的数据复用、碰撞过程向量化、主从异步并行计算隐藏通信开销等优化策略，测试了高达 56000 亿网格的数值模拟，持续浮点计算性能达 4.7 PFlops，软件模拟速度提高了 172 倍。相比百万核心 1000×1000×5000 网格风场模拟，软件千万核心的并行效率可达 87%。Liu 等[3]针对具有良好数值稳定性的多弛豫时间模型格子 Boltzmann 方法（Multi-Relaxation Time Lattice Boltzmann Method，MRT-LBM），结合大涡模拟湍流模型和曲面边界插值格式在神威蓝光超级计算机上的测试，提出了适合于大规模分布式集群的网格生成、流场信息初始化和迭代计算的并行算法，运行结果证明该并行算法在十万计算核的量级下仍具有良好的加速比和可扩展性。以上工作均是自研软件在神威上的移植优化。我们需要在神威超算上优化一款通用的 LBM 开源软件，以方便更多从业人员使用。当前，流行的 LBM 代码有 TheLMA[5]，OpenFSL[6]，LBsoft[7]，Hemocell[8]，Musubi[9] 和 Palabos[10-11]。Palabos 继承了 OpenLB 的思想，设计了一种格子玻尔兹曼方法，用于计算流体力学（Computational Fluid Dynamics，CFD）框架，在格子波尔兹曼开源软件上具有重要地位。另外，Palabos 是并行软件（Parallel Lattice Boltzmann），支持大规模 MPI 并行计算。Palabos 于 2011 年发布第一版以后，目前最新版本为 2022 年提出的 2.2 版本，2011—2019 年，流体行业使用 Palabos 软件完成计算论文 300 多篇，博士论文 35 篇（见图 1），在粒子运动、自由界面[12]、多孔介质[13]、血液流动[14-17]方面均有应用，其中血液流动案例开展了 GPU 加速设计。

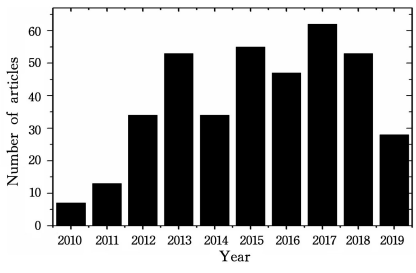


图 1 Palabos 软件论文数量

Fig. 1 Number of Palabos software papers

新一代神威超算上的优化工作已经取得优异的成绩。Liu 等[18]的神威量子模拟器、Xiao 等[19]的核聚变托克马克装置千万核心并行计算以及 Shang 等[20]的第一性原理的千万核心并行计算获得了戈登贝儿奖的提名。其中，Liu 等[18]对悬铃木的回击，获得了 2021 年度超算最高荣誉戈登贝儿奖。本项目针对新一代神威超算体系架构，对计算流体力学软件 Palabos 进行移植和众核并行计算设计，结合 LBM 算法的体系架构，提出了高级语言的众核化思路和解决方案，为神威的软件生态提供 LBM 算法的通用开源软件。同时，使用优化后的求解器模拟复杂化工过程中液液和液固的相互作用，实现百万核心的大规模并行计算，为化工行业数值模拟并行计算提供示范。

1 新一代神威体系架构和 Palabos 软件结构

1.1 新一代神威体系架构

神威新一代超级计算机系统继承和发展了“神威·太湖之光”体系架构，是基于申威新一代高性能异构众核处理器和互连网络芯片构建的。如图 2 所示，申威新一代众核处理器 SW26010pro 集成 6 个核组（Core Group，CG），每个核组包含一个运算控制核心（Management Processing Element，MPE）和一个 8 * 8 运算核心（Computing Processing Elements，CPE）阵列，这些部件通过环形网络进行连接。MPE 的主要功能是计算、控制、通信、IO 等，CPE 主要用于计算。CG 中 CPE 以 8 * 8 阵列方式进行排布，运算核心之间以及运算核心与外部交互通过阵列内的网络进行互连。运算核心阵列中任意两个运算核心之间可以通过 RMA 方式进行数据通信和集合通信。

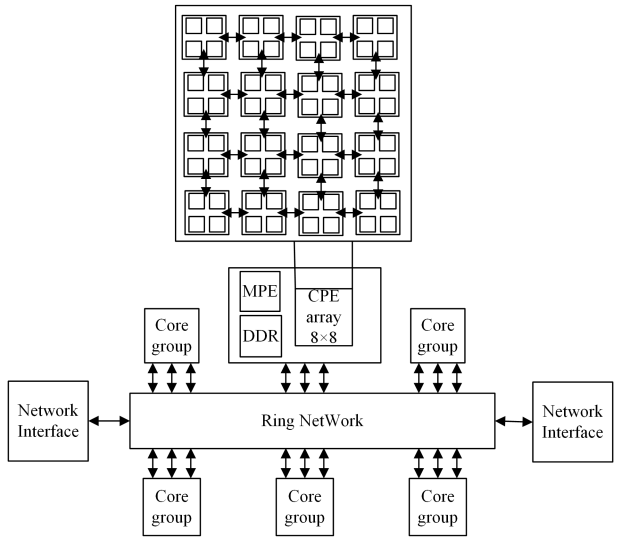


图 2 新一代申威众核处理器 SW26010pro 的结构

Fig. 2 Architecture of new generation Sunway many-core processor SW26010pro

MPE 采用自主 SW64 指令集，具有 32KB L1 指令缓存、32KB L1 数据高速缓存和 512KB L2 高速缓存。CPE 采用自主 SW64 指令集，具有 512 位 SIMD（Single Instruction Multiple Data）向量，支持双精度、单精度、半精度浮点计算和整数运算。每个 CPE 都拥有独立的指令缓存和数据存储空间。

数据存储空间可以配置成完全由用户控制的局部数据存储器(Local Data Memory,LDM),也可以将部分数据存储空间配置成硬件自动管理的局部数据高速缓冲(LDcache)。主从核之间可通过直接内存访问(Direct Memory Access,DMA)的方式实现数据传输,也能使用加载/存储指令实现主从核之间的数据传输。

神威新一代超级计算机软件系统针对科学与工程计算、人工智能、大数据等新型应用的需求进行设计,包括基础软件、并行操作系统、并行语言环境、并行开发环境、AI 支撑软件、应用支撑系统等组成部分。

1.2 Palabos 程序结构

Palabos 实现的 LBM 算法,将连续介质看作大量位于网格节点上的离散流体质点粒子,粒子按照碰撞和迁移规则在网格上运动,通过对各网格流体质点及运行特征进行统计,以获得流体的宏观运动规律。LBM 算法中最经典的求解器为 D3Q19,主要面向大规模工程计算,因此首先选取三维 D3Q19 模型进行众核优化,之后对多相流求解器进行优化。

LBM 具有天然的并行性,其核心的逻辑体现为求解大型线性代数方程。从计算机的角度来看:首先根据需求,定义变量的数据结构,用于存储每个格点各个方向的平衡分布数据和每个格点的速度、密度等宏观参量;其次由这些宏观参量计算出各个方向上的平衡分布态函数,以此作为计算的初场;之后开始迭代,使用理论方法计算格点的状态;迭代完成之后根据分布态函数求得速度、密度和流函数等宏观参量,并进行结果输出。

Palabos 作为一款使用 C++语言编写的软件程序,存在面向对象这一特性:将网格上的一个点用于表示粒子不同方向上的速度,将其抽象为一个整体并作为一个对象类存储,即计算网格由一个个格点对象类组成,格点对象组成类对象数组;软件中的函数方法存在多种函数名相同而参数不同的重载函数;软件的模块化编程与层次调用。若要实现程序在新一代神威超算上的优化加速,则必须理清程序调用路径才能保证众核程序的正确性,而 C++的继承和多态影响了对调用关系的梳理,因此需要从数十种名称相同的函数方法中寻找到该模块所调用的程序路径。解决的办法是添加打印执行,根据打印结果判断调用路线(见图 3)。

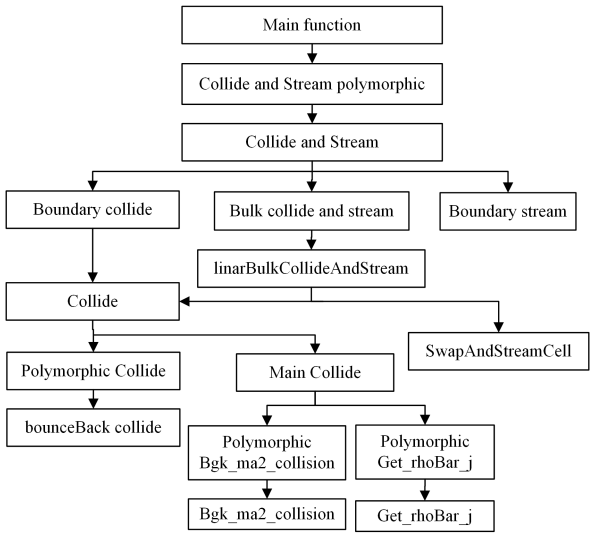


图 3 Palabos 计算主体的调用路径

Fig. 3 Call path of Palabos computing body

2 面向新一代神威 Palabos 的优化方法

Palabos 软件[4]是一款计算流体力学的科学计算软件,在优化过程中提前设计好研究思路,将会减小整个优化的工作量。目前的研究思路是,在移植的同时理解 Palabos 软件用到的数学物理方程,寻找相关简单物理过程的代码,理解软件案例,之后对软件程序进行热点分析,制作程序调用关系图。分析软件使用 C++语言编写的数据结构、模块化编程及面向对象编程思想,结合神威计算机特征,适配硬件特性,设计并行算法,开展性能优化(见图 4)。

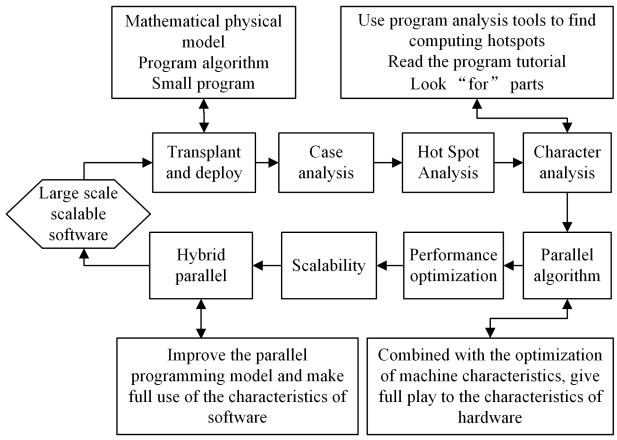


图 4 Palabos 优化的流程

Fig. 4 Optimization process of Palabos

Palabos 计算程序的核心过程包含多个嵌套循环,最外层为迭代时间步,在每个时间步内计算多个嵌套循环。由于神威计算机从核 LDM 空间的限制,无法存储过多的数据,因此每一步迭代我们都采用数据获取-处理-数据写回的方式,使用神威加速卡进行局部的并行处理。为了保持软件自身的并行性和扩展性,我们尽可能地在底层修改,通过打补丁的形式将函数热点段代码进行重写,改造成使用神威加速卡加速的众核程序代码。当前神威高性能计算机的从核芯片不能很好地支持 C++语言的使用,这意味着使用从核加速程序的计算需要使用 C 语言重新实现 C++的程序代码。在工程实践过程中,我们总结了数据切片、数据结构标记、直接取址、函数降阶和利用新一代神威共享空间的优化方法。

2.1 碰撞和迁移过程在众核上的切割

神威众核优化最关键的步骤是将数据切割后合理分配到 64 个从核中。LBM 主要的计算步骤为碰撞和迁移过程,对碰撞和迁移的切割成为了优化的前提。

LBM 的碰撞过程只涉及格点内部数据之间的计算,碰撞过程中各格点的数据互相独立,可并行执行。迁移过程涉及同一行、上下两列的数据交换。最底层的代码如表 1(以二维代码为例)所列。

表 1 碰撞迁移过程的原始代码

Table 1 Original code of collide and stream

```
for (plint iX=domain.x0;iX<=domain.x1;++iX) {
    for (plint iY=domain.y0;iY<=domain.y1;++iY) {
    grid[iX][iY].collide(this->getInternalStatistics());
    latticeTemplates<T,Descriptor>::swapAndStream2D(grid,iX,iY);
}
}
```

3 个点之间的数据交换过程中，iX，iY，iZ 为格点的坐标，$iPop$ 所代表的偏移为格点内部的点，流动过程涉及 iX，iY，iZ 和 nX，nY，nZ 之间的数据交换，首先需明确该过程为交换，非单向之间的数据传递，即两个点之间存在数据相关性。为了完成碰撞和流动过程的并行，我们将原程序中一步操作完成的碰撞和流动过程拆分为两步，分别执行碰撞和流动过程，具体代码如表 2 所列（以二维代码为例）。

表 2　二维碰撞迁移过程拆分代码

Table 2　Split code of collide and stream

```
for (plint iX=domain.x0;iX<=domain.x1;++iX) {
    for (plint iY=domain.y0;iY<=domain.y1;++iY) {
    grid[iX][iY].collide(this->getInternalStatistics());
    }
}
for (plint iX=domain.x0;iX<=domain.x1;++iX) {
    for (plint iY=domain.y0;iY<=domain.y1;++iY) {
      latticeTemplates<T,Descriptor>::swapAndStream2D(grid,iX,iY);
    }
}
```

由于芯片自身的限制，只能存储 3.2 万个负浮点类型的数据。当前一个类对象占据 19 个 double 类型变量、8 字节标记类型及填充、8 字节的函数指针，占据 21 个 double 类型的地址空间。虽然存在内存限制，但是每一个从核应尽可能地多分配计算量，否则会导致特定计算规模被分配到更多的 MPI 进程上，造成优化效果下降。

在执行三维问题时，要保证主核所分配到的数据能在从核中存储，从核内存的大小限制了 MPI 层所能分配给主核的最大数据量。在三维计算中在保证从核能存储下时分配的数据规模只有 50 * 50 * 50。在大规模测试中，因单节点分配的数据规模过小，计算相同规模的数据需要使用更多的节点，通信开销剧增，直接导致时间的倒加速。且数据量为 50 * 50 * 50 时，无法很方便地为每个从核分配数据计算，存在计算资源的浪费。

为了提高优化版本在大规模并行下的执行效率，让单核节点能分配到的数据量尽可能大，分别将 D3Q19 过程的数据切割成单独的 X 轴维度的流动、单独的 Y 轴维度的流动和单独的 Z 轴维度流动（见图 5）。三维层面的流动过程变为 3 次二维层面的流动过程（图 5 中黄色的点代表每一面的计算所需），如此可保证从核能拉取足够的数据。

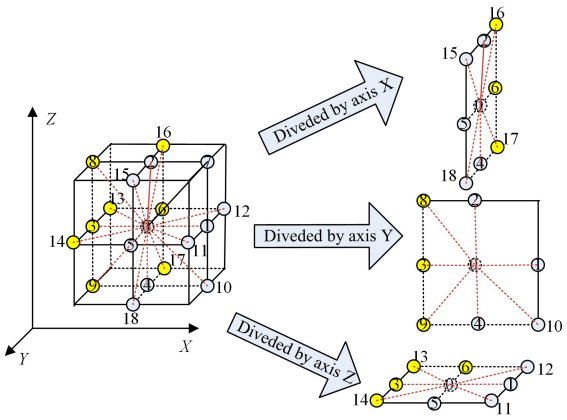


图 5　数据切片（电子版为彩图）

Fig. 5　Data slicing

将 1 次三维数据的遍历变为 3 次二维数据的遍历，且因为数据为类对象和数组结构体形式，所需的有效数据不连续，所以采取全获取的方式获取格点的全部数据，获取的数据量是有效数据量的 3 倍。数据切片解决了 LDM 空间较小的问题，但是还存在 3 次遍历带来的数据的冗余存储、迁移过程中的数据交换的问题。

数据切片后，经过测试，MPI 层分配给每个主核划分的数组最长为 180 左右，即 X 轴、Y 轴、Z 轴所分配的数组最长为 180，较原有 50 * 50 * 50 的计算规模提升了 3.6 倍。后续的计算中，我们选取 180 为 MPI 层分配给主核的数据，主核获取到这些数据之后，将计算分发给自身的 64 个从核处理，在从核上进行二级并行。

在数组长度为 180 时，将数据均分给 64 个从核，每个从核需要处理的行数为 3 或 4。单从核迭代中保证 3 或 4 行数据的正确性，分配到的数据首行和尾行需要与相邻行数据交换，以维持本行数据的正确性。这里采用冗余存储，从核获取 $N+2$ 行数据，即首行和尾行多取 1 行相邻的数据。在一次迭代中，首行与尾行数据皆有数据与之交换，为维持正确性，多取的 1 行数据不写回，其正确性由其他从核保证（见图 6）。

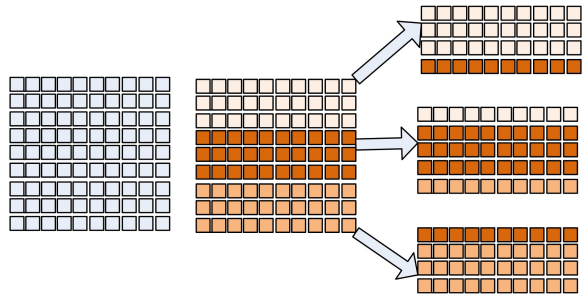


图 6　从核数据划分

Fig. 6　Data division in CPEs

通过上述方法解决了 LBM 中碰撞迁移过程的并行化，且保证了程序的正确性。所带来的缺点是冗余存储了 2/3 的数据量，额外增加了通信时间，但对于并行而言，神威芯片的从核被有效地利用了起来。经过测试，在修改前后，相对于源代码存在加速效果。在相同的数据规模和并行情况下，主从核计算时间较单次迭代主核计算缩短为原来的 1/2。

2.2　函数调用与直接取址

程序实现 LBM 的碰撞和流动算法，通过对象名->方法名调用执行，所调用方法的实现又存在其他更底层的方法调用，整个程序存在一种嵌套的调用关系，即模块化调用。

例如，二维方腔流碰撞流动过程的调用执行如表 3 所列。

表 3　模板化碰撞过程代码

Table 3　Template code of collide

```
for (plint iX=domain.x0;iX<=domain.x1;++iX) {
    for (plint iY=domain.y0;iY<=domain.y1;++iY) {
        grid[iX][iY].collide(this->getInternalStatistics());
      grid[iX][iY].revert();
    }
}
```

外层二维循环遍历调用 Grid[iX][iY].collide()执行格点内部的碰撞过程，revert 函数执行格点间的流动过程。往下寻找 collide 与 revert 的方法的调用实现，碰撞过程代码的

子函数如表 4 所列 。

表 4 碰撞过程代码的子函数

Table 4 Subfunction of collide

```
void BGKdynamics<T,Descriptor>::collide
(Cell<T,Descriptor>& cell,BlockStatistics& statistics )
{
  T rhoBar;
  Array<T,Descriptor<T>::d> j;
    momentTemplates<T,Descriptor>::get_rhoBar_j(cell,rhoBar,j);
    TuSqr=dynamicsTemplates<T,Descriptor>::bgk_ma2_collision(cell,
    rhoBar,j,this->getOmega());
  if (cell.takesStatistics()) {
    gatherStatistics(statistics,rhoBar,uSqr);
  }
}
```

由表 4 可知，grid 是类的实例化对象数组，Array，External，Dynamics 等数据结构在 C 语言中无法表述。这里根据类对象中存储的具体数据分析，在类公有方法中打印每个变量的起始地址，根据地址空间的起止、偏移，推断出类对象中的数据类型。最后用 C 语言中的 double，char，int 等类型进行强制转换，解决数据类型不同导致的存储困难。

使用神威 DMA 获取数据，通过首地址＋获取长度＋偏移的方式获取地址中对应位置的值，将该值存储至从核中用 C 定义的数据变量中，解决了碰撞及流动过程所需数据的传递及接收问题。

2.3 数据结构标记

针对 C＋＋中使用的多态因为实例化不同而出现的不同调用，神威众核版本必须使用 C 语言改写。使用 C 语言如何在特定位置执行特定实现的调用，我们采用设计标记字段在实例化时进行标记，在具体调用时通过判断这些标记来调用相关实现。

Palabos 的格点类 Cell 存储格点的状态信息。为了支持程序的并行运行，存在一些辅助变量，其定义如表 5 所列。

表 5 Palabos 的数据结构

Table 5 Data structure of Palabos

Array<T,Descriptor<T>::numPop> f;	
External	external;
bool	takesStat;
Dynamics<T,Descriptor> *	dynamics

Array 中 numPop 为宏定义，根据 Main 函数中调用的模型设定不同的大小，在 D2Q9，D3Q19，D3Q27 中分别为 9，19，27。在本次优化中，我们处理的是 D3Q19 模型，数组 f 的长度为 19，数据类型为 Double；External 变量用于判断该格点是否是扩展格点；bool 类型的 takesStat 用于数据统计，避免 MPI 划分时相邻两数据块之间冗余分配的一行或一列数据重复统计；Dynamics 类为类对象的函数指针，指向 Cell 类中实现的方法(见图 7)。

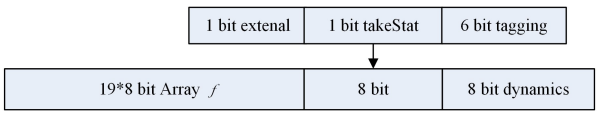


图 7 Palabos 的数据结构

Fig. 7 Data structure of Palabos

软件执行圆柱绕流模拟，对边界柱面的格点执行反弹操作及其他非默认操作时，存在重载函数的调用。对程序而言，核心段部分程序代码未做任何修改，但随着传递的参数不同，因函数的重载，出现了执行不同的函数调用以及程序分支现象。

我们通过直接取址来获取数据，获取到的有效数据的实例化类型并不明确。为了保证程序的正确性，解决软件调用过程中分支路线导致的程序计算结果异常，我们从 Main 函数的初始化开始至核心段分支函数的调用，寻找对象实例化的位置；在软件源码中检索该分支函数名，添加打印输出，通过输出语句判断调用情况。

寻找到程序分支调用的函数实现后，为了明确哪些格点实例化了不同属性，我们向上寻找类对象的实例化，在实例化处为标记字段赋值，与调用关系进行绑定。

同时，我们还面临在众核中使用 C 语言实现 Palabos C＋＋代码的类实例对象方法名的调用。在 C 语言中，失去了类变量这一属性，如何让从核知道在执行到这格点时去执行分支函数的程序代码成为难题。我们的解决方案是：设计标记变量，随格点数据传递至从核中，从核在计算前先判断标记字段，然后选择不同的分支执行。

首先考虑标记变量的定义，如何添加标记字段才能最大程度地保持软件的原生性。根据对 Palabos 数据结构中类变量的数据地址空间的分析，我们选取 6 个填充字节作为存放标记变量的空间，为了保持字节对界，我们选取后 4 位字节定义 int 类型的 flagcc 变量并将其作为标记，同时在其间定义两个字节的填充字段。

最后 Cell 类对象的数据结构如表 6 所列。

表 6 添加多态标记

Table 6 Add polymorphic flag

Array<T,Descriptor<T>::numPop> f;	
External	external;
bool	takesStat;
bool	pad1;
bool	pad2;
int	flagcc;
Dynamics<T,Descriptor> *	dynamics

如此操作，对类变量的修改没有新增地址空间的占用，修改前后，之前所优化的程序无须做任何调整。标记字段添加完毕后，因面向对象编程的特性，我们也需要添加公有的 setFlagcc，getFlagcc 方法，并将其作为变量获取和修改的接口。

之后，在 Cell 类初始化处，用 set 方法为 flagcc 变量设置标记属性，将其与格点类初始化的类型进行绑定。从核通过直接取址的方式获取数据，其只需要将对象指针的首地址偏移至标记字段的首地址，再通过类型强制转换，即可获取标记字段 flagcc 的实际值。通过判断 flagcc 的值，程序执行不同的重载方法，如此便解决了 C＋＋程序多态、函数的重载等问题。添加 if-else 判断选取分支路线保持了程序的正确性，与添加判断之前相比，从核计算时间有所增加，付出的代价是计算时间变为了原来的 1.1 倍。

2.4 高阶差分到低阶差分

在液滴碰撞案例中，Palabos 采取了两阶中心差分，差分

问题的计算是典型的 Stencil 问题，其计算具有强度低且访存不连续的特点。Stencil 计算的性能主要受内存访问延迟和数据低效重用的限制。我们的工作在神威计算机上优化时受到从核地址空间大小、D3Q19 三维计算的数据量以及计算的基本类型为面向对象思想抽象出的基类的限制。

在二阶差分时，数据的偏移量为±2，其离散程度导致 X 轴的数据偏移为 1+2+2，Y 轴偏移为 1+2+2，以及 X 轴、Y 轴确定时的 Z 轴数据，在并行策略为一个从核处理一行数据时，一次获取的数据为 5 * 5 * ARRAY_Z。

当 X 轴确定时，Y 轴和 Z 轴的数据连续，当 X 轴±1 时，其数据不连续，获取这些数据需发起多次 DMA 请求，或一次跨步的 DMA 通信。

为了降低数据的离散性，我们将二阶中心差分降低为一阶中心差分，数据偏移变为±1，一次计算需要获取的数据为 1+1+1(X 轴)，1+1+1(Y 轴)及 Z 轴上的数据。从单从核获取的数据量为 3 * 3 * ARRAY_Z，数据量降低为原来的 9/25，极大地缓解了从核地址空间的紧缺。

程序优化的核心是降低通信开销和计算开销。该差分为某一热点函数的内部调用，需要并行改造该热点函数，其内部调用、处理都需要进行并行化处理。同时，考虑到软件的 MPI 层划分给单个芯片的数据量越大，相同规模下所需的计算节点数少，软件的并行规模就越大。从高阶差分降低为低阶差分，以降低一定精度为代价，降低了数据的离散程度以及内存空间的需求，提高了软件的并行规模。

2.5 LDM 共享空间尝试

LBM 方法中，浸没边界法拉格朗日差值函数产生了一个全局静态数组。并行优化时，主核将数据分发给从核进行处理，程序迭代过程中需要使用拉格朗日差值，将其作为一个固定的静态数组。考虑到从核 LBM 内存所拉取的有效数据应尽可能多，因此使用从核共享空间这一内存分配方式。芯片中的 64 个从核提供部分 LBM 内存空间，将其作为全局共享内存，对该地址空间的读写，所有从核皆可见。拉格朗日插值函数所生成的数据使用静态全局数组存储，迭代过程中该数组只读不写。选择 LDM 共享空间，存储拉格朗日插值数组，很好地支持了从核在计算时获取拉格朗日插值的需求，避免了每个从核各存储一份数据，提高了 LBM 空间的利用效率。LDM 共享空间的设置缺陷在于整个程序的优化过程，从核 LBM 内存空间的分配模式只能设置一次，之后求解器的优化也需要使用 LBM 共享空间，导致一些无共享内存空间使用需求的热点函数在众核优化时所能使用的 LDM 空间变小。

3 测试与分析

3.1 众核加速效率

本文在新一代神威超算上主要对 Palabos 软件的方腔流、圆柱绕流、液滴碰撞和搅拌槽案例进行众核优化。

其中，三维方腔流案例众核版本和主核版本进行单核时间测试的结果表明：众核程序整体加速 2.28 倍，核心部分计算优化时间缩短为原来的 2/7，局部加速最大为 14 倍，计算耗时最大的 collide 过程众核加速效率为 3.66 倍，存在一定的优化空间。非计算部分主要耗时的函数为 executeInternalProcess 过程，每一次迭代占据 20% 的时间开销，难以众核化，需修改程序的处理方式，才能将其优化，提升整体的优化效率。圆柱绕流案例涉及 Bounceback 分支，众核加速倍数为 2.00 倍，较三维方腔流案例加速比略低。液滴碰撞案例原始程序采用 4 阶中心差分，数据离散，从核获取数据不便。百万核心计算网格较密，改为二阶中心差分后误差较小，不影响结果的正确性，同时减小了程序的计算量，有助于众核优化。众核加速倍数为 6.00 倍，核心计算部分加速倍数为 7.74 倍，小循环加速效率较好，大于 30 倍，但是计算量占比 70% 的循环优化倍数为 6.30，限制了整体加速，需要进行进一步优化。搅拌槽案例目前只优化浸没边界法部分，优化后效率为原来的 18 倍，从而该案例整体加速 1.33 倍。

3.2 百万核心加速效率

方腔流案例的众核版本配置进程数为 1 000(约 6.5 万核)、2 000(约 13 万核)、4 000(约 26 万核)、8 000(约 52 万核)和 16 000(约 104 万核)，计算规模为 1 800 * 1 800 * 1 800，迭代 55 步，算例提交运行，程序正常结束后记录运行时间，计算百万核心并行效率为 58.8%，程序整体呈线性加速(见图 8)。

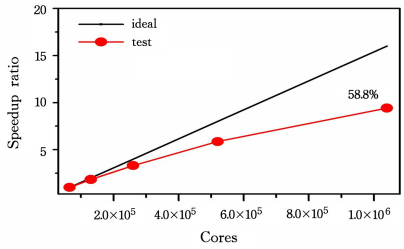


图 8 方腔流强扩展性测试结果

Fig. 8 Strong scaling results of cavity flow case

圆柱绕流案例的众核版本配置进程数为 1 000(约 6.5 万核)、2 000(约 13 万核)、4 000(约 26 万核)、8 000(约 52 万核)和 16 000(约 104 万核)，计算规模分别为 1 500 * 750 * 1 500，1 500 * 750 * 1 500，2 400 * 1 200 * 2 400，3 600 * 1 800 * 3 600，3 600 * 1 800 * 3 600，模拟真实演化时间为 0.11s，算例提交运行，程序正常结束后记录运行时间；计算百万核心并行效率为 82.5%(弱扩展)(见图 9)。同时，该部分程序因为弱扩展性以及计算规模的变化，并未呈现线性加速。

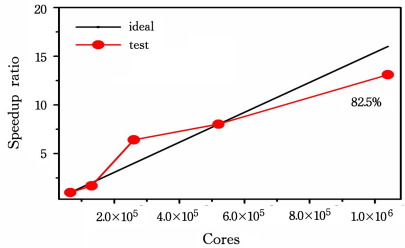


图 9 圆柱绕流弱扩展性测试

Fig. 9 Scaling results of circular cylinder case

液滴碰撞的众核版本配置进程数为 1 000(约 6.5 万核)、2 000(约 13 万核)、4 000(约 26 万核)、8 000(约 52 万核)和 16 000(约 104 万核)，计算规模为 1 600 * 1 600 * 1 600，迭代 20 步，算例提交运行，程序正常结束后记录运行时间，计算百万核心行的效率为 94.33%(见图 10)。因为液滴碰撞案例

规模为 1 600 * 1 600 * 1 600，在 1 000 MPI 进程下单 MPI 网格的大小为 160 * 160 * 160，从核内存基本占满。随着进程数的增加，从核内存存储达到最优，因此测试时部分点的计算效率大于 100%。

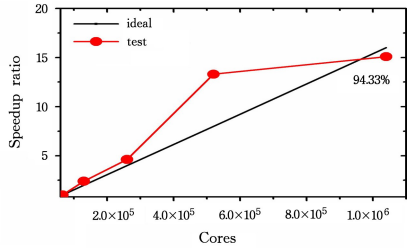


图 10 液滴碰撞强扩展性测试

Fig. 10 Strong scaling results of colliding bubbles case

将优化后的液滴碰撞案例在新一代神威超级计算机上的执行时间与在通用超级计算机上的执行时间进行比较，结果显示，神威上优化的液滴碰撞案例众核优化版本同样计算量的计算时间相比上一代 Intel 通用架构超级计算机缩短了 2/7～5/9(见图 11)，这表明新一代神威超算众核优化后的程序对 Intel 体系架构超级计算机有显著优势。

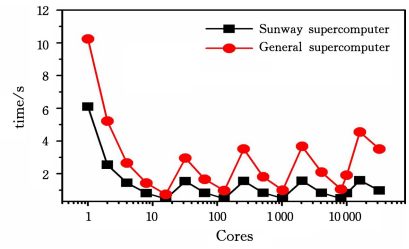


图 11 液滴碰撞案例众核版本在新一代神威超算和通用超级计算机上的计算时间测试对比

Fig. 11 Comparison of computing time test between new generation Shunway supercomputer and general supercomputer about colliding bubbles case

搅拌槽的众核版本配置进程数为 1 000(约 6.5 万核)、2 000(约 13 万核)、4 000(约 26 万核)、8 000(约 52 万核)和 16 000(约 104 万核)，计算规模为 500 * 500 * 500，迭代 200 步，算例提交运行，程序正常结束后记录运行时间，计算百万核心行的效率为 41.00%(见图 12)。由于搅拌槽采用 LDM 共享模式，数据存储开销较大，因此百万核心计算网格数仅为 1.25 亿。同时搅拌槽涉及物理过程复杂，导致百万核心并行效率仅有 41%，但达到了 30%的任务指标，同时该案例基本呈线性加速。

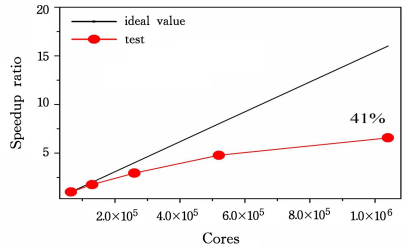


图 12 搅拌槽强扩展性测试

Fig. 12 Strong scaling results of mixer tank case

结束语 本文介绍了在神威众核计算机上对 LBM 程序中热点段的性能进行优化的过程，通过对数据的直接取址，解决了 C++数据与 C 数据类型不一致的问题；利用类对象中地址空间的填充字节，设置标记字段解决 C++中多态导致的分支，从而解决从核中路径调用的问题。由一次三维迭代计算变为多次二维迭代计算，解决从核 LDM 空间无法存储三维迭代计算所需数据的问题。将碰撞和流动过程拆分以解决数据依赖导致无法并行的问题。高阶差分降为低阶差分，降低了数据的离散性，虽提高了从核计算量，但降低了通信开销。本文完成了三维方腔流、圆柱绕流、液滴碰撞和搅拌槽案例热点函数使用的求解器的众核化工作，其中局部加速平均 20 倍，整体加速效果为 2～6 倍。本工作完成了计算流体力学软件 Palabos 在神威超算的移植优化和百万核心并行测试，能够服务于使用 LBM 的计算流体力学的从业人员。

参 考 文 献

[1] LV X J, LIU Z, CHU X S, et al. Extreme-scale simulation based LBM computing fluid dynamics simulations [J]. Computer Science, 2020, 47(4): 13-17.

[2] LIU Z, CHU X S, LV X J, et al. SunWayLB: Enabling extreme-scale Lattice Boltzmann Method based computing fluid dynamics simulations on Sunway TaihuLight [C]// 2019 IEEE International Parallel and Distributed Processing Symposium (IPDPS). IEEE Computer Society, 2019: 557-566.

[3] LIU Z X, FANG Y, SONG A P, et al. Large-Scale scalable parallel computing based on LBM with multiple-relaxation-time model [J]. Journal of Computer Research and Development, 2016, 53(5): 1156-1165.

[4] TIAN M, GU W, PAN J, et al. In Performance analysis and optimization of palabos on petascale sunway BlueLight MPP supercomputer [C]// International Conference on Parallel Computing in Fluid Dynamics. Springer, 2013: 311-320.

[5] OBRECHT C, KUZNIK F, TOURANCHEAU B, et al. The TheLMA project: a thermal lattice Boltzmann solver for the GPU [J/OL]. Computers & Fluids, 2014, 54: 118-126. https://doi.org/10.1016/j.compfluid.2011.10.011.

[6] YE H, SHEN Z, XIAN W, et al. OpenFSI: A highly efficient and portable fluid－structure simulation package based on immersed-boundary method [J/OL]. Computer Physics Communications, 2020: 107463. https://doi.org/10.1016/j.cpc.2020.107463.

[7] BONACCORSO F, MONTESSORI A, TIRIBOCCHI A, et al. LBsoft: a parallel open-source software for simulation of colloidal systems [J/OL]. Computer Physics Communications, 2020: 107455. https://doi.org/10.1016/j.procs.2017.05.084.

[8] ZAVODSZKY G, VAN ROOIJ B, AZIZI V, et al. Hemocell: a high-performance microscopic cellular library [J/OL]. Procedia Computer Science, 2017, 108: 159-165. https://doi.org/10.1016/j.procs.2017.05.084.

[9] HASERT M, MASILAMANI K, ZIMNY S, et al. Complex fluid simulations with the parallel tree-based lattice Boltzmann solver

Musubi[J]. Journal of Computational Science,2014,5 (5):784-794.

[10] LATT J,MALASPINAS O,KONTAXAKIS D,et al. Palabos: Parallel Lattice Boltzmann Solver[J]. Computers & Mathematics with Applications,2021,81(1):334-350.

[11] LATT J,CHOPARD B. VLADYMIR—a C++ matrix library for data-parallel applications[J]. Future Generation Computer Systems,2004,20(6):1023-1039.

[12] MOHAMMADREZAEI S,SIAVASHI M,ASIAEI S. Surface topography effects on dynamic behavior of water droplet over a micro-structured surface using an improved-VOF based lattice Boltzmann method[J/OL]. Journal of Molecular Liquids,2022:118509. https://doi.org/10.1016/j.molliq.2022.118509.

[13] XIA T,FENG Q,WANG S,et al. A numerical study of particle migration in porous media during produced water reinjection[J/OL]. Journal of Energy Resources Technology,2022,144 (7):073002. https://doi.org/10.1115/1.4052165.

[14] ZAVODSZKY G,VAN ROOIJ B,CZAJA B,et al. Red blood cell and platelet diffusivity and margination in the presence of cross-stream gradients in blood flows[J/OL]. Physics of Fluids,2019,31(3):031093. https://doi.org/10.1063/1.5085881.

[15] KOTSALOS C,LATT J,CHOPARD B. Bridging the computational gap between mesoscopic and continuum modeling of red blood cells for fully resolved blood flow[J/OL]. Journal of Computational Physics, 2019, 398: 108905. https://doi.org/10.1016/j.jcp.2019.108905.

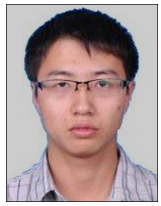
[16] KOTSALOS C,LATT J,BENY J,et al. Digital blood in massively parallel CPU/GPU systems for the study of platelet transport[J/OL]. Interface Focus:a Theme Supplement of Journal of the Royal Society Interface, 2021, 11 (1): 20190116. https://doi.org/10.1098/rsfs.2019.0116.

[17] BOUDJELTIA K Z,KOTSALOS C,DRIBEIRO D,et al. Spherization of red blood cells and platelet margination in COPD patients[J]. Annals of the New York Academy of Sciences,2021,1485(1):71-82.

[18] LIU Y,LIU X,LI F,et al. In Closing the "quantum supremacy" gap:achieving real-time simulation of a random quantum circuit using a new Sunway supercomputer[C/OL]// Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. IEEE Computer Society, 2021. https://doi.org/10.48550/arXiv.2110.14502.

[19] XIAO J,CHEN J,ZHENG J,et al. In Symplectic structure-preserving particle-in-cell whole-volume simulation of tokamak plasmas to 111.3 trillion particles and 25.7 billion grids[C/OL]// Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. IEEE Computer Society, 2021. https://doi.org/10.1145/3458817.3487398.

[20] SHANG H,LI F,ZHANG Y,et al. In Extreme-scale ab initio quantum raman spectra simulations on the leadership HPC system in China[C/OL]// Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. IEEE Computer Society,2021. https://doi.org/10.1145/3458817.3487402.



LIU An-jun, born in 1990, Ph.D. His main research interests include parallel computing and mass/momentum/heat transfer.

YANG Mei-hong, born in 1966, postgraduate, professor, Ph.D, supervisor. Her main research interests include cloud computing, big data and software engineering.

(责任编辑:喻藜)