中国科学: 信息科学 2015 年 第 45 卷 第 3 期: 385–397
www.scichina.com info.scichina.com
《中国科学》杂志社
SCIENCE CHINA PRESS

论 文

数值模拟领域并行编程模型的要素与实例研究

莫则尧*, 张爱清, 刘青凯, 曹小林

北京应用物理与计算数学研究所计算物理重点实验室, 北京 100094
* 通信作者. E-mail: zeyao_mo@iapcm.ac.cn

收稿日期: 2014–05–28; 接受日期: 2014–08–23; 网络出版日期: 2015–02–11
国家自然科学基金 (批准号: 91430218, 11372049)、国家重点基础研究发展计划 (973)(批准号: 2011CB309702) 和国家高技术研究发展计划 (863)(批准号: 2012AA01A309) 资助项目

摘要 本文面向高性能数值模拟, 分析了通用并行编程模型的薄弱环节, 凝练形成了通用并行编程模型栈. 在此基础上, 提出领域并行编程模型, 讨论了模型的数据结构、计算模式、构件模型、编程框架等构成要素和要素间的内在关联, 并以 JASMIN(J's Adaptive Structured Mesh application INfrastructure) 框架为例, 验证了模型的技术可行性和有效性. 领域并行编程模型将显著提升高效并行应用软件的研发效率, 具有重要意义.

关键词 数值模拟 应用软件 领域并行编程模型 并行计算模型 JASMIN 框架

1 引言

科学与工程数值模拟是当前高性能计算的重要应用领域 [1~3]. 随着数值模拟向多时空尺度、多物理过程、强非线性耦合和三维真实构型的复杂化发展 [4~6], 以及高性能计算机在体系结构、编程模型和运行时状态等方面的复杂化发展 [7,8], 并行计算需要解决双重复杂化带来的新问题 [8~10], 支持应用软件与高性能计算机 (以下简称 "计算机") 的协调发展.

数值模拟应用领域专家 (以下简称 "领域专家") 通常从计算机研制人员提供的并行编程模型入手 [11,12], 理解计算机体系结构、设计并行算法、研制并行程序、开展性能优化, 从而掌握 "结构 — 算法 — 编程 — 优化" 的并行计算研究思路 [11]. 并行编程模型揭示的高性能特征越多, 应用软件获得高效率的潜力就越大 [13]. 当前, 计算机峰值性能已超过每秒千万亿次 [14], 体系结构呈现出多层嵌套的发展趋势, 单一的并行编程模型已经很难适应. 例如, 消息传递模型 MPI[15] 不能识别结点内 CPU 间和 CPU 内的访存差异, 共享存储模型 OpenMP[16] 和 Pthread[17] 无法识别结点内 CPU 间的非均匀访存 (NUMA), 尚缺少模型来揭示存储墙 [8,18]、通信墙 [19]、可靠墙 [20] 对性能的影响. 并行编程模型需要由单一层次发展到多层, 形成嵌套栈, 才能更好地匹配于高性能特征的层次化, 适应日趋复杂的计算机体系结构 [13].

可编程性是并行编程模型的另一个设计目标. 遗憾的是, 并行编程模型揭示高性能特征的层次越多, 可编程性和可移植性通常就越弱 [13]. 凝练和集成领域的高性能计算需求, 研发领域并行编程模型, 支持领域专家在无需理解高性能特征的情形下研发应用软件, 是当前攻克编程墙、实现高效能编程的

引用格式: 莫则尧, 张爱清, 刘青凯, 等. 数值模拟领域并行编程模型的要素与实例研究. 中国科学: 信息科学, 2015, 45: 385–397, doi: 10.1360/N112013-00197

技术途径之一 [13,21]. 区别于领域并行编程模型, 称计算机研制人员提供的并行编程模型为通用并行编程模型.

并行应用编程框架 (Framework) 是实现高效能编程的重要技术途径 [21,22]. 在非数值计算领域, 已有多个实例, 例如, 屏蔽数据搜索复杂性的 MapReduce 模型 [23]、屏蔽图相关计算复杂性的 Pregel 框架 [24] 和 GraphLab 框架[1]; 在数值计算领域, 也有类似的工作, 包括 SAMRAI[2], Uintah[25], PETSc[3], PARAMESH[26], UG[4]等. 我国自主研发的 JASMIN 框架 [27,28] 是面向高性能科学与工程计算结构网格而研制的该类框架, 它提供屏蔽并行计算的编程接口, 支持领域专家在个人电脑上 "串行编程" 地研发适应于千万亿次计算机的高效并行应用软件. JASMIN(J's Adaptive Structured Mesh application INfrastructure) 框架的实践表明, 研发领域并行编程模型的技术途径是可行和有效的.

基于以上认识和实践, 本文第 2 节深入分析了当前通用并行编程模型的薄弱环节, 凝练形成了通用并行编程模型栈. 在此基础上, 第 3 节提出了面向高性能数值模拟的领域并行编程模型, 阐述了模型的数据结构、计算模式、构件模型、编程框架等构成要素和要素间的内在关联. 第 4 节以 JASMIN 框架为实例, 验证模型的技术可行性和有效性. 第 5 节总结全文. 领域并行编程模型可将应用软件的研发模式从当前的 "并行设计 — 并行编程 — 手工优化" 提升到 "并行思考 — 串行编程 — 自动优化", 对促进应用软件和计算机的协调发展具有重要意义.

2 通用并行编程模型栈

深刻认识并行编程模型在实际应用中的作用, 需要综合考虑算法设计、编程实现和性能优化. 陈国良等 [29] 立足这 3 个层次, 提出了通用分层并行计算模型和 "结构 — 算法 — 编程" 的并行计算研究思路 [11], 系统地分析了通用并行编程模型. 本节细化通用分层并行计算模型中通用并行编程模型的构成要素, 分析其中的薄弱环节, 凝练形成通用并行编程模型栈.

如图 1 所示, 通用分层并行计算模型由通用并行算法设计模型、通用并行程序设计模型和通用并行程序执行模型构成 [11,29]. 通用并行编程模型是并行程序设计模型的表征, 是应用软件与计算机之间的接口, 它一方面将体系结构的高性能特征揭示给领域专家, 另一方面支持领域专家编程实现并行算法来研制软件. 根据并行编程模型揭示的高性能特征, 并行算法设计模型可用于指导领域专家设计并行算法并对算法的性能进行预测和分析; 同时, 可以结合计算机的运行状态, 对应用软件进行运行前的手工性能优化, 或者调用计算机提供的性能优化工具箱进行运行时性能优化. 性能优化的方法和工具箱是并行程序执行模型的表现形式.

图 1 中, 顶层是计算机体系结构: 计算机由结点组成, 结点间分布存储 (DM); 结点由多个 CPU 组成, CPU 间分布共享存储 (DSM) 非均匀访问 (NUMA); CPU 由多核组成, 多核间对称多处理共享存储 (SMP); 核内包含多级高速缓存 (Cache) 和多个功能部件, 分别支持快速访存和指令级并行 (ILP); 除此之外, 结点还可以配置众核 CPU 或 GPU 作为加速器. 体系结构呈现出 "结点 —CPU— 核 —Cache—ILP" 的多层嵌套和异构加速的高性能特征, 简称之为多层嵌套特征.

通用并行编程模型应该匹配于体系结构的多层嵌套特征. 目前, MPI[15] 支持结点间数据通信, OpenMP[16] 和 Pthread[17] 支持 CPU 内 SMP 编程, OpenCL[5]/OpenACC 支持 SIMD 的 GPU/众核编

1) GraphLab. Distributed Graph-Parallel API Documentation. http://docs.graphlab.org/index.html, 2014.
2) SAMRAI. http://www.lllnl.gov/CASC/SAMRAI, March, 2013.
3) PETSc. http://www.mcs.anl.gov/petsc, March, 2013.
4) UG. http: //cox.iwr.uni-heidelberg.de/∼ug, March, 2013.
5) OpenCL. http://www.khronos.org/registry/cl/specs/opencl-1.1.pdf.

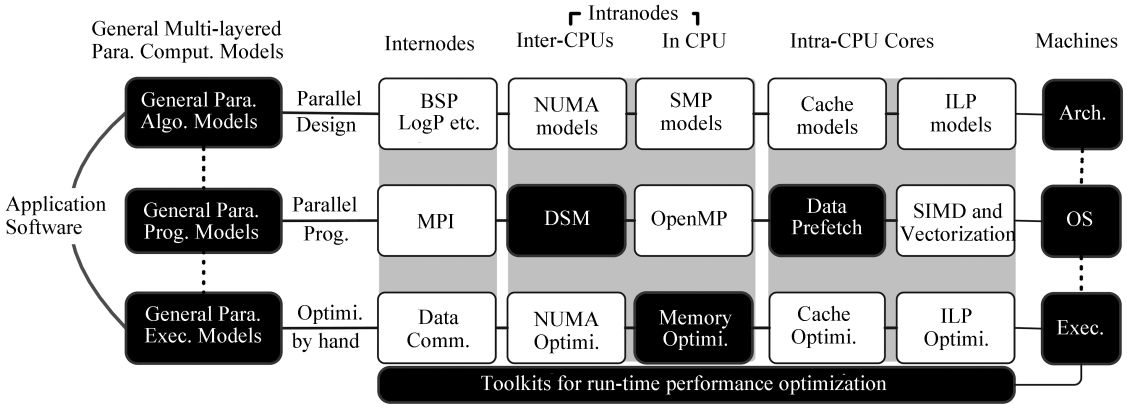

图 1 面向数值模拟的通用分层并行计算模型示图

Figure 1 General multi-layered parallel computing models for numerical simulation

程. 然而, 通用并行编程模型还不能识别结点内 NUMA 特征和支持 Cache 显式调度, 这对已知数据结构和访存行为的实际应用而言, 可能制约计算效率. 例如, 在天河 —1A 千万亿次计算机的结点 (2 个 Intel 至强 6 核 CPU) 上, 武器物理的 6 类典型测试程序 [30] 采用线程绑定和 First-Touch 内存调度策略进行 NUMA 访存优化, 性能提升的幅度可达 10%~50%; 同样的策略在某 32 个 CPU 共 256 核的结点上, 128 个 OpenMP 线程的并行效率可从 10%提升到 60%. 这里, 暂且称结点内揭示 NUMA 访存特征的新型编程模型为 DSM 并行编程模型, 称 Cache 显式调度等访存编程模型为数据预取指导语句.

根据通用并行编程模型揭示的高性能特征, 领域专家可以设计并行算法, 算法的性能可以由通用并行算法设计模型 (即传统的并行计算模型 [31,32]) 来度量. 例如, BSP[33] 和 LogP[34] 及其演化模型可用于度量结点间消息传递, HPM[35] 及其演化模型可用于度量结点内多级 Cache 访存, PRAM 及其演化模型 [11] 可用于度量 SMP 访存竞争. 对结点内 NUMA 访存和核内 ILP 等特征, 尚缺少可度量的模型.

针对通用并行编程模型揭示的高性能特征与计算机运行状态之间的差距, 领域专家可以采用不同类型的方法和工具进行性能优化, 包括 “源到源” 预处理语言 [36]、Intel VTune 性能优化工具[6)]、运行时性能优化工具箱 [37] 等. 随着实际应用和计算机体系结构的双重复杂化, 性能优化凸显重要, 已是国际超级计算年会 Gordan Bell 奖 [38] 的考察重点. 当前, 性能优化可以从数据通信、NUMA 访存、SMP 访存、Cache 访存、指令级并行度等方面入手. 其中, 数据通信大多围绕拓扑结构、通信调度、热点消融、流量平衡等多个方面来进行 [38~40]; 结点内 NUMA 访存可以通过线程捆绑的内存调度来进行; Cache 访存和指令级并行度可以结合数据结构和计算循环来进行; 但对 CPU 内随核数增长日趋严重的 SMP 访存竞争, 目前还缺少有效的方法和工具.

综上所述, 匹配于计算机体系结构的多层嵌套特征, 通用并行编程模型将呈现为多层嵌套的堆叠栈 “MPI—DSM—OpenMP— 数据预取 —SIMD 或向量化”, 称之为通用并行编程模型栈. 目前, 栈内还缺少对 DSM 编程和数据预取的显式支持. 随着计算机体系结构和微电子技术的持续发展, 结点内 CPU 间、CPU 内多核间可能需要更多类型的 DSM 组合, MPI 和 OpenMP 之间可能需要插入更多的通用并行编程模型. 同时, 匹配于栈的发展, 需要同步发展并行算法设计模型和性能优化方法, 研发运行时性能优化工具箱.

6) Intel Vtune Amplifier XE, 2013. http://software.intel.com/en-us/intel-vtune-amplifier-xe.

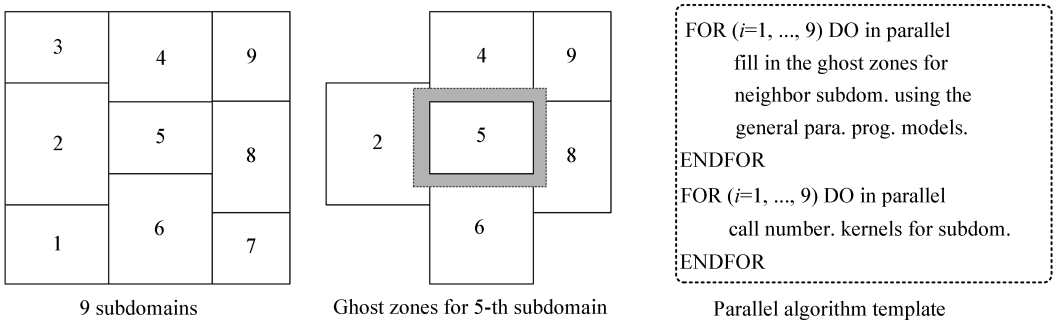

图 2　子区域抽象数据结构和 halo-exchange 并行算法模板示图

Figure 2　Subdomain data structures and halo-exchange parallel algorithm template

3　领域并行编程模型的构成要素

领域并行编程模型应该和通用并行编程模型一样, 是一类可实现的标准和规范的集合. 数据结构和并行算法是获取高性能的关键, 自然是领域并行编程模型的核心, 可以从 4 个层面进行剖析. 第一, 数据结构既要适应于复杂实际应用, 又要匹配于计算机高性能特征, 还要绑定于面向对象的程序设计语言 (例如 C++), 提供标准与规范的编程接口; 第二, 基于数据结构, 针对数值计算中常见的数据依赖关系, 可以凝练形成多种不同形式的计算模式, 提出相应的并行算法模板; 第三, 并行算法模板封装于可复用的构件模型 [41], 绑定到面向对象的程序设计语言, 提供标准与规范的编程接口; 第四, 数据结构和构件模型集成在一起, 形成并行应用编程框架, 具体实现领域并行编程模型. 编程框架提供的编程接口是领域并行编程模型的表征, 它们支持应用软件复用高效的数据结构和并行算法, 大幅降低高性能计算的算法设计与编程难度. 可见, 数据结构、计算模式、构件模型、编程框架等是领域并行编程模型的 4 个构成要素.

3.1　数据结构及其编程接口

数组是数值模拟中普适的数据结构, 基于数组剖分的数据分解策略是并行算法设计的常用策略 [11]. 然而, 数组不适应结点内的多层嵌套特征, 不利于计算模式和并行算法模板的凝练. 在高性能数值模拟领域, 数组通常是物理量的存储格式, 而物理量定义于计算区域的离散网格之上 [1]. 不妨以计算区域替代数组, 计算区域的区域分解替代数组的数据分解.

给定计算区域, 可将其剖分为子区域. 图 2 左端将计算区域剖分为 9 个子区域, 每个子区域由网格离散并定义物理量, 可以开展独立的数值计算. 子区域之间可以相互重叠, 重叠部分通常用于存储数值计算过程中相邻子区域对应位置的物理量值. 以图 2 中间的第 5 号子区域为例, 阴影部分就是重叠部分, 分别用于存储来自第 2, 6, 8, 9, 4 号子区域对应位置的物理量值. 可见, 以子区域为基本单位 (含离散子区域的网格、定义在网格上的所有物理量), 可定义面向对象的抽象数据结构, 提供创建、复制、访问、修改、管理、删除等操作的编程接口. 更进一步, 结合实际应用中数值计算对网格和物理量的声明、创建、管理、修改、删除等方面的需求, 可以将编程接口进行标准化和规范化. 根据抽象数据结构实例化所需的内存容量, 可确定子区域的剖分策略. 子区域可嵌套剖分, 一级剖分的子区域匹配于一级 Cache 结构, 二级和三级嵌套剖分的子区域匹配于更高层次的 Cache 结构.

对比于数组等计算机标准语言提供的数据结构, 子区域抽象数据结构具有数值模拟的领域语义, 既满足复杂实际应用数值计算的需求, 又适应计算机的高性能特征, 符合高效能设计要求. 绑定于面

向对象的程序设计语言, 例如 C++, 它们可由 C++ 类来设计和实现; 结合多层嵌套剖分的子区域, 它们可以实例化为可操作的数据对象, 支持数值计算并承载数值模拟结果. 这里不再示例和讨论.

3.2 计算模式与并行算法模板

基于前述的子区域抽象数据结构, 根据数值计算在子区域之间的数据依赖关系的不同, 可以定义不同形式的计算模式. 图 2 给出了常用的 halo-exchange 计算模式, 即每个子区域首先为物理量填充影像区 (图 2 中间的第 5 号子区域的阴影部分所示, 影像区的数据来自相邻的子区域), 然后调用领域专家编写的子区域数值计算程序在所有子区域之间完成并行计算任务 (图 2 右端所示). 该模式可用并行算法模板描述, 模板的抽象数据结构是子区域, 模板的核心内涵是图 2 右端上部的填充影像区的 DO 循环和右端下部的数值计算子区域 DO 循环控制, 模板的实例化需要调用子区域数值计算程序. 子区域数值计算程序通常可以是串行的, 但如果在子区域内部考虑指令级并行, 例如向量化 SIMD 并行加速, 则需要考虑并行化.

除了图 2 所示的 halo-exchange 计算模式和并行算法模板, 还可以凝练更多类型的计算模式和相应的并行算法模板. Mattson 等 [42] 立足并行程序设计, 凝练了任务并行、分而治之、几何区域分解、数据分解、流水线、事件驱动等多种通用模式, 它们与数值计算的子区域抽象数据结构结合, 可以形成数值模拟的相应计算模式. 例如, halo-exchange 计算模式就是典型的几何区域分解模式, 有向图计算模式 [27] 就是典型的流水线模式, 遍历所有子区域归约某个物理量的归约模式就是典型的分而治之模式, 粒子或能群分解并行的计算模式就是典型的数据分解模式等等. 除此之外, 结合实际应用的数据依赖关系, JASMIN 框架凝练形成了 10 多种典型模式, 将在下节简介.

3.3 构件模型及其编程接口

计算模式和并行算法模板可采用面向复用的构件化软件技术、由构件模型实现 [41]. 以 halo-exchange 模式为例, 构件模型包含图 2 右端上部的填充影像区的 DO 循环、图 2 右端下部的数值计算子区域 DO 循环控制、子区域数值计算串行程序的抽象接口等 3 项内容. 第三项内容分离了子区域数值计算程序的设计与实现, 第二项内容很直观, 第一项内容的实现比较复杂, 在通用并行编程模型栈的不同层次, 需要采用不同的实现技术. 例如, 针对结点间的分布存储、结点内 CPU 间的分布共享存储和 CPU 内的 SMP 共享存储, 影像区可以分别通过消息传递 MPI, DSM 模型和 OpenMP 模型来填充. 但是, 无论哪种实现技术, 对领域专家均是不可见的. 构件模型分离了子区域数值计算程序的研发与并行计算的实现, 只要领域专家提供程序, 满足第三项内容的接口要求, 构件模型就可以实例化为可计算对象, 称之为计算构件 (computational component). 绑定到面向对象的程序设计语言, 构件模型可由抽象类及其策略类 [43] 来实现, 其中, 抽象类实现构件模型, 策略类定义子区域数值计算程序的接口.

3.4 并行应用编程框架

基于数据结构和构件模型的编程接口, 可以研发不同类型和不同功能的计算构件并将其组装为应用软件. 数据结构和构件模型封装在一起, 可以形成并行应用编程框架. 编程框架的编程接口主要由数据结构和构件模型的编程接口组成, 可以表征领域并行编程模型, 也就是说, 领域并行编程模型可由编程框架具体实现. 编程框架之所以称为 “框架 (Framework)”, 主要是因为构件模型封装了子区域数值计算, 它们表示了领域相关的上层业务逻辑 [41]. 当前, 框架的典型代表包括 JASMIN[27,28], SAMRAI,

UTAH[25], PETSc, PARAMESH[26], UG 等. 在这些框架中, 只有 JASMIN 框架完整地分离了应用软件的研发和并行计算的实现.

除了以上介绍的计算模式和构件模型, 还有一类计算模式, 它们是数值模拟中具有完整数值计算功能的解法器, 例如稀疏线性代数方程组解法器、快速多极子解法器、第一原理计算中的 Kohn-Sham 方程解法器、Possion 方程解法器等等. 这类解法器也可以形成构件模型, 用于高效数值算法的复用, 称之为解法器计算模式 (solver pattern of computation), 称相应的构件模型为解法器构件模型. 以示区别, 称前述计算模式为内核计算模式 (kernel pattern of computation), 相应的构件模型为内核构件模型, 它们是数值模拟领域中不可再分的、最基本的计算模式和构件模型. 更进一步, 解法器构件模型可以集成为数值计算构件库, 由领域专家共享, 进一步丰富领域并行编程模型和编程框架. 通常地, 解法器构件模型可以根据数值算法的流程, 通过选择、实例化和组装内核构件模型来实现.

基于领域并行编程模型和编程框架, 领域专家可以将计算机体系结构抽象为 PRAM 模型 [11], 其中, 数据单元等价于子区域, 数据单元的读写访问等价于影像区的填充, 数据单元的数值计算等价于子区域的数值计算. 由此, 可以结合数据填充和数值计算的执行开销, 类似地建立领域 PRAM 模型, 用于评价并行算法的性能, 指导 “并行思考”.

基于领域并行编程模型和编程框架, 领域专家可以结合实际应用的需求, 选择数据结构来定义物理量, 选择构件模型并编写子区域数值计算程序将其实例化为计算构件, 通过计算流程将构件组装为应用软件. 其中, 并行计算由数据结构和构件模型来实现, 领域专家无需了解其实现细节. 也就是说, 领域并行编程模型可以将并行计算的实现从应用软件的研发中分离出来, 将应用软件的研发模式从基于通用并行编程模型的 “并行设计 — 并行编程” 提升到 “并行思考 — 串行编程”.

基于领域并行编程模型和编程框架, 应用软件的计算效率由内核构件和解法器构件决定. 无论内核构件, 还是解法器构件, 它们的计算效率可以分解为两个部分, 第一是构件模型, 第二是子区域数值计算程序. 构件模型由框架研制人员结合计算机体系结构的多层嵌套特征、基于通用并行编程模型来编写, 计算效率可以验证和保证; 子区域数值计算程序由领域专家编写, 计算效率较难验证和保证. 为了支持领域专家编写高效的子区域数值计算程序, 可以结合数据依赖关系, 凝练其中耗时大的浮点、访存、通信、I/O 等计算行为, 凝练形成串行计算模板, 提供相应的 “源到源” 模板语言. 基于串行计算模板, 领域专家无需逐循环、逐行、逐表达式地编写代码, 只需填写模板所需的内容 (例如数学公式), 就可以由 “源到源” 模板语言将模板代码转变为适应于 CPU 核体系结构的源代码. “源到源” 模板语言是高性能计算的前沿热点之一, 已有较多的成功典范 [36]. “源到源” 模板语言可以进一步成为领域并行编程模型的组成部分, 提升模型的支撑能力. 可见, 基于编程框架的构件模型和 “源到源” 模板语言, 应用软件可以将性能从 “手工优化” 提升到 “自动优化”.

为了深入理解领域并行编程模型, 类比于通用分层并行计算模型, 图 3 给出了领域分层并行计算模型. 该模型由领域并行算法设计模型、领域并行程序设计模型和领域并行程序执行模型构成. 其中, 领域并行算法设计模型的内涵是领域 PRAM 模型, 是对计算机体系结构的领域抽象建模; 领域并行程序设计模型和领域并行程序执行模型的内涵是领域并行编程模型. 数据结构和并行算法模板构成 PRAM 模型, 并行应用编程框架和解法器构件库实现领域并行编程模型, “源到源” 模板语言支持子区域数值计算性能优化, 三者一起构成了面向高性能数值模拟领域的高性能科学与工程计算中间件. 中间件的发展是长期的, 需要在实际应用中凝练、完善和验证.

综上所述, 领域并行编程模型通过对计算机体系结构的 PRAM 抽象建模, 屏蔽了日趋复杂的多层嵌套特征, 极大地降低了领域专家的算法设计难度; 同时, 在中间件的支持下, 应用软件又可以很好地适应高性能计算. 对比于通用并行编程模型, 领域并行编程模型可以支持领域专家按 “结构 — 算法

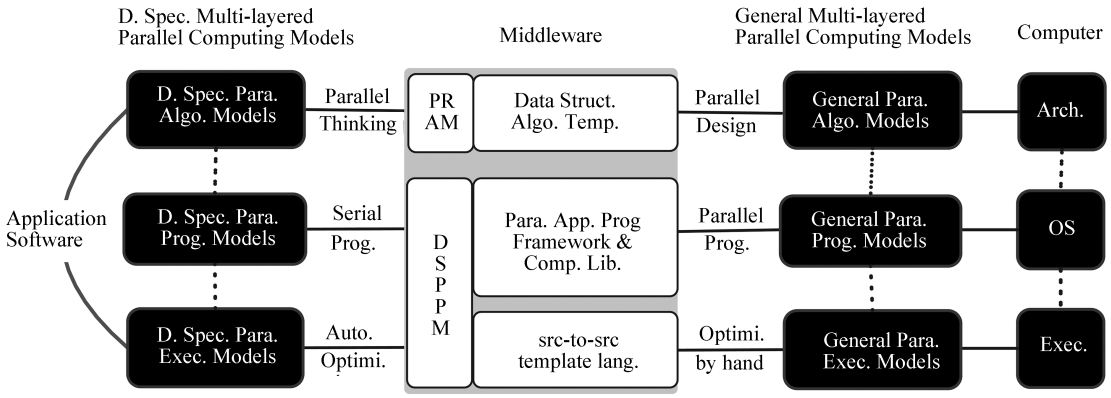

图 3 面向数值模拟的领域分层并行计算模型示图

Figure 3 Domain-specific multi-layered parallel computing models for numerical simulation

— 编程 — 优化” 的思路来掌握并行计算, 按 “并行思考 — 串行编程 — 自动优化” 的模式来研发应用软件, 从而普适地提升应用软件的计算效率, 大幅提升应用软件的研发效率.

4 实例研究: JASMIN 框架

网格是科学与工程计算的基础, 用于离散计算区域, 支持数学物理方程的求解. 目前, 网格主要有两种类型 [1,14], 一类是结构网格, 另一类是非结构网格. 目前, JASMIN 框架面向结构网格, 支持最广泛使用的均匀矩形结构网格和变形结构网格以及不同类型的演化网格. 所有网格均支持一维、二维和三维. 基于这些网格, JASMIN 框架支持科学与工程计算中普遍的 Euler, Lagrange、任意 Euler-Lagrange、粒子模拟等 4 类计算方法 [1,44]. 限于篇幅, 这里不做介绍, 请参考用户指南 [28].

下面, 紧密结合 JASMIN 框架的实践, 简要介绍领域并行编程模型设计和编程框架研发的数据结构和内核计算模式, 阐述解法器计算模式和性能优化的主要思路.

4.1 数据结构

JASMIN 框架采用子区域的抽象数据结构. 如图 4 所示, 嵌套剖分将输出子区域层次结构, 其中, 顶层为计算区域和结构网格, 底层为离散和求解方程的子区域, 每个子区域拥有各自的结构网格. 称底层子区域为 “网格片 (Patch)”, 所有网格片构成 “网格层 (Patch Level)”. 网格层中的网格片分配到 CPU 核, 每个核可以拥有 1 个或多个网格片. 在计算区域和底层剖分之间, 可以引入多个层次的剖分, 匹配于计算机体系结构的多层嵌套, 提升多级访存的局部性. 目前, JASMIN 框架引入了两个中间层, 分别对应结点和 CPU, 它们依然可以由 Patch 和 PatchLevel 来定义. 在层次结构中, 中间层次的 Patch 和 PatchLevel 对领域专家是不可见的, 只有底层 Patch Level 和 Patch 对领域专家可见. 在网格层的每个网格片上, JASMIN 框架提供数据片 (PatchData)[27,28], 用于存储物理量的值.

匹配于计算机体系结构的多层嵌套特征和通用并行编程模型栈, 如图 5 所示, JASMIN 框架采用多层嵌套的数据结构 “一层剖分 Patch— 二层剖分 Patch— 底层剖分 Patch— 物理量 PatchData— 单元循环”. 在此基础上, 可以通过 “二层剖分 Patch— 物理量 PatchData— 单元循环” 适应 GPU 或众核异构加速, 也可以在底层剖分 Patch 的单元循环内, 支持 “源到源” 模板语言.

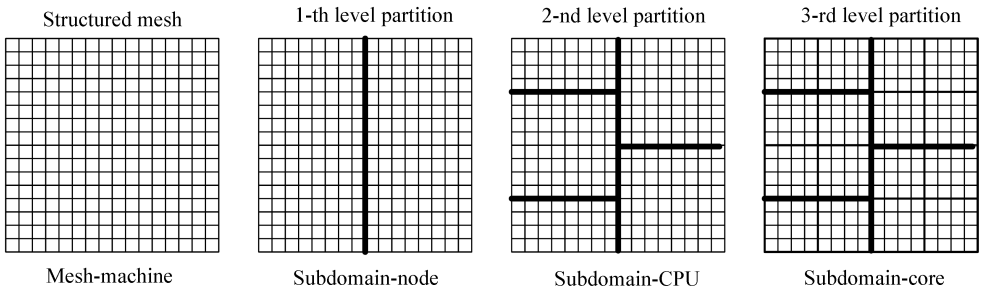

图 4　计算区域及结构网格的 3 层嵌套剖分示图

Figure 4　Three-level partition for structured mesh of computational domain

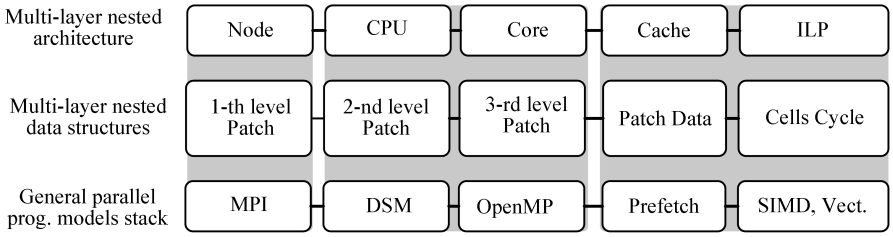

图 5　JASMIN 框架的多层嵌套数据结构示图

Figure 5　Multi-layer nested data structures for JASMIN framework

4.2　内核计算模式与构件模型

内核计算模式如上节定义, 是 JASMIN 框架中不可分割的、最基本的并行计算模式. 通常地, 它们用于完成网格片之间的一次数据交换和依赖于交换的数值计算. 除了图 2 所示的 halo-exchange 模式, JASMIN 框架还凝练了 10 多种内核计算模式, 设计了相应的并行算法模板, 研制了相应的构件模型, 包括: 初值模式、规约模式、扫描模式、迁移模式、影像模式、标记模式、聚集模式、剖分模式、校正模式、接触拼接模式、复制模式、克隆模式等. 初值模式为物理量赋初值, 规约模式求物理量的规约值, 扫描模式执行基于有向图的数据驱动计算 [40], 迁移模式迁移粒子等非规则物理量的个体, 影像模式将影像区的数据填充到网格片内部, 标记模式标记待局部加密或局部优化的网格单元, 聚集模式将标记的网格单元聚集为子区域, 剖分模式将计算区域剖分为网格片, 校正模式在多层网格自适应计算的粗细网格层之间校正物理量, 接触拼接模式支持接触碰撞与滑移 [44], 复制模式将物理量的值复制给另一个物理量, 克隆模式支持粒子和能群之间的独立无关并行计算.

4.3　解法器计算模式与构件模型

解法器计算模式求解某类数值计算问题, 由解法器构件模型实现. JASMIN 框架提供多种解法器计算模式, 涉及负载平衡、数学运算、网格自适应、多物理耦合、数值代数、微分方程、计算方法等多个方面. 其中, 负载平衡剖分计算区域为网格片, 将网格片分配到 CPU 核; 数学运算遍历网格片, 对稀疏矩阵或物理量执行一次 BLAS-2 或 BLAS-1 运算 [45]; 网格自适应局部加密粗网格为细网格或优化网格单元的品质; 多物理耦合在不同物理过程之间填充物理量; 数值代数涉及稀疏线性代数方程组、稠密矩阵特征值、稀疏矩阵特征值等问题, 包含快速 Fourier 变换 (FFT)、快速多极子算法 [46] 等算法; 微分方程包含 Poisson、Euler、粒子运动、第一性原理 Kohn-Sham 方程 [47] 等方程解法器; 计算方

法包含复杂几何计算区域的网格生成方法、大变形网格的网格优化方法、大变形网格重分和物理量重映算法、接触与碰撞中滑移检测与处理算法、多介质界面识别与追踪算法等多个方面 [44].

以上解法器计算模式中, 很多问题是传统的并行计算难题, 性能的扩展需要多个方面的集成创新和关键技术突破. JASMIN 框架通过数据结构、并行算法和软件技术的体系化集成创新, 较好地解决了这些难题. 例如, 网格自适应和多物理耦合需要可迁移数据结构、动态负载平衡方法和全局稀疏数据通信算法的支撑, 在通用并行编程实现中, 很难同时做到这 3 点. 然而, 得益于 JASMIN 框架中网格自适应构件模型的支撑, 三维辐射流体力学软件 LARED-S[7)]在 4096 个 CPU 核上实现了高效计算, 相对于 128 核的单层网格计算, 自适应算法加速了 160 倍, 并行加速了 32 倍, 二者综合将计算规模提升了 5000 倍; 多物理耦合构件模型已广泛应用于武器物理和激光惯性约束聚变等领域, 实现了多介质流体力学、多群辐射扩散、多群辐射输运、多束光路传播等物理过程的耦合, 可高效扩展到 8192 个 CPU 核 [48]; 同时, 多物理耦合构件模型还支撑地球系统模式二维并行耦合器的研制, 支持了大气模式、海洋模式、海冰模式、陆面模式在 TH-1A 上 5800 个 CPU 核的高分辨率耦合模拟 [49], 并行效率达到 33%.

4.4 性能优化与 “源到源” 模板语言

JASMIN 框架的性能优化围绕结点间数据通信、结点内访存竞争、CPU 内 Cache 访存、CPU 核内指令级并行度展开, 其中, 前 3 者由框架结合运行时性能优化工具箱进行, 后者通过减少循环迭代之间的数据依赖关系、降低 Cache 访存失效率、优化寄存器文件来进行. JASMIN 框架支持 “源到源” 模板语言的集成和应用, 用于改进应用软件的跨平台可移植性. 目前, “源到源” 模板语言支持粒子模拟类应用的 GPU 异构编程, 其他模板语言还在进一步发展之中.

4.5 实际应用验证

基于 JASMIN 框架, 领域专家可以在个人电脑上 “并行思考、串行编程” 快速研发适应于千万亿次计算的高效并行应用软件. 目前, 该框架已经在武器物理、激光惯性约束聚变、复杂电磁环境、材料科学、地球资源环境、全球气候变化预测等重大应用领域和第一原理、分子动力学、位错动力学、粒子模拟、流体力学、扩散与输运、冲击动力学、结构力学、电磁效应分析等方向, 支撑研制了 30 多个高效并行应用软件. 它们可以有效使用数千至数万个 CPU 核完成数十个小时、数十亿网格单元、数百亿粒子的大规模数值模拟, 实现了与千万亿次计算机的协调发展. 限于篇幅, 这里不再列出这些程序及其性能结果, 有兴趣者请参考 JASMIN 框架网页 [28] 及其网页中列出的相关参考文献.

5 结论

当代计算机体系结构呈现出 “结点 —CPU— 核 —Cache—ILP” 的多层嵌套和异构加速的高性能特征. 与之相适应, 通用并行编程模型也将呈现出 “MPI—DSM—OpenMP— 数据预取 —SIMD 或向量化” 的多层嵌套发展趋势, 形成通用并行编程模型栈, 其中, 结点内 DSM 编程模型和数据预取编程是当前的薄弱环节. 需要发展性能优化方法, 用于弥补通用并行编程模型与计算机实际运行状态之间的差距. 性能优化可以从数据通信、NUMA 访存、SMP 访存、Cache 访存、指令级并行度等方面入手, 需要研发运行时性能优化工具箱.

7) 徐小文, 范征峰, 刘青凯, 等. 数千 CPU 核上的三维辐射流体力学界面不稳定性数值模拟. http://www.iapcm.ac.cn/jasmin/index.php?page=lareds.

领域并行编程模型分离并行计算的实现与应用软件的研发, 支持领域专家在个人电脑上串行编程研发高效并行应用软件, 从而将应用软件的研发模式从基于通用并行编程模型的 “并行设计 — 并行编程 — 手工优化” 提升为 “并行思考 — 串行编程 — 自动优化”, 可望普适地提升应用软件的计算效率, 显著降低应用软件的研制难度, 为日趋严重的编程墙提供新思路、新方法和新技术, 具有重要的应用价值和广阔的发展前景.

领域并行编程模型包含数据结构、计算模式、构件模型、编程框架等 4 个要素, 由数据结构和构件模型的编程接口表征, 由编程框架具体实现. 基于编程框架, 可进一步研制解法器构件模型库和 “源到源” 模板语言, 它们一起构成高性能科学与工程计算中间件. 中间件的完善和发展是一个逐步积累的过程, 其中, 子区域数据结构是基础, 内核构件模型是核心, 解法器构件模型是必需. 中间件需要借助于通用并行编程模型栈和性能优化方法来适应计算机体系结构的多层嵌套特征.

JASMIN 框架的成功实践已经表明, 面向高性能数值模拟的领域并行编程模型是可行的、有效的. 尽管如此, 普适的领域并行编程模型还需要进一步凝练, 还需要形成新的标准和规范. 在新的标准和规范之下, 可以期待更多或更有效的编程框架的实现.

致谢 本文工作是对北京应用物理与计算数学研究所 JASMIN 框架研究成果的一次提升, 期间科研团队进行了多次研讨; 本文的许多认识得益于与深圳大学陈国良院士的讨论, 陈院士亲自阅读了文稿的初稿并提出了很多宝贵意见. 在此, 表示衷心感谢.

参考文献

1 Dongarra J, Foster I. Sourcebook of parallel computing. Beijing: Electronic Industry Press, 2005 [Dongarra J, Foster I. 莫则尧, 陈军, 曹小林, 译. 并行计算综论. 北京: 电子工业出版社, 2005]
2 Oden J T, Belytschko T, Fish J, et al. Simulation-Based Engineering Science: Revolutioninizing Engineering Science Through Simulation. Arlington: National Science Foundation, 2006
3 Bader D A. Petascale Computing: Algorithms and Applicaitions. New York: Chapman & Hall/CRC, Computational Science Series, 2007 [Bader D A. 都志辉, 译. 面向千万亿次计算的算法与应用. 北京: 清华大学出版社, 2008]
4 Simon H. Exascale challenges for the applied mathematics community. In: Proceedings of DOE Applied Mathematics Program Meeting, Berkeley, 2010
5 Guest M. PRACE: the scientific case for HPC in Europe 2012–2020 from petascale to exascale. 2012
6 Glotzer S C, Kim S, Cummings P T, et al. WETC panel report on international assessment of research and development in simulation-based engineering and science, world technology evaluation center. 2009
7 Kogge P, Bergman K, Borkar S, et al. DAPRA IPTO report on exscale computing study: technology challenges in achieving exscale systems. 2008
8 Yang X J. Sixty years for parallel computing. Chin J Comput Eng Sci, 2012, 34: 1–10 [杨学军. 并行计算六十年. 计算机工程与科学, 2012, 34: 1–10]
9 Sarkar V, Harrod W, Snavely A E. Software challenges in extreme scale system. J Phys, 2009, 180: 012045
10 Harrod W. A journey to exascale computing. Keynote Presnet SC12, 2012, 10–16
11 Chen G L. Parallel Computing-Architecture • Algorithm • Programming. 3rd ed. Beijing: High Education Publisher, 2011 [陈国良. 并行计算: 结构 • 算法 • 编程. 第 3 版. 北京: 高等教育出版社, 2011]
12 Zhang L B, Chi X B, Mo Z Y, et al. Introduction to Parallel Computing. Beijing: Tsinghua Publisher, 2006 [张林波, 迟学斌, 莫则尧, 等. 并行计算引论. 北京: 清华大学出版社, 2006]
13 Amarasinghe S, Hall M, Lethin R, et al. DOE Workshop on Exascale Programming Challenges Report. 2011
14 Zhang Y Q, Yuan G X, Sun J C, et al. Ten years memory of top10 for high performance computers. Chin J Comput Eng Sci, 2012, 38: 11–16 [张云泉, 袁国兴, 孙家昶, 等. 中国高性能计算机 TOP100 十周年回顾与展望. 计算机工程与科学, 2012, 38: 11–16]
15 MPI Forum. MPI: A Message Passing Interface Standard. Version 3.0. 2012

16 OpenMP Forum. OpenMP: Application Program Interface. Version 4.0. 2012
17 Nichols B, Buttlar D, Farrell J P. Pthreads Programming. Sebastopol: O'Reilly & Associates, Inc., 1996
18 Sun X H, Wang D. APC: a performance metric of memory systems. ACM SIGMETRICS Perform Eval Rev, 2012, 40: 125–130
19 Yang X J, Lin Y, Xue J L, et al. The communication wall for exascale supercomputing. IEEE Trans Parall Distr Syst, 2012
20 Yang X J, Wang Z Y, Xue J L, et al. The reliability wall for exascale supercomputing. IEEE Trans Comput, 2012, 61: 767–779
21 Heroux M A. Next generation programming environment: what we need and do not need. In: Proceedings of DOE Workshop on Exascale Programming Challenges, Washington, 2011
22 Vetter J S. Automatic exploration of the HPC co-design space. In: Proceedings of Keynote Presentation on the International HPC Forum on the Exa-Scale Computing, Changsha, 2013
23 Dean J, Ghemawat S. MapReduce: simplied data processing on large clusters. Commun ACM, 2008, 51: 107–113
24 Malewicz G, Austern M H, Bik A J C, et al. Pregel: a system for large-scale graph processing. In: Proceedings of SIGMOD'10, New York, 2010. 135–146
25 Berzins M. Status of Release of the Uintah Computational Framwork. UUSCI Technical Report UUSCI-2012-001. 2012
26 MacNeice P, Olson K M, Mobarry C, et al. PARAMESH: a parallel adaptive mesh refinement community toolkit. Comput Phys Commun, 2000, 126: 330–354
27 Mo Z Y, Zhang A Q, Cao X L, et al. JASMIN: a parallel software infrastructure for scientific computing. Front Comput Sci China, 2010, 4: 480–488
28 Mo Z Y, Zhang A Q, Liu Q K. User's guide to JASMIN: an infrastructure for parallel adaptive structured mesh applications. Institute of Applied Physics and Computational Mathematics, Technical Report J09-JMJL-01. 2011 [莫则尧, 张爱清, 刘清凯. 并行自适应结构网格应用支撑软件框架 JASMIN 用户指南. 北京应用物理与计算数学研究所技术报告 J09-JMJL-01. 2011]
29 Chen G L, Miao Q K, Sun G Z, et al. Multilayered parallel computing models. Acta Univ Chin Sci Technol, 2008, 38: 841–847 [陈国良, 苗乾坤, 孙广中, 等. 分层并行计算模型. 中国科学技术大学学报, 2008, 38: 841–847]
30 Gao X Y, Cao X L, Zhao W B, et al. Performance benchmarks for complex application codes using tens of thousands of CPU cores. Acta Chin Softw, 2011, 22: 157–162 [高兴誉, 曹小林, 赵伟波, 等. 数万核上复杂应用程序的性能测试与分析. 软件学报, 2011, 22: 157–162]
31 Chen G L, Sun G Z, Zhang Y Q, et al. Study on parallel computing. J Comput Sci Technol, 2006, 21: 665–673
32 Zhang Y Q, Chen G L, Sun G Z, et al. Models of parallel computation: a survey and classification. Front Comput Sci China, 2007, 1: 156–165
33 Valiant L G. A bridging model for parallel computation. Commun ACM, 1990, 33: 103–111
34 Culler D, Karp R, Patterson D, et al. LogP: towards a realistic model of parallel computation. In: Proceedings of the 4th ACM SIGPLAN Symposium on Principles and Pratics of Parallel Programming, San Diego, 1993. 1–12
35 Qiao X Z, Chen S Q, Yang L T. HPM: a hierarchical model for parallel computations. Int J High Perform Comput Netw, 2004, 1: 117–127
36 Cheisten M, Schenk O, Burkhart H. Patus: a code generation and autotuning framework for parallel iterative stencil computations on modern microarchitectures. In: Proceedings of 25th IEEE International Parallel & Distributed Processing Symposium (IPDPS 2011), Anchorage, 2011. 676–687
37 Bergan T, Anderson O, Devietti J, et al. CoreDet: a compiler and runtime system for deterministic multithreaded execution. In: Proceedings of ASPLOS'10, Pittsburgh, 2010. 53–64
38 Zhang L L, Deng X G. Analysis for Gordan bell prize. Chin J Comput Eng Sci, 2012, 34: 44–52 [张理论, 邓小刚. 戈登奖 — 分析与思考. 计算机工程与科学, 2012, 34: 44–52]
39 Mo Z Y. Research on key techniques for parallelization and optimization of applied codes. Chin J Numer Math Appl, 2002, 24: 45–58
40 Mo Z Y, Zhang A Q, Wittum G. Scalable heuristic algorithms for the parallel execution of data flow acyclic digraphs. SIAM J Sci Comput, 2009, 31: 3626–3642
41 Huang G, Zhang L, Zhou M H, et al. Design and Implementation for Component-Based Software. Beijing: Tsinghua

Publisher, 2008 [黄罡, 张路, 周明辉, 等. 构件化软件设计与实现. 北京: 清华大学出版社, 2008]
42 Mattson T G, Sanders B A, Massingill B L. Patterns for Parallel Programming. Boston: Addison-Wesley Professional Publisher, 2004
43 Gamma E, Helm R, Johnson R, et al. Design Patterns, Element of Reusable Object-Oriented Software. Boston: Addison-Wesley Professional Computing Series, 2000
44 Li D Y, Xu G R. Computational methodes for two-dimensional non-steady hydrodynamics. Beijing: Science Press, 1987 [李德元, 徐国荣. 二维非定常流体力学计算方法. 北京: 科学出版社, 1987]
45 Baker V A, Blackford L S, Dongarra J, et al. LAPACK95 Users' Guide. Philadelphia: SIAM Publication, 2001
46 Cao X L, Mo Z Y, Liu X, et al. Parallel fast multiple methods in JASMIN framework. Sci Sin Inform, 2010, 40: 1187–1196 [曹小林, 莫则尧, 刘旭, 等. 基于 JASMIN 框架的快速并行多极子并行解法器. 中国科学: 信息科学, 2010, 40: 1187–1196]
47 Fang J, Gao X Y, Zhou A H. A kohn-sham equation solver based on hexahedral finite elements. J Comp Phys, 2012, 231: 3166–3180
48 Zhang A Q, Mo Z Y, Cao X L, et al. Federation computing and multiphysics applications for JASMIN framework. Chin J Comput Eng Sci, 2013, 35: 15–23 [张爱清, 莫则尧, 曹小林, 等. JASMIN 框架中联邦并行计算及其在多物理耦合中的应用. 计算机工程与科学, 2013, 35: 15–23]
49 Zhang A Q. Parallel Application Software Frameworks for Earth Climate Models. IAPCM Technical Report for National 863 Project. 2013 [张爱清. 地球系统模式并行应用框架研制. 国家 863 项目课题技术总结报告. 2013]

Research on the components and practices for domain-specific parallel programming models for numerical simulation

MO ZeYao*, ZHANG AiQing, LIU QingKai & CAO XiaoLin

Institute of Applied Physics and Computational Mathematics, Laboratory of Computational Physics, Beijing 100094*, China*
*E-mail: zeyao_mo@iapcm.ac.cn

Abstract This paper analyzes general parallel programming models and identifies the general parallel programming models stack used for high-performance numerical simulation. Then, on the basis of this analysis, domain-specific parallel programming models are presented and their main constitution, including data structures, computational patterns, component models, and programming frameworks, discussed. The inherent relationship among these elements is also analyzed. The J Adaptive Structured Mesh application INfrastructure (JASMIN) framework is used to verify and validate the possibility and effectiveness of these models. Domain-specific parallel programming models will significantly improve the development efficiency of parallel application software.

Keywords numerical simulation, application software, domain-specific parallel programming models, parallel computing models, JASMIN

MO ZeYao was born in 1971. He received a Ph.D. degree in computer science from the National University of Defense Technology, Changsha, in 1997. He is currently a professor at IAPCM. His research interests include high-performance computing for numerical simulation.

ZHANG AiQing was born in 1976. She received a Ph.D. degree from the Graduate School, China Academy of Engineering Physics, Beijing, in 2008. She is currently an associate professor at IAPCM. Her research interests include high-performance computing for numerical simulation.

LIU QingKai was born in 1976. He received a Ph.D. degree from the Graduate School, China Academy of Engineering Physics, Beijing, in 2005. He is currently a professor at IAPCM. His research interests include high-performance computing for numerical simulation.

CAO XiaoLin was born in 1974. He received a Ph.D. degree from Chengdu University of Technology, Chengdu, in 2000. He is currently a professor at IAPCM. His research interests include high-performance computing for numerical simulation.