中国科学: 信息科学 2016 年 第 46 卷 第 10 期: 1392–1410
SCIENTIA SINICA Informationis
《中国科学》杂志社
SCIENCE CHINA PRESS
高性能科学计算若干前沿问题研究专刊

并行算法与并行编程: 从个性、共性到软件复用

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: 2016–06–03; 接受日期: 2016–07–13; 网络出版日期: 2016–10–25
国家自然科学基金重大研究计划重点项目 (批准号: 91430218)、国家重点研发计划项目 (批准号: 2016YFB0201301)、国防基础科研计划项目 (批准号: C1520110002) 和国家自然科学基金面上项目 (批准号: 11372049) 资助

摘要 面向高性能数值模拟, 并行算法设计与并行编程实现是领域专家在超级计算机上研制应用软件的重要环节. 近十多年来, 随着超级计算机性能的持续提升, 应用问题的求解日趋复杂, 如何发展并行算法与并行编程, 并在应用软件之间实现复用, 支持应用软件高效使用并同步研发超级计算机, 是高性能计算的重要研究内容. 本文结合作者的实践, 阐述并行算法与并行编程研究从个性到共性、再到软件复用的必要性和关键技术. 本文对高性能数值模拟应用软件的研发以及高性能计算的应用研究具有参考价值.

关键词 数值模拟 并行算法 并行编程 应用软件 软件复用

1 引言

并行算法与并行编程是高性能计算应用研究的主要内容. 面向高性能数值模拟, 它们主要用于支持物理建模和计算方法在超级计算机上的设计与实现, 从而支持超大规模并行应用软件的研制与应用. 近十多年来, 我国超级计算机的性能持续提升, 已经从每秒万亿次、十万亿次、千万亿次提升到了亿亿次量级, 体系结构和编程模型发生了很大的变化 [1,2]1). 与此同时, 数值模拟的实际应用也逐步呈现多物理场耦合、多时空尺度、强非线性强间断、多介质大变形、三维复杂几何构型的复杂特征, 超大规模并行应用软件的研发越来越依赖于并行算法与并行编程的创新研究 [3,4]. 本文将这种具备复杂实际应用特征, 同时适应超级计算机发展的超大规模并行应用软件称为超级并行应用软件.

陈国良院士在专著《并行计算 – 结构算法编程》中 [5], 提出了 “体系结构 – 并行算法 – 并行编程 – 并行应用” 的研究体系, 从高性能计算机体系结构、并行算法设计与分析、通用并行编程模型的 3 个方面, 阐述了并行计算的研究基础. 与此同时, 面向实际应用, 并行算法和并行编程研究方兴未艾 [3,6~8], 极大地推动了高性能计算的应用发展.

近十多年来, 在武器物理、激光聚变、电磁环境、核能开发等应用领域, 我国并行算法与并行编程经历了从个性到共性, 再到软件复用的发展过程. 这里, 个性指基于通用并行编程模型, 围绕单个串

1) Supercomputer TOP500 List. http//www.top500.org/lists/2015/11. 2016.

引用格式: 莫则尧, 张爱清, 刘青凯, 等. 并行算法与并行编程: 从个性、共性到软件复用. 中国科学: 信息科学, 2016, 46: 1392–1410, doi: 10.1360/N112016-00144

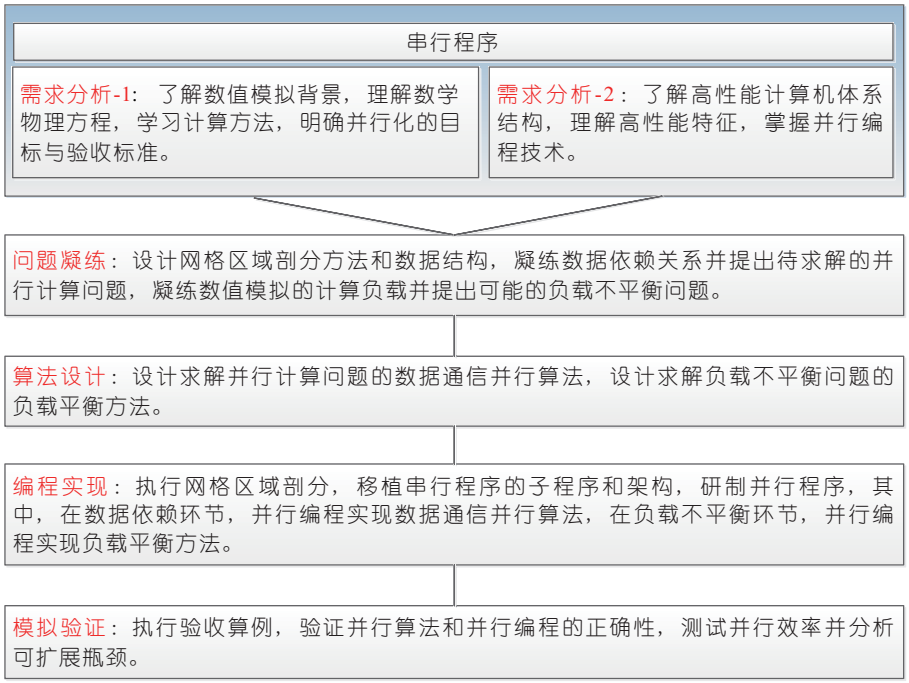

图 1 (网络版彩图) 串行程序并行化的主要步骤

Figure 1 (Color online) The main steps to parallelize a serial program

行程序的并行化, 解决其中的并行算法与并行编程问题; 共性指凝练实际应用领域对高性能计算的共性需求并进行需求建模, 采用模型驱动的研究方法, 解决并行算法与并行编程问题; 软件复用指集成并行算法与并行编程技术研制编程框架 [9,10]、基于编程框架研制超级并行应用软件. 通过从个性到共性, 再到软件复用的技术发展, 我们为相关应用领域的超级并行应用软件高效使用并同步研发于国产超级计算机提出了一条新的可行技术途径, 缓减了高性能计算应用面临的性能墙和编程墙两大瓶颈.

本文内容安排如下. 第 2 节介绍串行程序的并行化研究工作, 第 3 节介绍需求建模和模型驱动的研究工作, 第 4 节介绍集成并行算法与并行编程技术于编程框架, 基于编程框架研发超级并行应用软件的关键技术. 最后, 总结全文并提出发展建议.

2 串行程序并行化研究

随着国产万亿次和十万亿次计算机投入使用, 在武器物理、激光聚变等应用领域, 串行程序亟待并行化. 图 1 给出了串行程序并行化的研究流程, 可分为 “需求分析 – 问题凝练 – 算法设计 – 编程实现 – 模拟验证” 的 5 个步骤.

为了方便领域用户掌握和发展并行程序, 并行算法与并行编程的研究通常还需要如下约束条件:

并行算法不改变物理建模和计算方法. 此时, 并行算法的设计主要涉及网格区域剖分、空间离散在不同处理器核之间的数据通信、隐式时间离散的迭代求解方法在不同处理器核之间的数据通信、动态负载平衡、并行输入输出等多个方面.

并行编程不改变数据结构. 共享存储的数据结构, 通常直接复制给并行程序并在各个进程中保持一致, 而分布存储的数据结构, 例如数组, 通常依据网格的区域剖分进行数据分解后复制给并行程序.

并行程序的流程架构与串行程序基本一致. 通常, 并行程序逼近串行程序越多, 领域用户越容易掌握.

并行编程不改变关键物理量的计算结果. 通常, 随着数值模拟时间步数的增加, 关键物理量的并行计算结果随舍入误差的累积会发生较大的变化. 此时, 需要特殊处理, 确保它们和串行计算结果在一定长度的有效数字范围内一致.

围绕武器物理、激光聚变等领域的串行程序在万亿次和十万亿次计算机上的并行化, 我们提出了可扩展至数百上千个处理器核的系列并行算法和并行编程技术, 其中, 并行算法主要包括: 结构网格和非结构网格的区域剖分算法、"流体 – 扩散 – 输运" 耦合的并行连接算法 [11]、粒子输运通量扫描并行算法 [12]、动态 Monte-Carlo 模拟的并行 I/O 算法 [13]、多层均权动态负载平衡算法 [14]、辐射流体力学模拟的相关并行算法 [15]、并行代数多重网格算法 [16] 等等, 并行编程技术主要包括: 消息传递 MPI 并行编程技术 [17]、共享存储 OpenMP 并行编程技术 [18]、多物理场耦合的多层嵌套并行编程技术、消息传递性能优化技术、共享存储性能优化技术、浮点运算指令级并行优化技术等.

3 并行算法与并行编程: 从个性到共性

随着国产超级计算机由十万亿次提升到千万亿次和亿亿次量级, 体系结构由 MPP 向异构融合发展, 呈现多层嵌套通用并行和异构众核加速并行的高性能特征. 图 1 的技术途径面临 3 个瓶颈 [19]:

数据结构不适应六层嵌套的存储结构. 网格和物理量以数组的单一格式进行存储. 然而, 超级计算机呈现 "结点间分布存储 (DM) – 结点内 CPU (中央处理器) 间非均匀存储 (NUMA) – CPU 内多核间共享存储 (SMP) – CPU 内多核间共享高速缓存 (Cache) – CPU 核内私有高速缓存 (Cache) – CPU 核内浮点运算寄存器 (Register)" 的六层嵌套存储结构, 数组的单一格式很难适应.

并行算法不适应四层嵌套的并行度挖掘. 四层嵌套并行包括: 数千结点之间并行、结点内多个 CPU 之间并行、CPU 内十多个 CPU 核之间并行、CPU 核内多功能部件指令级并行. 千万亿次和亿亿次计算依赖于上述四层嵌套的并行度挖掘. 然而, 受限于数据结构, 即使并行算法可以挖掘四层嵌套的并行度, 并行编程难度也极大.

并行编程不适应复杂应用的模块拓展. 并行程序继承了串行程序的模块架构, 程序模块之间关联度大. 某个程序模块的局部修改, 通常引起其他程序模块的适应性修改. 但是, 新型并行算法、计算方法和物理建模可能需要新的数据结构, 甚至需要为程序模块设计新的连接接口. 这样的研发工作极其繁琐, 需要重写大部分程序代码, 工作量很大, 很难在短期内完成.

针对发展瓶颈, 有两条技术途径可以解决. 第一, 设计新型数据结构, 逐个重构并行程序, 使之适应当前和未来发展; 第二, 凝练数值模拟应用领域的共性需求并进行需求建模, 设计模型驱动的并行算法和并行编程技术, 支持批量的并行程序适应超级计算机和复杂实际应用的发展. 显然, 第二条技术途径具有普适性、创新性和先进性. 近十多年来, 我们对此进行了成功的探索, 以下分节介绍.

3.1 面向共性需求的并行算法与并行编程研发流程

图 2 给出了面向共性需求的并行算法与并行编程的研发流程, 包含 "需求分析 – 需求建模 – 算法设计 – 编程实现 – 运行优化 – 模拟验证" 的 6 个阶段. 对比图 1, 图 2 在 4 个方面基本不同:

数据结构. 图 2 要求数据结构进行抽象建模, 既适应计算机体系结构又支持复杂实际应用; 图 1 直接从串行程序复制, 仅考虑了实际应用的需求.

图 2 (网络版彩图) 并行算法与并行编程: 面向共性的研发流程

Figure 2 (Color online) Parallel algorithm and parallel programming: flowchart for generalization

数据依赖. 图 2 要求数据依赖关系进行抽象建模, 适应众多应用领域数值模拟的物理建模和计算方法; 图 1 聚焦于单一程序的个性化数据依赖关系.

算法和编程. 图 2 立足于模型驱动, 普适性强, 图 1 聚焦于单个串行程序和应用对象, 较少考虑普适性和推广应用.

模拟验证. 图 2 立足于模型驱动的单元分解与定量评估, 可以指导并行算法与并行编程的持续创新; 图 1 聚焦于单一程序, 受限于实际应用的个性化特征, 较难进行单元分解与定量评估.

图 2 的研究内容可用图 3 表示, 其中, 网格数据模型是基础, 它反映的是数值模拟访存特征, 并行计算模式是物理建模和计算方法中数据依赖关系的抽象建模, 计算负载模型是数值模拟实际应用中计算负载特征的抽象建模, 运行时状态的特征建模反映超级计算机在访存、通信、浮点运算、输入输出等方面的实际运行状态, 可用于指导并行算法和并行编程的运行时性能优化.

3.2 并行算法与并行编程的需求建模进展

图 2 中, 需求建模力求高效和普适, 其中, 高效指建模适应于当前和未来一段时期内的超级计算机体系结构, 普适指建模适应于众多的数值模拟应用. 对照图 3, 我们在网格数据模型、并行计算模式和计算负载模型方面取得了进展.

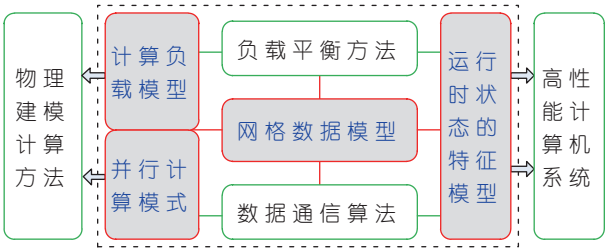

图 3 (网络版彩图) 面向共性需求的并行算法与并行编程研究

Figure 3 (Color online) The common requirement oriented parallel algorithm and parallel programming

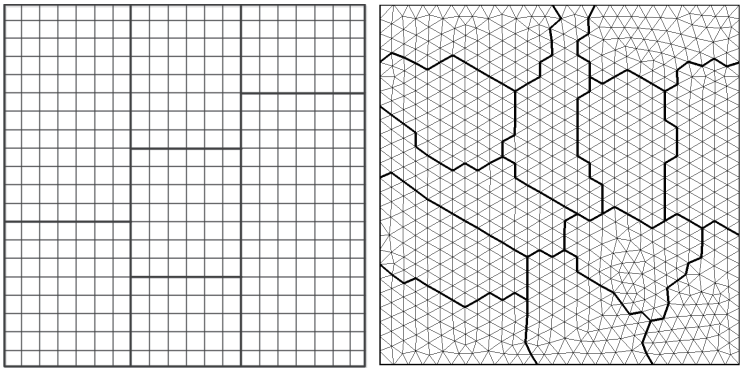

图 4 网格片示意图. (a) 结构网格, 含 7 个网格片; (b) 非结构网格, 含 12 个网格片

Figure 4 Mesh with patches. (a) A structured mesh with 7 patches, and (b) an unstructured mesh with 12 patches

3.2.1 网格数据模型

莫则尧等提出了面向结构网格和非结构网格的网格数据模型 [20~22], 并进行了标准化 [23~25]. 它们以网格片 (Patch) 为核心, 呈现六层嵌套的特征, 即 "网格片层次结构 (PatchHierarchy) – 网格层 (PatchLevel) – 网格区 (PatchDomain) – 网格域 (PatchRegion) – 网格片 (Patch) – 网格单元 (PatchCell)", 还包含在网格片上存储物理量的数据片 (PatchData). 图 4 示例了 7 个网格片构成的一个结构网格和 12 个网格片构成的一个非结构网格.

六层嵌套的网格数据模型具有高效性. 首先, 它们匹配于超级计算机的通用计算体系结构, 既适应六层嵌套的存储结构, 又支持四层嵌套并行度的挖掘. 图 5 给出了匹配关系的示图, 其中, 网格层被剖分为网格区, 网格区捆绑于结点内存, 可实施进程级的消息传递 MPI 并行 [17]; 结点内, 网格区被剖分为网格域, 网格域捆绑于 CPU 内存, 可实施 NUMA 感知的 MPI 或共享存储 OpenMP [18] 并行; CPU 内, 网格域被剖分为网格片, 网格片调度至 CPU 核, 适应共享缓存 Cache, 可实施 OpenMP 并行; CPU 核内, 网格单元调入私有 Cache 内, 可实施线程级 OpenMP 并行; 线程执行过程中, 浮点操作数在寄存器之间, 可实施指令级并行和向量化加速.

其次, 网格数据模型适应异构众核的加速计算体系结构. 在异构计算中, 结点通常配置多个异构加速器, 例如 GPU 或 Intel Xeon Phi (MIC), 它们可等同于 CPU 参与网格域的剖分与分配, 只是, 网格域的大小需要根据异构加速器的计算能力进行裁剪. 在网格域的内部, 可实施网格单元间的线程并行和向量化加速.

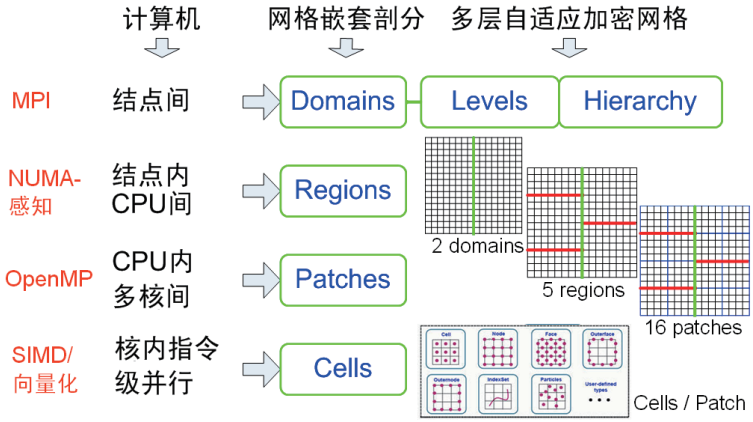

图 5 (网络版彩图) 与超级计算机体系结构匹配的六层嵌套网格数据模型

Figure 5 (Color online) Six-layer grid data model matched with the supercomputer architecture

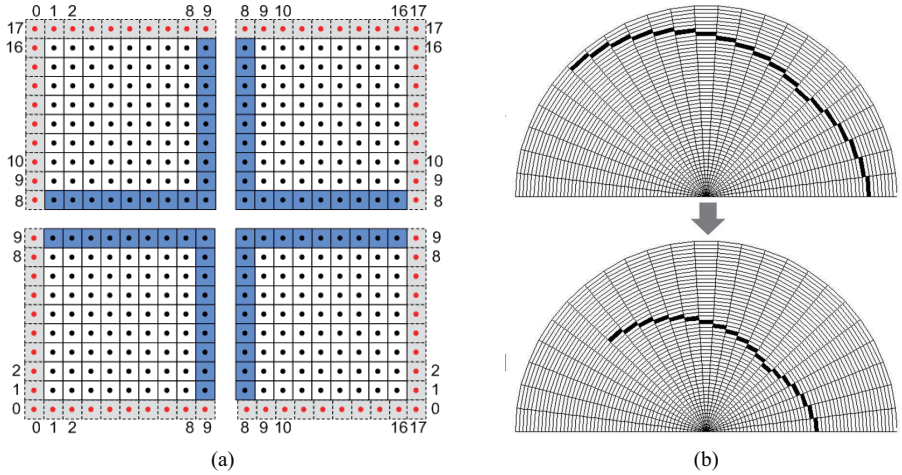

图 6 (网络版彩图) 两种常用的并行计算模式. (a) HALO 模式, 阴影标识待填充的影像区; (b) SWEP 模式, 阴影标识当前可计算的网格单元

Figure 6 (Color online) Two general parallel computing patterns. (a) The HALO pattern in which the shadow shows the cells to be filled; (b) the SWEP pattern in which the shadow shows the active cells currently

六层嵌套的网格数据模型具有普适性, 它们可以支持图 2 列出的物理建模和计算方法的研究, 以及相应领域的超级并行应用软件的研制 [19], 这里不再讨论.

3.2.2 并行计算模式

图 6 示意了两种常用的并行计算模式, 即 Halo-Exchange 模式 (HALO) 和 Sweeping 模式 (SWEP). 对于前者, 在每个网格片上, 存储某些物理量的数据片影像区 (阴影部分标识) 由相邻网格片的数据片填充, 之后各自独立开展数值计算. 对于后者, 扫描从右上角到左下角进行, 相邻网格单元构成上下游的数据依赖关系, 下游网格单元只有等到上游网格单元计算完毕, 才能开始计算, 其中, 阴影标识的是当前可并行计算的网格单元. 也就是说, 网格片内部的网格单元的数值计算是数据驱动的, 莫则尧等为这种扫描数据驱动的并行计算模式提供了基于有向图建模的并行算法设计框架 [26, 27].

表 1　并行计算模式列表

Table 1　Parallel computing models

序号	模式名称	缩写	内涵
1	Halo-Exchange	HALO	网格层的相邻网格片填充数据片的影像区并计算
2	网格数据重分布	REDS	不同剖分的两个网格层完成数据片的重分布映射
3	网格数据归约	REDT	遍历网格层的网格片, 逐个单元地归约数据片
4	联邦数据传输	FEDR	在两个网格层之间, 传输计算所需的数据片值
5	克隆数据归约	CLON	遍历所有克隆网格层, 为网格单元归约数据
6	克隆脉动计算	SYST	所有克隆网格层按一维环完成数据传输和计算
7	Sweeping 计算	SWEP	按扫描方向遍历所有网格片及网格单元并计算
8	稀疏数据归约	SPCL	遍历网格层的部分网格片, 逐个单元地归约数据片
9	稀疏数据分布	SPDS	两个网格层间部分网格片重分布
10	置换数据传输	PERM	基于索引下标置换的网格层数据重分布映射

基于图 5 的网格数据模型, 表 1 列出了 10 种常用的并行计算模式, 其中, 第 2 列是模式名称, 第 3 列是模式缩写, 第 4 列是模式内涵. 表 1 中, 克隆网格层指某个网格层的多个备份网格层, 它们具有完全一致的网格数据模型及其实例化网格片.

3.2.3　计算负载模型

基于六层嵌套的网格数据模型, 计算负载以网格片为单位进行度量. 在数值计算的过程中, 网格片的计算负载可用一个多元向量进行描述, 向量中的元素可以分为静态和动态两类, 其中, 静态通常包含网格单元数、粒子数、浮点运算次数等可以在计算之前统计的量, 动态通常包含计算时间、内存大小、I/O 大小等需要在计算的过程中统计的量. 例如, 在粒子模拟类应用中, 如果粒子的计算占据主导地位, 则粒子个数和内存大小通常可用于标定网格片的计算负载; 在流体力学或辐射流体力学计算中, 网格单元个数通常被用于标定网格片的计算负载.

以网格片为核心, 面向 “结点间 – 结点内 CPU 间 – CPU 内核间” 和 “CPU – 异构加速器” 的并行度的挖掘, 可以建立三层嵌套的计算负载模型. 对应图 5, 顶层是结点间计算负载模型, 次层是结点内 CPU 之间、CPU 与异构加速器之间的计算负载模型, 底层是 CPU 内部多核之间的计算负载模型. 由此, 负载平衡可以在结点间、结点内 CPU 间/CPU 与异构加速器间、CPU 内多核间/异构加速器内的 3 个层次进行. 其中, 底层是局部的, 开销小, 高层是全局的, 开销大, 负载平衡应该尽可能在底层进行.

3.3　模型驱动的并行算法与并行编程研究

基于六层嵌套的网格数据模型, 针对并行计算模式和计算负载模型, 我们提出了系列的并行算法和并行编程技术, 它们在 “天河二号” 亿亿次计算机上 [28], 可以扩展到十万 CPU 核和百万异构加速众核. 典型代表如下所述.

3.3.1　面向 HALO 模式的并行算法与并行编程技术

面向 HALO 模式, 提出了三阶段四层嵌套的影像区数据填充并行算法, 并进行了相应的并行编程实现. 第 1 阶段是数据填充的通信声明阶段, 用于声明待填充的物理量及其数据片; 第 2 阶段是数据

填充的通信调度阶段, 用于创建数据片影像区填充的通信事件, 描述两个相邻网格片的数据片影像区的数据复制、传输、填充的操作方法; 第 3 阶段是数据填充的通信执行阶段, 完成通信调度创建的所有通信事件.

第 2 阶段的通信调度创建开销是影响性能的关键因素之一. 为了降低开销, 我们将网格片之间的邻居关系抽象为无向图模型, 提出了统一框架的通信调度创建算法. 特别地, 对多块拼接结构网格, 我们提出 "旋转 – 平移" 的描述方法 [29], 使之也适应于统一框架. 在此基础上, 我们还提出了区间树搜索 Box 交集算法和区域分解 Box 差集算法, 将无向图模型及通信事件的创建开销降低到最优计算复杂度 O(NlogN/C) [30, 31], 其中, N 为网格片数, C 为线程数. 测试表明, 对三维的 N=104 万、C=4 的情形, 在 "天河二号" 的 4 个 CPU 核上, 通信调度的创建时间可以控制在 1.6 秒内.

第 3 阶段匹配于 "网格区 – 网格域 – 网格片 – 网格单元", 分 4 个层次嵌套执行. 第 1 层次是网格区之间, 由结点间 MPI 编程实现, 其中, 消息的打包、解包和长度配置可在运行时进行优化. 第 2 层次是网格域之间, 由结点内 CPU 间 MPI 或 OpenMP 编程来实现 NUMA 访存, 其中, 网格域与 CPU 内存的捆绑对降低访存开销十分必要 [32]. 第 3 层次是网格片, 由 OpenMP 编程来实现 CPU 内多核间 SMP 访存, 其中, 多个线程访存竞争是影响性能的关键因素. 第 4 层是 CPU 核内多级 Cache 之间, 通常以 Cache 线为基准, 通过网格单元的排序优化来提升计算效率. 另外, 对于异构众核加速计算, 第 2 层次需要考虑 CPU 和异构加速器之间的数据传输, 第 3 层次需要考虑加速器内众核的访存竞争, 第 4 层次需要考虑网格单元间的指令级并行.

3.3.2 面向 SWEP 模式的并行算法与并行编程技术

面向 SWEP 模式, 提出了基于双层有向图模型的数据驱动并行算法 [33], 并进行了相应的并行编程实现. 外层有向图模型以网格片为结点, 以网格片之间的有向数据依赖关系为有向边; 内层有向图模型局部于每个网格片, 以网格片内网格单元为结点、以网格单元间有向数据依赖关系为有向边, 以外层有向图模型的相关有向边为边界输入或输出. 围绕单层有向图模型, 我们提出了数据驱动统一框架的并行算法以及结点优先级算法 [26, 27], 进行了 MPI 编程实现 [34], 在数千上万个 CPU 核上获得了良好性能; 围绕双层有向图模型, 我们改进了数据驱动统一框架的并行算法, 实现了 MPI 和 OpenMP 的混合编程, 在十万 CPU 核上获得了可扩展的并行性能.

3.3.3 面向多物理场耦合的并行算法与并行编程技术

FEDR 模式是支持多物理场耦合的并行计算模式 [35], 负责在多个网格层之间填充耦合所需的数据片并实施插值, 其中, 数据传输可以复用 HALO 模式的三阶段四层嵌套并行算法, 数值插值需要调用高精度、守恒、单调型插值算法. 数据依赖关系不同于 HALO 模式, 通常由不同网格层的重叠区以及插值算子宽度确定. 因此, FEDR 模式的数据传输开销通常远大于 HALO 模式, 主要原因是重叠区可能属于不同的网格片, 甚至可能分配到不同的结点或结点内不同 CPU 中. 为了降低数据传输的开销, 通常需要优化重叠区的网格剖分以及网格区到结点、网格域到 CPU、网格片到 CPU 核上的映射. 例如, 针对辐射流体力学耦合中子输运的数值模拟的多物理场耦合, 我们提出了极小化数据移动的两层嵌套数据传输并行算法 [11], 将开销控制在合理的范围之内.

3.3.4 面向 CLON 模式的并行算法与并行编程技术

CLON 模式是支持相空间并行的计算模式, 负责对克隆网格层的数据实施并行计算, 负责在克隆网格层之间广播并行计算所需的数据片数据, 收集并行计算之后的数据片数据.

相空间并行指空间维度之外其他维度的并行计算, 主要包括能群和粒子并行计算. CLON 模式描述相空间并行计算中的数据依赖关系, 能群或粒子之间的并行是完全独立的, 并行可扩展能力由两个因素决定, 其一是能群与粒子数据分解的负载均衡, 其二是广播和收集的数据量. 对于前者, 能群或粒子的循环分配通常可以解决负载不平衡的问题; 对于后者, 基于流水线的阵列脉动算法可以降低数据传输的开销. 当前, 在天河二号计算机系统上, CLON 模式可以实现能群和粒子群在数十上百个克隆网格层上的并行计算.

3.3.5 计算负载模型驱动的负载平衡方法

针对三层嵌套的计算负载模型, 负载平衡方法负责将网格层剖分成网格片 (域、区), 再将网格片 (域、区) 映射到 CPU 核 (CPU、结点). 当前, 负载平衡方法主要由 4 个步骤组成.

负载平衡状态判据. 衡量当前计算负载的不平衡程度, 其中, 负载平衡效率是最常用的判据;

网格片 (域、区) 剖分. 将网格层剖分成网格片 (域、区), 其中, 三层嵌套 (区 – 域 – 片) 的网格区域剖分方法是有效的剖分策略;

网格片 (域、区) 映射. 将网格片 (域、区) 映射到 CPU 核 (CPU、结点);

网格片迁移. 将网格片从旧的负载不均衡网格层重分布到新创建的负载均衡网格层上, 这里, 网格片的迁移需要调用表 1 列出的 REDS 模式.

负载平衡的开销主要源自步骤 2 和步骤 4. 目前, 三层嵌套网格剖分方法可以将步骤 2 的开销降低到最优计算复杂度 O($N\log N/C$), 其中, N 为网格片数, C 为线程数; 但是, 步骤 4 的开销依赖于新旧两个网格层的近似程度, 如果一致或近似一致, 则开销较小, 否则密集的结点间数据传输将导致很大的开销.

在数值模拟应用中, 负载的不平衡现象随着数值模拟的时间步进而逐步呈现. 通常, 负载不平衡首发于局部区域, 然后逐渐扩散到整个区域. 如果网格的剖分和网格片的迁移也采用扩散的方法来进行, 则可将负载的不平衡始终局限于局部区域, 从而负载平衡的开销可以得到有效的控制. 当前, 基于自动扩散的负载平衡方法是大规模并行计算应用的重要方法 [31, 36], 应用该类方法, 我们解决了激光等离子体相互作用模拟中粒子分布不均匀、气候模拟中化学物理过程计算量不均造成的负载不平衡问题, 将数万 CPU 核的大规模并行计算的负载平衡效率从 20% 提升到 80% 以上.

3.4 并行算法与并行编程的运行时性能优化

随着超级计算机体系结构的日趋复杂和可靠性的日趋下降, 系统运行状态的不稳定逐步成为制约性能提升的重要因素. 由此, 运行时性能优化成为并行算法和并行编程研究的一个重要环节, 我们从访存、通信、容错的 3 个方面建立了运行时性能优化方法, 开展了相应的研究工作.

在访存方面, 基于内存绑定技术, 提出了匹配 CPU 内多核、结点内 NUMA 结构、结点内异构众核加速器等特征的高效内存管理机制, 建立了相应的动态内存管理模块. 基于该模块, 可以在超级计算机上实例化前述网格数据模型并完成它们与体系结构的匹配.

在通信方面, 基于计算机系统的网络拓扑结构和网络的数据流状态, 提供结点间数据通信的性能优化支持. 在它们的支持下, 并行算法的设计与分析只需考虑通信事件的创建, 不必关心通信事件的

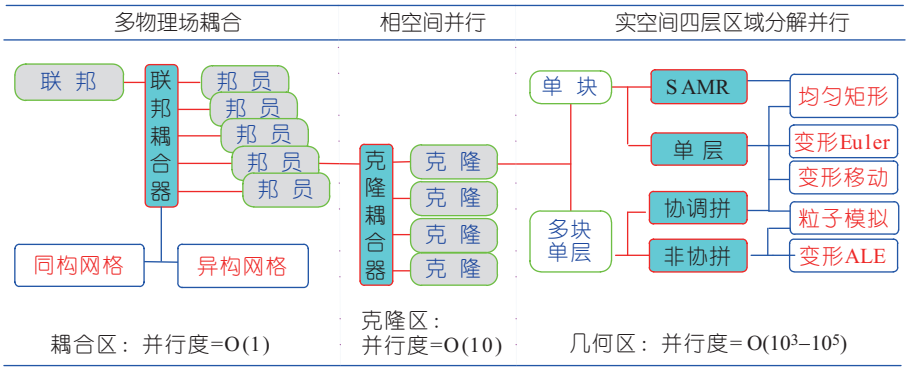

图 7　(网络版彩图) 支持复杂实际应用数值模拟的并行可扩展能力谱图

Figure 7　(Color online) Parallel scalability of numerical simulation for complex and real applications

具体执行. 随着结点数的增加和结点内 CPU 核数的增加, 以及结点间互联网络拓扑结构的优化发展, 我们对运行时通信性能优化的需求越来越大, 当前仅支持常见的批量点对点通信.

在容错方面, 基于重启动和并行 I/O 功能, 我们建立了应用级检查点容错机制, 支持超级并行应用软件的自动容错. 在该机制中, 应用软件只需注册待保存和恢复的网格数据, 容错模块就能选择合适的时机将数据备份到磁盘或伙伴结点的内存, 并在结点出错时从这些备份恢复数据并重新启动计算. 一般情况下, 网格数据模型的实例化数据结构已经包含了应用软件需要备份的数据, 领域用户只需注册应用个性的私有数据.

3.5　并行算法与并行编程的可扩展性

面向图 2 顶层列出的应用领域, 针对相关物理建模和计算方法对高性能计算的需求建模, 结合并行算法与并行编程的上述研究进展, 我们给出了图 7 所示的并行可扩展能力谱图.

分布图分耦合区、克隆区和几何区的 3 个区域. 左端为耦合区, 表示多物理场耦合的网格层间并行计算, 由 FEDR 模式支持, MPI 编程实现, 可挖掘的并行度在 O(1) 量级. 中间为克隆区, 表示网格层上空间维度之外其他维度的并行计算, 由 CLON 模式和 SYST 模式支持, MPI 编程实现, 可挖掘的并行度在 O(10) 量级. 右端为几何区, 表示实空间的 “网格区 – 网格域 – 网格片 – 网格单元” 的四层嵌套区域分解并行计算, 由 HALO 耦合 REDS, REDT 等模式支持, MPI 与 OpenMP 编程实现, 可挖掘的并行度在 $O(10^3 \sim 10^5)$ 量级.

针对 HALO 模式, 以网格片为单位, 并行扩展能力随离散网格和计算方法的不同而不同, 具体分如下的 3 个量级.

$O(10^5)$ 量级: 对单块或多块协调拼接的单层均匀矩形网格、变形 Euler 网格、变形移动网格和粒子模拟, MPI 耦合两层 OpenMP 编程实现.

$O(10^4)$ 量级: 对单块并行网格自适应计算 (SAMR), MPI 耦合两层 OpenMP 编程实现, 受限于网格自适应导致的动态负载不平衡.

$O(10^3)$ 量级: 对单层多块非协调拼接的变形 ALE 网格, MPI 耦合两层 OpenMP 编程实现, 受限于多块非协调拼接的负载不平衡和数据传输开销.

在几何区, 其他并行计算模式的并行扩展能力低于 HALO 模式. 例如, REDT 模式在 $O(10^3)$ 量级, 与 HALO 模式相当, 但较难突破 $O(10^4)$ 的量级; REDS 模式只能达到 $O(10^3)$ 量级. 不过, 面向复

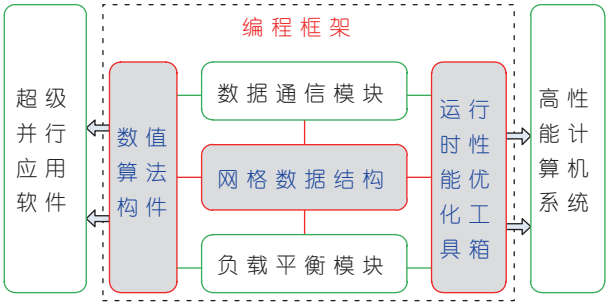

图 8 (网络版彩图) 并行算法与并行编程的软件复用架构

Figure 8 (Color online) Software reuse architecture for parallel algorithm and parallel programming

杂实际应用, HALO 模式和 REDT 模式是最常用的两个模式, 其他模式的影响相对较小.

由此, 给定一个超级并行应用软件, 针对物理建模和计算方法, 可以在图 7 中从左至右找到唯一的路径, 求出路径所经过区域的并行度的乘积, 为其预估并行扩展能力. 图 7 表明, 超级并行应用软件的并行可扩展能力处于 $O(10^3 \sim 10^6)$ 量级. 如果以单处理器核的百亿次计算为单位, 则当前的并行算法和并行编程研究可以在不同的应用情形下, 支持超级并行应用软件分别扩展到十万亿次、百亿亿次、千万亿次或亿亿次计算. 例如, 对单层多块非协调拼接的变形 ALE 网格应用情形, 只能扩展到十万亿次计算; 对多物理场耦合、能群克隆、实空间 Euler 网格或粒子模拟、显式时间积分的复杂实际应用, 最强可扩展到亿亿次计算.

4 并行算法与并行编程: 从共性到软件复用

并行算法设计和并行编程实现是超级并行应用软件研制的重要环节. 根据前面的论述, 它们可以凝练为共性的模型, 可以研制为共享的软件模块, 复用于众多的超级并行应用软件. 由此, 一个公开的问题是, 采用什么样的软件技术来实现这个目标? 本节立足面向构件化的软件设计和实现技术 [37] 以及模型/模式驱动的软件架构技术 [38], 阐述我们已经开展的相关研究工作.

总体架构如图 8 所示. 首先, 构件化网格数据模型, 支持领域用户为并行应用软件创建匹配于超级计算机体系结构的网格数据结构; 其次, 基于网格数据模型, 凝练并行应用软件数值算法的计算特征为多种不同类型的数值算法构件; 再次, 层次化和构件化并行计算模式和计算负载模型, 封装模型驱动的并行算法和并行编程技术以及运行时性能优化工具箱, 支持数值算法构件在超级计算机上的实现; 最后, 研制数值模拟并行应用编程框架, 以数值算法构件的接口为编程接口, 实现并行算法和并行编程技术的软件复用, 支持超级并行应用软件的研制和应用.

4.1 网格数据模型的构件化

图 5 所示的网格数据模型可以分解为 3 类, 即变量模型、数据片模型和网格片模型. 变量模型规范物理量的声明, 实例化为变量; 数据片模型规范物理量在网格上的存储, 实例化为数据片; 网格片模型规范网格的管理, 实例化为网格片. 在此基础上, 离散计算区域的网格可以由网格层实施管理, 网格层可以分解为网格区、网格域, 直至网格片. 在网格片上, 可以根据变量索引所有数据片. 在自适应结构网格计算中, 网格片层次结构可以由多个局部嵌套加密的网格层构成; 在多物理耦合或接触碰撞的结构网格计算中, 多块结构网格可以由多个拼接的网格层构成. 由此, 在结构网格和非结构网格之上,

表 2 典型的内核构件

Table 2 Typical kernel component models

序号	构件名称	并行计算模式	内涵
1	数值	HALO	完成 HALO 模式的一次并行计算
2	归约	REDT	完成 REDT 模式的一次并行计算
3	联邦	FEDR	完成 FEDR 模式的一次并行计算
4	克隆	CLON	完成 CLON 模式的一次并行计算
5	脉动	SYST	完成 SYST 模式的一次并行计算
6	扫描	SWEP	完成 SWEP 模式的一次并行计算
7	接触	REDS	完成一次非协调拼接的并行计算
8	内存	独立并行	为网格层的所有数据片完成一次并行内存调度
9	赋值	独立并行	为网格层的所有数据片完成一次并行赋值操作

网格数据模型可以构件化, 实现并行应用软件数据结构的标准化和规范化, 支持并行应用软件的数据存储和访问适应计算机体系结构的高性能特征.

关于变量模型、数据片模型和网格片模型的详细介绍及其实例化操作, 请参考 JASMIN 框架用户手册 [21], 这里不再论述.

4.2 数值算法构件

基于网格数据模型, 数值算法构件实现数值算法的抽象和复用. 它们分为 3 类, 第 1 类是时间积分算法构件, 第 2 类是内核构件, 第 3 类是数值代数构件, 分别介绍如下.

时间积分算法构件在网格片层次结构上为数据片赋初值、为时间步的时间积分求步长、积分一个时间步、更新时间积分后的数据片结果. 时间积分算法构件是对应用软件主程序的数值算法的抽象建模, 其实例化需要输入:

初值构件. 在时间积分的初始时刻, 在网格层上创建相关内核构件, 实例化和组装内核构件为流程, 为网格层所有数据片赋初值;

步长构件. 在网格层上创建相关内核构件, 实例化和组装内核构件为流程, 实现求解时间积分步长的计算方法, 为每个时间步计算时间步长;

积分构件. 在网格层上创建相关内核构件, 实例化和组装内核构件为时间积分流程, 实现物理建模和计算方法, 完成一个时间步的时间积分;

终值构件. 积分一个时间步后, 为网格层上相关数据片更新数值计算结果并调度内存.

内核构件是对时间积分构件实例化所需的数值算法的抽象建模. 表 2 列出了典型的内核构件以及它们依赖的并行计算模式, 包括数值、归约、联邦、克隆、脉动、扫描、接触、内存、赋值等构件. 表 2 表明, 内核构件唯一地依赖于某个并行计算模式. 此外, 数值、归约、扫描等内核构件的实例化需要网格片数值计算子程序, 它们可以集成到网格片数值计算类 [21], 这里不详细论述.

数值代数构件在网格层上, 以网格层为数据结构, 对数值代数算法进行抽象而建立的数值算法构件, 包括矩阵向量运算、稀疏 (非) 线性代数方程组解法器、矩阵特征值解法器、快速离散变换、快速多极子算法等. 通常地, 该类构件仅提供连接功能, 调用第三方解法器库来实现数值代数问题的求解. 结合网格数据模型, 也可以创建和组装内核计算构件, 通过这些构件的组装来研制更高效的专用解法器. 这里, 不再论述.

4.3 并行算法与并行编程的软件架构及编程框架研制

为了集成并行算法和并行编程研究成果, 我们提出并采用了图 9 所示的软件架构. 该架构分为 6 层, 桥接并行应用软件和高性能计算机体系结构. 该架构与莫则尧等提出的数值模拟领域并行编程模型的实现架构类似 [25].

第 1 层为运行时支撑层, 它集成运行时性能优化工具箱, 从访存、通信、容错、输入输出的 4 个方面, 跨平台地支持并行算法与并行编程实现于超级计算机.

第 2 层为网格数据模型层, 它集成网格数据模型, 提供网格数据模型的实例化支持, 包含多种不同类型的结构网格与非结构网格、物理量与数据片, 为网格层中网格片的相邻关系生成无向图和有向图. 网格数据模型的实例化需要运行时支撑层在微处理器资源调度和访存方面的支撑.

第 3 层为并行计算模式层, 它集成实现并行计算模式的并行算法和并行编程技术. 并行计算模式层建立在网格数据模型之上, 并且需要运行时支撑层在通信、输入输出、运算、稳定性等方面的支撑.

第 4 层为计算负载模型层, 它支持以网格片为核心的计算负载建模以及模型驱动的负载平衡判据、负载重新分配、负载动态迁移的动态负载平衡, 支持 “网格层 – 网格区 – 网格域 – 网格片” 的四层嵌套网格剖分. 该层需要并行计算模式层和网格数据模型层的支撑.

第 5 层为快速数值并行算法层, 它继承数值代数、离散变换等问题的快速数值并行算法, 支持数值代数构件的实现. 该层需要网格数据模型层、并行计算模式层和计算负载模型层的支撑.

第 6 层为数值算法构件层, 它支持领域用户选择和实例化数值算法构件, 组装数值算法构件来研制超级并行应用软件. 该层是对并行算法与并行编程技术的封装.

基于 6 层软件架构, 我们研制了结构网格的编程框架 JASMIN [19, 21] 和非结构网格的编程框架 JAUMIN [22]. 两个编程框架支持并行算法和并行编程的软件复用, 可以从 4 个层面来加深理解.

第 1 个层面, 编程框架以网格数据模型为基础, 提供数值算法构件, 支持领域用户串行编程地研制时间积分算法构件, 以及实例化时间积分算法构件所需的内核构件和数值代数构件, 从而可以串行编程地研制超级并行应用软件, 具体如图 10 所示. 图 10 中间表示编程框架提供的部分内核构件、初值构件和步长构件; 左端表示领域用户研制的网格片数值计算子程序, 它们通过策略设计模式 [39] 封装于数值算法构件, 用于实例化构件; 右端表示初值构件和网格层时间积分算法构件的构件组装流程.

第 2 个层面, 编程框架以并行计算模式和负载平衡建模支持内核构件的实例化, 其中, 并行计算模式依赖于数据通信的并行算法和并行编程技术, 负载平衡建模支持处理器核之间的负载平衡. 没有并行算法和并行编程技术的高效能, 就没有内核构件在超级计算机上的高效能实例化.

第 3 个层面, 网格数据模型和并行计算模式的实例化需要超级计算机的运行时性能优化, 为此, 我们凝练形成运行时性能优化工具箱, 在访存、通信、容错的 3 个方面提供支撑. 运行时性能优化屏蔽超级计算机的运行时特征, 提供跨平台的性能优化技术, 支持并行算法和并行编程技术的研究.

第 4 个层面, 数值代数解法器库的研制可以基于编程框架, 也可以独立于编程框架而调用第三方解法器库. 不管怎样, 如图 8 所示, 它们均无缝对接于编程框架, 共同支持超级并行应用软件的研制.

通过以上分析可知, 图 8~10 构成了并行算法与并行编程研究从个性到共性, 再到软件复用的基础研究架构. 特别地, 并行应用编程框架在并行应用软件之间实现了并行算法和并行编程的软件复用, 可以支持领域用户 “并行思考、串行编程” 地研制高效使用超级计算机的超级并行应用软件. 这里, “并行思考” 指领域专家根据物理建模和计算方法的数据依赖关系选择数值算法构件 (图 10 中间所示), “串行编程” 指领域用户编写网格片数值计算子程序 (图 10 左端所示)、实例化时间积分算法构件和构件化组装计算流程 (图 10 右端所示).

图 9 (网络版彩图) 并行算法与并行编程的六层软件架构

Figure 9 (Color online) Six-layer software architecture for parallel algorithm and parallel programming

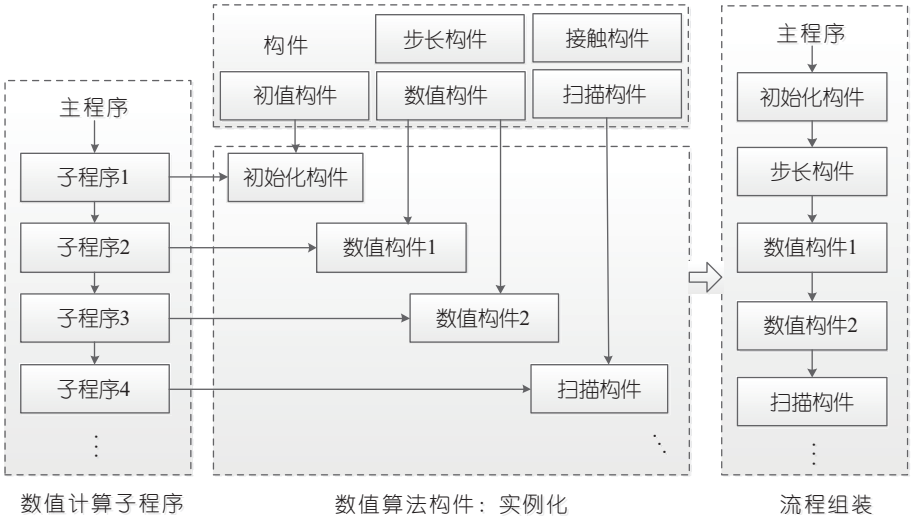

图 10 基于编程框架的并行应用软件研制方法

Figure 10 The development method for parallel software using programming framework

表 3 5 个典型应用软件在天河二号上的并行效率测试结果

Table 3 The parallel efficiency of five typical application softwares on TianHe-2

超级并行	2 CPU/结点		2 CPU+3 MIC/结点	
应用软件	最大核数	并行效率 (%)	最大核数	并行效率 (%)
LAP3D	122880	78	936000	54
JEMS-TD	122880	92	936000	80
LARED-S	98304	76	798720	84
LARED-P	115200	76	936000	78
MOASP	49152	94	399360	43

5 基于编程框架的超级并行应用软件研发验证

目前, JASMIN 框架和 JAUMIN 框架在武器物理、激光聚变、电磁环境、材料科学、地球环境、气候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、结构力学分析等方向, 支持研制了批量的超级并行应用软件, 它们可以有效使用数千至数十万个 CPU 核完成数十个小时、数十亿网格单元、数百亿粒子的数值模拟.

表 3 给出了 5 个超级并行应用软件在 "天河二号" 亿亿次计算机系统上的并行性能测试结果[2)], 它们均基于 JASMIN 框架研制, 其中:

LAP3D 软件. 激光聚变应用软件之一 [40]. 该程序可以模拟激光在等离子体中传播时所激发的成丝、受激 Raman 散射、受激 Brillouin 散射等不稳定性过程. 测试模型选取了三维激光光束的传播和成丝过程, 研究不同相干手段对成丝的抑制作用. 弱可扩展性测试采用单节点 256×256×256 的网格规模, 纯 CPU 计算的最大规模为 859 亿网格单元, CPU+MIC 协同计算的最大规模为 805 亿网格单元.

LARED-S 软件. 激光聚变应用软件之一 [41]. 该程序可以进行欧拉网格下的二维、三维多介质辐射流体力学界面不稳定性模拟, 物理建模包括流体动力学、电子和离子热传导、辐射过程、热核反应、带电粒子输运等物理过程. 测试模型选取了点火靶丸内爆阻滞阶段流体力学不稳定性多尺度模拟. 弱可扩展性测试采用单结点 120×120×120 的网格规模, 纯 CPU 计算和 CPU+MIC 协同计算的最大规模均为 71 亿网格单元.

JEMS-TD 软件. 三维时域全波电磁平台级模拟应用软件 [42]. 该程序使用时域有限差分方法求解 Maxwell 方程组通过精确建模及全波电磁模拟, 获取时域近场和远场电磁信息. 该软件可应用于关键区域高分辨率电磁环境模拟、大型平台的电磁特性分析、电磁兼容性分析等. 测试模型选取了三维时域全波电磁模拟. 弱可扩展性测试采用单结点 600×400×400 的网格规模, 纯 CPU 计算的最大规模为 4915 亿网格单元, CPU+MIC 协同计算的最大规模为 4608 亿网格单元.

LARED-P 软件. 激光聚变应用软件之一 [43]. 该程序采用三维粒子云网格法, 通过追踪大量的在自洽和外加电磁场作用下的带电粒子的运动, 研究等离子体集体性质. 测试模型选取了激光等离子体相互作用粒子模拟研究强激光与锥结构靶相互作用中光束的聚焦特征和高能电子能谱分布. 弱可扩展性测试采用单节点 60×90×60 的网格规模, 每个网格单元 100 粒子, 纯 CPU 计算和 CPU+MIC 协同计算的最大规模均为 16 亿网格单元和 1555 亿粒子.

2) "天河 -II" 超级计算机系统位于国家超算广州中心, 含 16000 个结点, 每个结点含 2 颗 Xeon E5 2692 型号 CPU 和 3 颗基于众核架构的 Xeon Phi (MIC) 的 57 核协处理器, 共计 32000 颗 Xeon E5 CPU 和 48000 个 MIC, 312 万个计算核. 单结点的峰值计算能力为 3.431 TFlops, 16000 个结点总峰值性能达 54.9 PFlops.

MOASP 软件. 分子动力学模拟软件 [44]. 该程序能够模拟气体、流体、金属、高分子等体系, 能够进行绝热系综、正则系综、恒温 — 恒压系综等系综控制, 已经成功应用于反应堆包壳材料的辐照损伤模拟. 测试模型选取了 Lennard-Jones 流体体系. 弱可扩展性测试采用单结点 27×27×27 的网格规模, 887K 粒子规模, 纯 CPU 计算和 CPU+MIC 协同计算的最大规模均为 0.4 亿网格单元和 18 亿粒子.

测试结果表明: 五个软件实现了十万 CPU 核、数十万到百万 MIC 核的异构协同计算, 其中, LAP3D、JEMS-TD 和 LARED-P 在 CPU+MIC 协同计算下可扩展到近 94 万核, 并行效率分别为 54%、80% 和 78%; LARED-S 在 CPU+MIC 协同计算下可扩展到近 80 万核, 并行效率为 84%; MOASP 可扩展到近 40 万核, 并行效率为 43%.

6 结束语

并行算法和并行编程是高性能计算应用研究主要内容, 是超级并行应用软件研发的重要环节. 近十多年来, 面向武器物理、激光聚变、电磁环境和核能开发等重大应用, 并行算法和并行编程经历了串行程序并行化的个性研究、高性能计算需求建模和模型驱动的共性研究、再到并行应用编程框架的软件复用的发展过程. 特别地, 我们构建了图 8∼10 所示的基础研究架构, 在发展并行算法与并行编程技术的同时, 为超级并行应用软件高效使用和国产超级计算机系统的同步研发探索了一条新的可行技术途径, 即 “集成并行算法与并行编程技术研制编程框架、基于编程框架研发超级并行应用软件”, 支持领域用户 “并行思考、串行编程” 地研发超级并行应用软件.

基于网格数据模型, 除了并行计算模式、计算负载模型和运行时性能优化之外, 浮点运算模式是另一类需要关注的研究内容, 它对物理建模和计算方法的数值计算子程序进行循环级和语句级的特征建模, 实施跨微处理器平台的自动性能优化, 是有效提升计算效率并屏蔽性能优化实现的一类重要方法 [33]. 限于篇幅, 这里不再论述.

当前, 并行算法和并行编程技术以及编程框架初步适应于亿亿次量级的数值模拟应用. 面向更高性能的十亿亿次量级或百亿亿次计算, 需要紧密结合计算机体系结构的高性能特征和复杂实际应用的数值模拟应用特征, 持续改进网格数据模型、并行计算模式、浮点运算模式和运行时状态特征模型, 提升并行算法和并行编程技术的总体效能. 在此过程中, 尽可能不改变数值算法构件的实例化和函数调用接口, 以便当前基于 JASMIN 框架和 JAUMIN 框架研制的批量超级并行应用软件可以快速移植到更高性能的超级计算机.

致谢 论文撰写工作得到了 JASMIN 框架研制团队的帮助, 杨章、成杰、刘旭、杨扬等副研究员提供了 “天河二号” 上的并行性能测试数据, 在此表示衷心感谢.

参考文献

1 Yang X J. Sixty years for parallel computing. China 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]

2 Zhang Y Q, Yuan G X, Sun J C, et al. Review and perspectives of 10 years' China HPC TOP100. 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, 34: 11–16]

3 Amarasinghe S, Hall M, Lethin R, et al. DOE workshop on exascale programming challenges. http://science.energy.gov/∼/media/asrc/pdf/program-documents/ProgrammingChallengeWorkshopReport.pdf. 2011

4 Mo Z Y. Research on programming framework for high performance science and engineering computation. Commun CCF, 2014, 10: 8–12 [莫则尧. 面向高性能科学与工程计算的领域编程框架研究. 中国计算机学会通讯, 2014, 10: 8–12]

5 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]

6 Dongarra J, Foster I, ed. Mo Z Y, Chen J, Cao X L, et al. trans. Sourcebook of Parallel Computing. Beijing: Electronic Industry Press, 2005 [Dongarra J, Foster I, 主编. 莫则尧, 陈军, 曹小林, 等译. 并行计算综论. 北京: 电子工业出版社, 2005]

7 Bader D A. Petascale Computing: Algorithms and Applicaitions. New York: Chapman & Hall/CRC, Computational Science Series, 2007

8 Simon H. Exascale challenges for the applied mathematics community. In: Proceedings of DOE Applied Mathematics Program Meeting, Berkeley, 2010

9 Dubey A, Almgren A, Bell J, et al. A survey of high level frameworks in block-structured adaptive mesh refinement packages. J Parallel Distrib Comput, 2014, 74: 3217–3227

10 Mo Z Y. High performance programming frameworks for numerical simulation. Nat Sci Rev, 2016, 3: 1–3

11 Mo Z Y. Concatenation algorithm for parallel numerical simulation of hydrodynamics coupled with neutron transport. Int J Parallel Program, 2005, 33: 57–71

12 Mo Z Y, Fu L X. Parallel flux sweep algorithm for neutron transport on unstructured grid. J Supercomput, 2004, 30: 5–17

13 Mo Z Y, Huang Z F. Application of MPI-IO in parallel particle transport Monte-Carlo simulation. Parallel Algorithm Appl, 2004, 19: 227–236

14 Mo Z Y, Zhang B L. Multilevel averaging weight method for dynamic load imbalance problems. Int J Comput Math, 2001, 76: 463–477

15 Mo Z Y, Zhang A Q, Cao X L, et al. Research on parallel numerical simulation for multi-media radiation hydrodynamics. China J Prog Natural Sci, 2006, 16: 287–292 [莫则尧, 张爱清, 曹小林, 等. 多介质辐射流体力学数值模拟中的并行计算研究. 自然科学进展, 2006, 16: 287–292]

16 Mo Z Y, Xu X W. Relaxed RS0 or CLJP coarsening strategy for parallel AMG. Parallel Comput, 2007, 33: 174–185

17 MPI Forum. MPI: a message passing interface standard. Version 3.1. http://www.mpi-forum.org. 2016

18 OpenMP Forum. OpenMP: application program interface. Version 4.5. http://www.openmp.org. 2015

19 Mo Z Y, Pei W B. Scientific computing application codes. China J Physics, 2009, 38: 552–558 [莫则尧, 裴文兵. 科学计算应用程序探讨. 物理, 2009, 38: 552–558]

20 Mo Z Y, Zhang A Q, Cao X L, et al. JASMIN: a parallel software infrasture for scientific computing. Front Comput Sci China, 2010, 4: 480–488

21 Mo Z Y, Zhang A Q. 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. JASMIN 框架用户指南 (2.0 版). 北京应用物理与计算数学研究所, 技术报告 T09-JMJL-01. 2011]

22 Liu Q K, Zhao W B, Cheng J, et al. A programming framework for large scale numerical simulations on unstructured mesh. In: Proceedings of the 2nd IEEE International Conference on High Performance and Smart Computing (IEEE HPSC), New York, 2016

23 Mo Z Y, Zhang A Q. Programming standards and specifications for high performance numerical simulation on structured mesh V1.0. Institute of Applied Physics and Computational Mathematics, Technical Report IAPCM-SCNS-RB01-2014. 2014 [莫则尧, 张爱清. 面向结构网格应用的高性能数值模拟编程标准与规范 V1.0 版. 北京应用物理与计算数学研究所, 技术报告 IAPCM-SCNS-RB01-2014. 2014]

24 Mo Z Y, Liu Q K. Programming standards and specifications for high performance numerical simulation on unstructured mesh V1.0. Institute of Applied Physics and Computational Mathematics, Technical Report IAPCM-SCNS-RB02-2014. 2014 [莫则尧, 刘青凯. 面向非结构网格应用的高性能数值模拟编程标准与规范 V1.0 版. 北京应用物理与计算数学研究所, 技术报告 IAPCM-SCNS-RB02-2014. 2014]

25 Mo Z Y, Zhang A Q, Liu Q K, et al. Research on the components and practices for domain-specific parallel programming models for numerical simulation. Sci Sin Inform, 2015, 45: 385–397 [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]

26 Mo Z Y, Zhang A Q, Gabriel W. Scalable heuristic algorithms for the parallel execution of data flow acyclic digraphs. SIAM J Sci Comput, 2009, 31: 3626–3642

27 Mo Z Y, Zhang A Q, Yang Z. A new parallel algorithm for vertices priorities of data flow acyclic digraphs. J Supercomput, 2014, 68: 49–64

28 Liao X K, Xiao L Q, Yang C Q, et al. MilkyWay-2 supercomputer: system and application. Front Comput Sci, 2014, 8: 345–356

29 Guo H, Mo Z Y, Zhang A Q. A parallel module for the multi-block structured mesh in JASMIN and its applications. China J Comput Eng Sci, 2012, 34: 62–67 [郭红, 莫则尧, 张爱清. JASMIN 框架中多块结构网格拼接并行计算与应用. 计算机工程与科学, 2012, 34: 62–67]

30 Liu X, Zhang A Q, Xiao L, et al. A fast communication algorithm for parallel structured mesh applications. Chinese J Comput Phys, 2012, 29: 38–44 [刘旭, 张爱清, 肖丽, 等. 面向结构网格并行应用的一类快速通信算法. 计算物理, 2012, 29: 38–44]

31 Liu X, Cao X L, Mo Z Y. A projection load balancing algorithm for parallel adaptive computing on structured mesh. In: Proceedings of Conference on High Performance Computing China, Wuxi, 2008. 254–258 [刘旭, 曹小林, 莫则尧. 并行自适应结构网格计算中的逐层投影负载平衡算法. 见: 全国高性能计算学术年会, 无锡, 2008. 254–258]

32 Yang Z, Zhang A Q. JArena: a NUMA-aware multi-thread memory manager for parallel numerical simulation application. Institute of Applied Physics and Computational Mathematics, Technical Report IAPCM-SCNS-HPC201511. 2015 [杨章, 张爱清. JArena: 面向数值模拟混合并行应用的 NUMA 感知多线程内存管理器. 北京应用物理与计算数学研究所, 技术报告 IAPCM-SCNS-HPC201511. 2015]

33 Yan J, Tan G M, Mo Z Y, et al. Graphine: programming graph-parallel computation of large natural graphs for multicore clusters. IEEE Trans Parallel Distrib Syst, 2016, 26: 1–13

34 Zhang A Q, Mo Z Y, Yang Z. Three-level hierarchical software architecture for data-driven parallel computing with applications. China J Comput Res Dev, 2014, 51: 2538–2546 [张爱清, 莫则尧, 杨章. 数据驱动并行计算的三层软件架构设计及应用. 计算机研究与发展, 2014, 51: 2538–2546]

35 Zhang A Q, Mo Z Y, Cao X L, et al. Federation parallel computing in JASMIN and its application in multi-physics simulation. 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]

36 Menon H, Kale L. A distributed dynamic load balancer for iterative applications. In: Proceeding of International Conference for High Performance Computing, Networking, Storage and Analysis, Denver, 2013. 1–11

37 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]

38 Evans E, ed. Zhao L, Sheng H Y, Liu X, et al. trans. Domain-Driven Design: Tackling Complexity in the Heart of Software. Beijing: Posts and Telecom Press, 2010 [Eric Evans, 著. 赵俐, 盛海艳, 刘霞, 等译. 领域驱动设计: 软件核心复杂性应对之道. 北京: 人民邮电出版社, 2010]

39 Gamma E, Helm R, Johnson R, et al. Design Patterns: Elements of Reusable Object-Oriented Software. Boston: Addison-Wesley, 1994

40 Hu X, Hao L, Liu Z, et al. The development of laser-plasma interaction program LAP3D on thousands of processors. Aip Advances, 2015, 5: 087174

41 Fan Z F, Xu X W, Sun W J, et al. LARED-S: radiation fluid interface instability simulation program. In: Proceedings of the 16th National Symposium on Numerical Methods in Fluids, Fenghuang, 2013 [范征锋, 徐小文, 孙文俊, 等. 辐射流体界面不稳定性模拟程序 LARED-S. 见: 第十六届全国流体力学数值方法研讨会, 凤凰, 2013]

42 Li H, Zhou H, Liu Y, et al. Massively parallel FDTD program JEMS-FDTD and its applications in platform coupling simulation. In: Proceedings of the IEEE International Symposium on Electromagnetic Compatibility (EMC Europe), Oaks, 2014. 229–233

43 Cao X L, Zheng C Y, Zhang A Q. Development of 3D plasma particle simulation program on thousands of processors. China J Prog Natural Sci, 2009, 19: 544–550 [曹小林, 郑春阳, 张爱清. 面向数千处理器的三维等离子体粒子模拟程序研制. 自然科学进展, 2009, 19: 544–550]

44 Gao X, Fang J, Wang H. Sampling the Isothermal-Isobaric ensemble by Langevin-Dynamics. J Chem Phys, 2016, 144: 124113

Parallel algorithm and parallel programming: from specialty to generality as well as software reuse

Zeyao MO*, Aiqing ZHANG, Qingkai LIU & Xiaolin CAO

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

Abstract Parallel algorithm and parallel programming are important components of application software development on supercomputers, especially for numerical simulations. In recent years, in line with the rapid increase in the performance capabilities of supercomputers, these components have become increasingly challenging. Therefore, it is essential to produce application software that reuses these components in order to efficiently utilize supercomputers. In this paper, we draw on our experiences to argue for the necessity of promoting parallel algorithms and parallel programming research from a specialty to a general research field, as well as presenting similar arguments regarding software reuse. We also discuss key technologies that are required. Our work is particularly useful for the development of high performance application software for numerical simulations.

Keywords numerical simulation, parallel algorithm, parallel programming, application software, software reuse

Zeyao MO 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.

Aiqing ZHANG 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 a professor at IAPCM. Her research interests include high-performance computing.

Qingkai LIU was born in 1976. He received a Ph.D. degree from the Academy of Mathematics and Systems Science Chinese Academy of Sciences, Beijing, in 2005. He is currently a professor at IAPCM. His research interests include high-performance computing for numerical simulation.

Xiaolin CAO 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.