

神威太湖之光上 OpenFOAM 的移植与优化

孟德龙¹ 文敏华¹ 韦建文¹ 林新华^{1,2}

(上海交通大学高性能计算中心 上海 200240)¹ (东京工业大学 东京 152-8550)²

摘 要 神威太湖之光是最新一期 Top500 榜单上排名第一的超级计算机,峰值性能为 125.4 PFlops,其计算能力主要归功于国产 SW26010 众核处理器。OpenFOAM (Open Source Field Operation and Manipulation)是计算流体力学领域使用最广泛的开源软件包,但是由于其基于 C++ 实现,与神威太湖之光上的异构众核处理器 SW26010 的编译器不兼容,因此无法直接在该架构上有效运行。基于 SW26010 的主核/从核的体系架构移植了 OpenFOAM 的核心计算代码,并采用混合语言编程实现的方式来解决编译不兼容的问题。此外,通过寄存器通信、向量化和双缓冲等优化手段,单核组的性能较优化后的主核代码提高了 8.03 倍,较 Intel(R) Xeon(R) CPU E5-2695 v3 的串行执行性能提高了 1.18 倍。同时,将单核组的实现扩展到了神威太湖之光的大规模集群上,并进行了强可扩展性测试,256 个核组上实现了 184.9 倍的加速。采用的移植方式和优化手段也可以为其他复杂 C++ 程序在神威太湖之光上的应用提供借鉴。

关键词 计算流体力学, OpenFOAM, 异构多核处理器, 神威超级计算机

中图法分类号 TP317 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2017.10.012

Porting and Optimizing OpenFOAM on Sunway TaihuLight System

MENG De-long¹ WEN Min-hua¹ WEI Jian-wen¹ James LIN^{1,2}

(Center for High Performance Computing, Shanghai Jiao Tong University, Shanghai 200240, China)¹

(Tokyo Institute of Technology, Tokyo 152-8550, Japan)²

Abstract The Sunway TaihuLight supercomputer based on the Chinese-designed many-core processors is the world's fastest system with a peak performance of 125.4 PFlops. OpenFOAM (open source field operation and manipulation) is one of the most popular open source computational fluid dynamics (CFD) software which is written in C++ and not fully compatible with compilers on the heterogeneous many-core processor SW26010. This paper ported OpenFOAM based on SW26010's MPE (management processing element)/CPE (computing processing element) cluster architecture. To overcome the compilation incompatibility problem, we adopted the mixed-language application design. We also applied several SW26010's feature-specific optimizations on the hotspot of OpenFOAM to deliver high performance, such as the register communication, vectorization, and double buffering. The experiments on SW26010 using real datasets show that the single-CG (core group) code runs 8.03x faster than the well-tuned version on the MPE, and the performance of single-CG is 1.18x higher than the serial implementation of Intel(R) Xeon(R) CPU E5-2695 v3. We also optimized the parallel implementation of OpenFOAM and yielded speedups of 184.9x on 256 CGs. The porting methods and optimizations presented can also be referenced for other complex C++ programs to achieve high performance on SW26010.

Keywords CFD, OpenFOAM, Heterogeneous many-core processor, Sunway supercomputer

1 引言

计算流体力学是指使用数值方法,利用计算机的高速运算能力对流体流动和热传导等相关物理问题进行状态方程求

解和可视化的一门科学技术,是理解许多科学技术领域的基石,例如空气动力学、水利工程和工业制造等^[1]。计算流体力学所研究的领域在该技术出现之前对相关问题的求解验证实验代价大,人工计算量大甚至几乎不可能完成,计算流体动

到稿日期:2016-11-21 返修日期:2017-01-25 本文受国家重点研发计划(2016YFB0201400, 2016YFB0201800),日本学术振兴会 JSPS 的 RONPAKU 项目,并行计算机工程技术研究中心资助。

孟德龙(1992—),男,硕士生,主要研究方向为高性能计算;文敏华(1988—),男,工程师,主要研究方向为工程计算;韦建文(1986—),男,工程师,主要研究方向为大数据;林新华(1979—),男,高级工程师,主要研究方向为性能建模与优化, E-mail: james@sjtu.edu.cn(通信作者)。

力学的出现降低了相关工程问题的综合求解成本,扩展了研究人员的实验研究范围。目前,计算流体力学在从早期的理论阶段到最终产品的具体分析阶段均有广泛的应用^[2]。

OpenFOAM^[3]是目前使用最广泛的开源计算流体力学软件包,它基于 C++ 语言设计开发,包括一系列的预处理工具、偏微分方程求解器和后处理工具。OpenFOAM 的特点在于模型化,使得求解器的设计更为高效、灵活,软件包中包含丰富的物理模型,如拉普拉斯方程、不可压缩流动层流、不可压缩多相流及用户定制模型等。OpenFOAM 尽管功能强大,但是在求解复杂或精度要求较高的问题时存在计算量大、计算时间长的缺点,实际算例需要在可接受的时间内得到结果。

神威太湖之光是中国自主研发的超级计算机,峰值性能为 125.4 PFlops。其基于新一代的 SW26010 众核处理器设计^[4],每个处理器包含 4 个核组,每个核组包含 1 个主核和 64 个从核,从核分布在 8×8 的阵列中。其中主核上的编译器支持 C、C++ 和 Fortran 3 种编程语言,而从核上的编译器只支持 C 和 Fortran 两种编程语言。编译器的不兼容问题使得类似 OpenFOAM 的复杂 C++ 程序无法直接在从核上运行,从而无法有效利用神威太湖之光强大的计算能力。

为将太湖之光强大的计算性能应用于复杂的计算流体力学模拟中,需要解决编译器的不兼容问题,并结合其体系架构特征对求解器进行优化。基于上述研究初衷,首先对 OpenFOAM 软件包中的 3 个基础求解器和 10 个不可压缩流求解器进行研究。根据对求解器的性能分析,预处理共轭梯度算法(Preconditioned Conjugate Gradient, PCG)和预处理双共轭梯度算法(Preconditioned Bi-conjugate Gradient, PBiCG)是程序中最为耗时的部分,由于它们在原理上有一定的相似性,因此本文将主要介绍 PCG 算法的移植与优化。通过寄存器通信、向量化和双缓冲等优化手段对程序热点函数进行了调优,并对 MPI 的多节点版本进行了强扩展性的分析。

本文的贡献主要如下:

(1)解决了编译器的不兼容问题,将 OpenFOAM 软件包中的 3 个基础求解器和 10 个不可压缩流求解器移植到太湖之光上。

(2)单个核组的性能较优化后的主核代码提高了 8.03 倍,较 Intel(R) Xeon(R) CPU E5-2695 v3 提高了 1.18 倍。

(3)对 OpenFOAM 在超级计算机神威太湖之光上的并行实现进行了优化,并且在 256 个核组上实现了 184.9 倍的加速,获得了 72.2% 的强扩展效率。

本文第 2 节介绍 SW26010 架构及相关工作;第 3 节介绍纳维-斯托克斯方程和共轭梯度算法;第 4 节分别介绍混合语言编程设计、主核和从核上的优化方法;第 5 节通过实验验证优化方法的效果,并对 MPI 的多节点版本进行了强扩展性的分析;最后总结并展望下一步工作。

2 背景介绍

2.1 SW26010 众核处理器架构

SW26010 众核处理器架构如图 1 所示。该处理器分成 4 个核组,各核组间采用片上网络互连,每个核组包含 1 个主核、1 个从核集群、1 个协议处理单元和 1 个内存控制器

(Management Controller, MC)。核组内采用共享存储架构,内存与主/从核之间可通过 MC 传输数据,主核可以访问 PCI-E 接口,处理器可通过系统接口与外部设备相连。

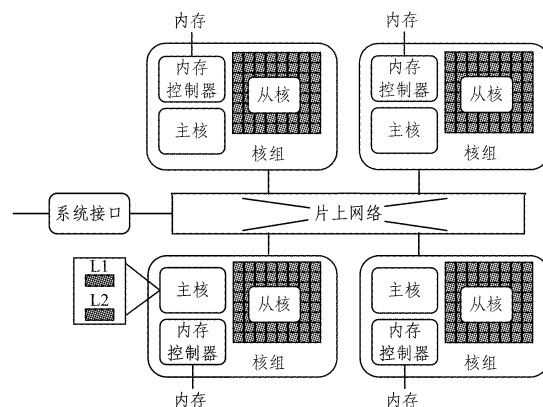


图 1 神威众核处理器架构图

主核是一个 64 位的 RISC 的核心,可以在用户模式和系统模式下运行,支持完整的中断处理、内存管理、超标量和乱序执行等功能;从核也是一个 64 位的 RISC 的核心,但是功能有限。从核只能在用户模式下运行,而且不支持中断功能,其设计目标是实现计算能力的凝集,同时最小化微架构的复杂性。从核之间采用网络互连,支持低延迟的寄存器通信。主、从核均支持 256bit 的向量操作^[5]。

主核采用传统的分层 cache 架构,拥有 32kB 的 L1 私有 cache 和 256kB 的 L2 cache。L1 cache 为 4 路组相联,行大小为 128 byte;L2 cache 为 8 路组相联,行大小为 128 byte,每个从核有 64kB 的高速存储由程序员自己控制。这种存储方式不仅省去了缓存实现上的控制开销,还避免了由从核一致性处理带来的设计复杂性和性能降级^[6],但从核上的局部存储空间需进行人为设计与规划。相比于 GPU 和 MIC,从核上的存储空间和带宽较小,这使得数据传输往往成为程序运行的瓶颈。

2.2 相关工作

近年来,OpenFOAM 作为高性能计算领域的一个非常重要的传统应用,已经成为了众多超级计算机平台上的主流应用之一,而计算流体力学中最耗时的部分就是线性方程组的求解,越来越多的研究人员开始使用 GPU 编程来改进传统的数值计算方法。

Bell 等^[7]在 GPU 上实现了一种高效的稀疏矩阵向量乘法,相较于四核 CPU,该算法取得了 10 倍以上的加速。在此基础和 Mark Harris^[8]实现的规约算法上,Andreas Klöckner^[9]实现了一个基于 CUDA 的稀疏线性系统迭代求解器并取得了较好的效果。

Thibault 等^[10]在多 GPU 平台上实现了一个针对不可压缩流体的纳维-斯托克斯方程求解器,其在 4 块 NVIDIA Tesla C870 上相较串行 CPU 代码取得了两个数量级的加速。Tölke^[11]设计并实现了一个基于 Lattice Boltzmann 方法的 GPU 算法,其相较于 CPU 上的串行代码取得了一个数量级的加速。Krawezik 等^[12]在 GPU 上加速了 ANSYS 的直接稀疏矩阵求解器,其相较于双核 CPU 取得了 2.9~4 倍的加速。

Combest 等^[13]使用 Cufflink 库将 OpenFOAM 线性求解

器移植到 GPU 上。应智^[14]研究了 GPU 上的系数矩阵运算并行方法,并实现了支持多种稀疏线性方程组求解方法的 OpenFOAM 的 GPU 并行加速插件。

上述工作都是关于 CPU/GPU 的研究,目前尚未有将 OpenFOAM 移植到神威超级计算机上的相关研究。本文做了一种新的研究尝试,即为 OpenFOAM 求解器设计了一种主从计算方法,并且使用一系列针对国产众核处理器架构的优化方法来提高 OpenFOAM 在 SW26010 上的运行效率。

3 算法介绍

线性代数方程求解模块是 OpenFOAM 中的一个重要模块,在几乎所有的偏微分方程求解器中都有应用,基础求解器和不可压缩流求解器使用共轭梯度法。

求解器首先在时间迭代中反复调用压力的隐式算子分割 (Pressure Implicit with Splitting of Operators, PISO) 算法来求解不可压缩层流纳维-斯托克斯方程。PISO 算法不使用松弛算法,在求解过程中执行多次动量修正。基于有限体积分法的不可压缩层流纳维-斯托克斯方程如下:

$$a_c u_c = -\sum_n a_n u_n + \frac{u^*}{\Delta t} - \sum_f S(p)_f \quad (1)$$

$$\sum_f S\left[\left(\frac{1}{a_c}\right)_f \nabla p_f\right] = \sum_f S\left(\frac{H(u)}{a_c}\right)_f \quad (2)$$

其中, a_n 是相较于相邻网格的系数, a_c 是网格中心的系数, f 表示变量在面中心的值, S 表示指向外部的面矢量。

该算法主要包含 3 个阶段:1)动量预测,即使用初始化或先前的压力场来求解动量方程;2)构造并求解压力方程;3)显式速度的校正阶段。

本文采用 Krylov 子空间方法迭代求解动量方程和压力方程。此方法不使用分解,而是通过矩阵和向量操作来最小化所得到的向量空间上的残差。其中,共轭梯度法是求解 $Ax=D$ 线性方程的一种重要 Krylov 子空间方法。理论上,对于对称超正定方程,只要迭代步数达到方程的阶数,利用普通的共轭梯度法就可以得到精确解;但实际上,当系数矩阵的条件数很大时,普通的共轭梯度法的收敛速度很慢。PCG 对系数矩阵作预处理,以加速迭代的收敛速度。PCG 的计算过程如图 2 所示。

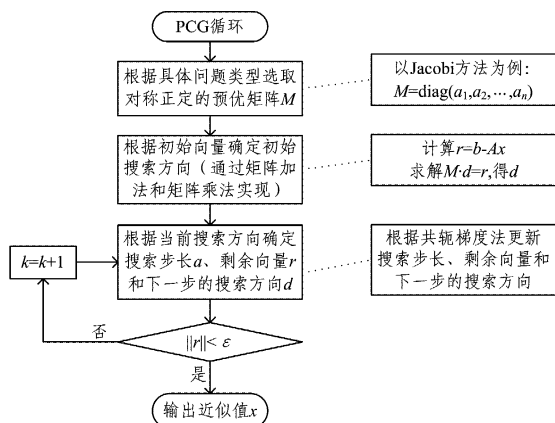


图 2 PCG 算法的流程图

4 OpenFOAM 在 SW26010 上的移植与优化

OpenFOAM 的程序特性导致其在神威处理器上的移植遇到挑战:1)OpenFOAM 是基于 C++ 开发的,而从核上的编译器只支持 C 和 Fortran,存在编译器不兼容的问题;2)OpenFOAM 默认编译模式是基于动态链接库设计的,主从计算模式不支持动态库链接;3)每个从核的局存容量为 64kB,稀疏矩阵乘法由于离散访存的特性,受到局存大小的限制。

4.1 OpenFOAM 在 SW26010 上的移植

4.1.1 C/C++ 混合语言编程实现

由于 OpenFOAM 的众多求解器基于 C++ 的面向对象的编程方式开发,因此将各种功能通过类加以封装,使用类提供的接口完成求解,将有利于用户使用,但主核上的 C++ 编译器 swg++ 不支持 SIMD 库,使得开发者无法手动调整 C++ 核心代码中的向量化计算;而且 C 编译器 sw5cc 针对 SW26010 的体系架构做出了一系列优化,可以通过编译选项对代码进行初步调优;此外,从核上的编译器不支持 C++,若要利用从核的计算能力,则需要基于 C 语言开发的核心代码。

考虑到整个程序的复杂性,难以用 C 语言重新设计,采用混合编程语言实现的方法,即使用 C 语言实现核心代码,并将其以静态链接库的方式添加进原求解器代码,其余模块仍基于 C++ 语言开发。

OpenFOAM 的核心代码包含几个比较重要的类,如 lduMatrix 和 solverPerformance。将其原有的存储格式改为数组,并通过函数来实现这些类的行为。消除掉模板和继承,并使用 sw5cc 编译核心代码,该部分代码的优化效果明显。

4.1.2 主核移植

由于 SW26010 上的 C/C++ 编译器为自主研发的,因此本文为 OpenFOAM 提出一种全新的编译模式。默认的编译模式下,第三方库和 OpenFOAM 共享同一个编译器;新的编译模式下,在编译第三方库时需要在 x86 管理节点上生成可执行工具,即分别使用 GCC 和 swg++ 编译第三方库和 OpenFOAM。

此外,主从计算模式不支持动态库链接。为解决链接问题,我们改用静态库链接,但部分函数在代码中没有显式调用,而是根据输入流确定的。为保证 OpenFOAM 功能的完整性,通过显式链接静态库中的函数项来补全列表。

4.1.3 从核并行方案

加速线程库 (athread 库) 是针对主从加速编程模型所设计的程序加速库,其目的是使用户能够方便、快捷地对核组内的线程进行控制和调度,从而更好地发挥核组内多计算核的计算性能。每个线程绑定一个计算核资源,因此在使用 athread 库时,需要考虑计算核资源的分配状况。我们使用加速线程库将 PCG 模块移植到从核上,PCG 的运算模式如图 3 所示。

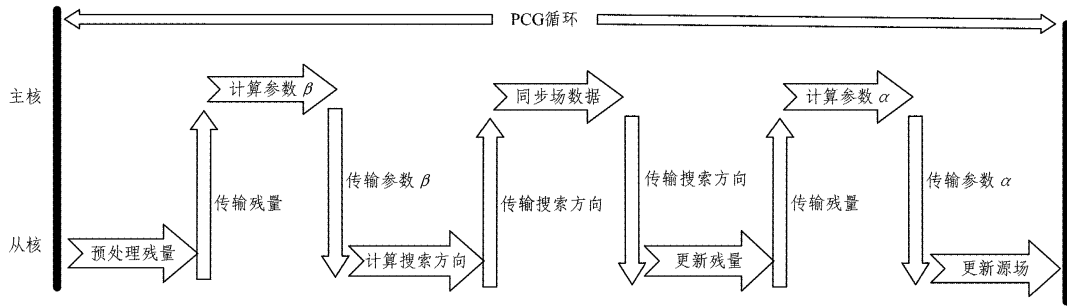


图 3 PCG 的并行方案设计

本文沿着最为连续的 x 方向将网格划分成若干个数据条,然后将每一条数据分配给一个从核,这样的划分方式使得从核计算得到的数据在空间上是连续的,额外开销比较小。

4.2 主核优化

4.2.1 向量化

在 PCG 的拉普拉斯算子计算中,数据的存储格式为 AoS (Array of Structure) 结构,而 SIMD 需要数据连续存储在内存中,否则载入向量寄存器时会造成额外的开销。本文将数据存储类型 AoS 改为 SoA (Structure of Array) 结构,并使用主核上的 256 位寄存器对 OpenFOAM 进行优化,通过 `simd_load` 函数将内存中连续的 4 个网格点的值存入一个向量寄存器 `doublev4` 中,并使用向量寄存器完成向量化计算,再通过 `simd_store` 函数将向量寄存器中 4 个浮点数的值存入内存。

此外,SW26010 上可用的向量寄存器有 32 个,寄存器数目达到上限时,若继续载入寄存器则需要清空原有数据并更新。由于 PCG 循环中常量场保持不变,因此将数据分块,并将可复用的数据存储在空闲寄存器中,如图 4 所示。在常量场相关的计算全部结束后再更新向量寄存器,以避免冗余的内存读写操作,提高程序的访存效率。

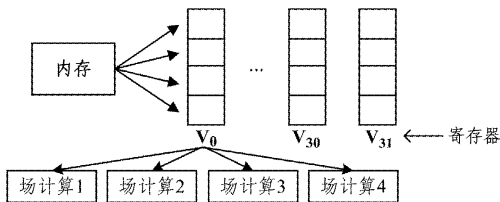


图 4 向量化

4.2.2 数据预排序

PCG 求解和 PBiCG 求解的预处理部分需要根据稀疏矩阵的标号来读取预处理向量的值,由于稀疏矩阵的列号是不连续的,因此需要读取离散存储的数据。但是预处理向量不需要随着计算循环更新,通过提前开辟空间计算并存储排序后的预处理向量,来减少程序中的离散访存。

4.2.3 优化运算

PCG 更新残量时,需要反复计算预处理向量与源场的乘积,通过改变代码的计算顺序,在预处理向量生成阶段将源场作为权值做乘后存入内存,通过将循环无关的计算提取到循环外,减少了循环内的冗余计算。此外,PCG 循环中含有几部分独立的程序段,它们均需读取搜索方向,并分别计算不同的参数和场,通过将它们融合为一个单一的循环,提高了 cache 的利用率。

4.3 从核优化

4.3.1 数据结构转换

OpenFOAM 中方程组的系数矩阵存储在 `lduMatrix` 中,其存储的形式如图 5 所示,其中对角元素和上三角及下三角元素分别存储在 3 个向量中。上、下两个三角元素的索引值共用两个索引向量,即 `UpperAddr` 和 `LowerAddr`,其中 `UpperAddr` 存储了上三角元素的行索引,同时表达了下三角对应元素的列索引; `LowerAddr` 既能表达上三角元素的列索引,又能表达下三角元素的行索引。 `lduMatrix` 格式的优点是大部分非零元不占用额外的存储空间,较为节省空间,尤其在非零元相对对角线对称分布时更是如此。

但此种格式在从核上并行的缺点在于,虽然矩阵的上半三角中每行元素是连续的,但下半三角中每行元素是不连续的,引用下三角元素进行运算时需要多次取址来确定行列索引。CSR 格式只保存稀疏矩阵中的非零元素和非零元素对应的列索引值,以及每行首个非零元素在所有非零元中的位置偏移。通过将 `lduMatrix` 格式转变为 `csrMatrix` 格式,使得计算得到的部分离散数据连续存储在内存中,以便 DMA 传输,提高并行效率。

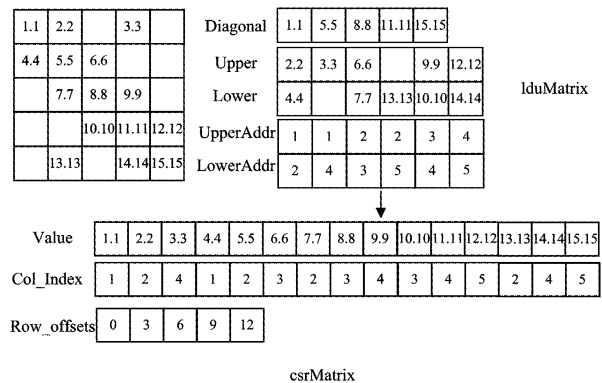


图 5 数据结构转换

4.3.2 寄存器通信

寄存器通信是从核通信的一种重要方法,通过阵列中同行/同列核心间的通信,减少了从核的冗余访存操作。

在 PCG 模块中计算搜索方向和源场时,需要使用全部残量的值计算参数 α 和 β ,而残量分散存储在众多从核上,将残量传回主核规约的效率较低,因此使用寄存器通信进行优化,通过阵列中同行或同列核心间的通信,减少了从核的冗余访存操作。而在同步搜索方向时,由于从核存储空间有限,相邻核的数据不足以完成当前计算,需要在主核更新整个向量。使用寄存器通信的 PCG 的运算模式如图 6 所示。

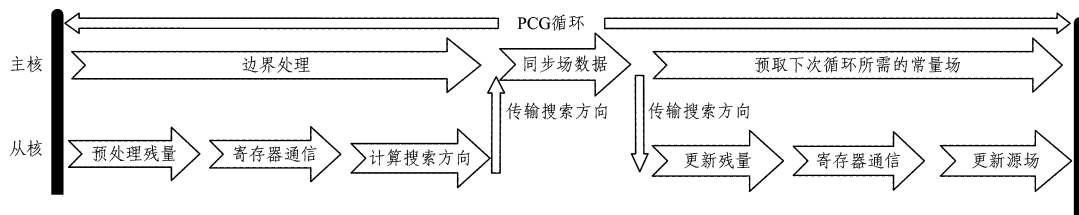


图6 使用寄存器通信的从核设计方案

4.3.3 双缓冲

在计算搜索方向时,每个网格在更新之前需要等待上个网格将搜索方向传回内存。我们对这部分计算改用双缓冲,即在从核的局存上同时维护两个缓冲区,在上个网格将数据传回内存的同时,当前网格进行计算,并将数据存储在另一片缓冲区内。

SW26010 处理器从核阵列不支持同时访存,64 个从核被分为 16 组来轮流占有带宽资源。由于计算搜索方向这一模块的计算强度较大,在该组从核计算任务未结束时已轮转到其执行 DMA 操作,因此通过双缓冲的方式使得缓冲区内的计算和访存并发执行,从而进一步提高了从核的访存效率。

4.3.4 其他优化

(1)DMA 传输优化。当从核发起 `athread_put` 和 `athread_get` 动作时,DMA 描述符需要重复初始化,将函数改成汇编指令,并提取其中的不变量。在直接使用汇编指令发起 DMA 时,可以在程序初始化时设定 DMA 描述符,在迭代过程中,数据传输量、操作属性和模式属性等参数相同的 DMA 传输可以使用同一个 DMA 描述符完成,这样便减少了 DMA 描述符的冗余设置。

(2)数据预取。在 PCG 求解模块中,首先需要对残量进行预处理,并通过寄存器通信计算参数 α 和 β ;然后利用上一步的搜索方向计算当前的搜索方向,而传输上一步的搜索方向与预处理、寄存器通信无数据依赖,因此对网格 0 的搜索方向进行预取优化,使得访存和寄存器通信并发执行,从而提高对从核流水线的利用效率。

(3)数据复用。在求解大规模算例时,需要将部分搜索方向存回内存,计算时再将其从内存载入局存中。由于局存的存储空间有限,在原版的计算模式下,将边界的搜索方向存储在缓冲区中,其余的数据需要频繁地传输,利用输入向量所占空间存储边界的搜索方向,将最后一个网格存储在缓冲区中,以节省这部分数据的访存开销。

5 实验结果与分析

5.1 测试环境及算例

本次实验使用神威众核处理器和 Intel(R) Xeon(R) CPU E5-2695 v3 处理器进行测试,具体的软、硬件环境如表 1 所列。

表 1 软、硬件环境配置

	SW26010 众核处理器	Intel(R) Xeon(R) CPU E5-2695 v3
主频(主核)	1.45GHz	2.3GHz
主频(从核)	1.45GHz	/
内存	8GB	8GB
页表大小	8kB	4kB
编译器	swg++/sw5cc	GCC/ICC

OpenFOAM 的测试版本为 2.1.1,本文选择不可压缩层流求解器 `icoFOAM` 来测试不同方法的优化效果;测试算例是一个顶盖驱动流,立方体的上边界是一个沿 x 方向移动的面,其它均是静止的面;网格是一个 $400 \times 400 \times 1$ 的二维网格,由 OpenFOAM 自带的网格生成工具 `blockMesh` 生成。

5.2 优化效果

5.2.1 主核优化效果

本文将使用 O3、预取等编译选项调优后的求解器性能作为原始版本,主核性能测试结果如图 7 所示。从图 7 可以看出,在使用以上优化策略对 PCG 进行优化后,`icoFOAM` 求解器的性能提升了 2.34 倍。

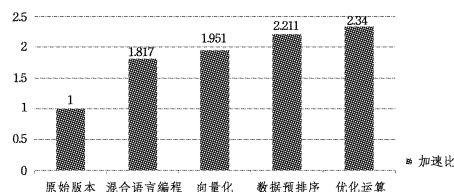


图7 主核测试结果

首先,使用 C 语言重新实现求解器的核心代码。在原始版本中,求解器直接使用 `new` 函数来申请内存空间,然而这种方法并不能保证数组的首地址在内存中是对界的,非对界的数据在访存时会造成额外的开销。我们通过 `malloc` 函数额外申请一片空间,在这片空间中选择第一个对界的地址作为数组的首地址,提高了程序的访存效率。然后通过 SIMD intrinsic 手动向量化 AoS 存储格式相关的计算,考虑到数据结构转换的开销,该部分代码实现了 2.8 倍的加速,但由于在全局中所占的时间只有 10%,因此求解器性能只有 7% 左右的提升。最后,通过数据预排序和调整运算,改善了 PCG 的访存和计算(如 4.2.2 节和 4.2.3 节所述),分别将程序的运行时间缩短了 13% 和 6%。

PCG 是一个用来求解线性代数方程的通用算法,并且它在 OpenFOAM 的大多数求解器中均有应用,因此优化的 PCG 模块对其他求解器也是有效的。我们对 3 个基础求解器和 10 个不可压缩流求解器均进行了测试,且都实现了 2 倍以上的加速。

5.2.2 从核优化效果

从核测试结果如图 8 所示。从图 8 中可以看出,在使用 6 种优化策略对 `icoFOAM` 求解器源码进行优化后,相比于主核优化版本,其性能提升了 8.03 倍。

在从核的核心函数中,首先利用向量化和 FMA (Fused Multiply-Add) 指令来加速计算,最终访存时间占程序总时间的 80%,因此使用不同的优化手段来优化从核访存。首先,CSR 的矩阵格式更适合 DMA 传输,提高了从核访存的带宽;原始版本中,PCG 算法通过主核进行残量的规约,

DMA 带宽大约是 30GB/s,而点对点通信的全片聚合带宽可达到 2043.4GB/s,广播通信的全片聚合带宽可达到 3575.9GB/s,因此通过同行/同列核心间的寄存器通信,极大地提高了数据规约的效率。

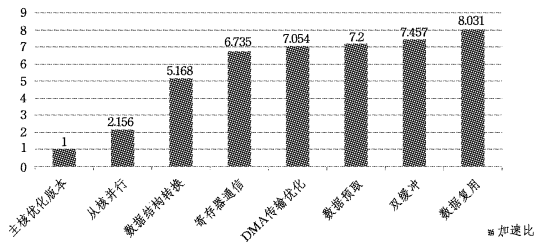


图 8 从核测试结果

从图 8 可以看出,双缓冲只有 3.6%的优化效果,因为计算搜索方向时程序性能主要受计算限制,访存几乎被隐藏,因此程序的整体加速效果不明显;数据预取可以节省一个网格的数据传输时间,在整个网格规模不大时,优化效果比较明显。

5.2.3 通用 CPU 与 SW26010 运行时间的对比分析

本节在主核、从核和 Intel(R) Xeon(R) CPU E5-2695 v3 上进行了性能对比,结果如表 2 所列。从表 2 可以看出,最终从核的性能相较于 Intel CPU 的串行执行性能提高了 1.18 倍。

表 2 Intel CPU 与 SW26010 运行时间的对比

测试算例	PCG 模块	运行时间/s	加速比
PCG icoFOAM 网格大小 (400 * 400 * 1)	主核	1044	1
	从核	130	8.03
	Intel(R) Xeon(R) CPU E5-2695 v3	154	6.80

尽管从核性能高于 Intel CPU,但 PCG 在从核上的运行效率低于 Intel CPU。通过测试发现,求解器的性能受到以下 3 方面的限制:

(1)SW26010 上的局存大小要远小于 Intel CPU 的 cache 大小,因此数据需要反复载入局存,限制了访存的性能。

(2)DMA 传输的带宽要低于 Intel CPU 访存的带宽。OpenFOAM 的低计算强度使得整个程序的性能受限于访存。

(3)稀疏矩阵相关的随机访存使得我们需要通过 gld/gst 从内存中读取一部分数据,而与 DMA 访存相比,gld/gst 是比较耗时的一个操作。

5.3 强可扩展性分析

本文在利用 pthread 库实现单核组版本的基础上,结合 MPI^[7]在太湖之光上实现了 OpenFOAM 的多节点并行版本,并对其进行了强可扩展性分析。问题规模较之前的性能对比部分增加了一个更大的输入(1185921 个网格),结果如图 9 所示。

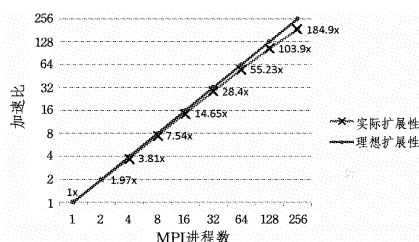


图 9 求解器在神威太湖之光上的强可扩展性分析

图 9 中的横轴代表 MPI 的进程数,1 个 MPI 进程控制一块 SW26010 芯片上的 1 个核组,即 4 个 MPI 进程控制整块芯片。对于一个 100 万网格的问题,实验的进程数从 1 扩展到 256,获得了 72.2%的强扩展效率。

结束语 本文将 OpenFOAM 中的 3 个基础求解器和 10 个不可压缩流求解器移植到 SW26010 处理器上,并根据 SW26010 处理器的体系架构对热点函数分别在主核和从核上进行了优化。

本文在 SW26010 处理器上进行了测试,主核性能在优化后相比优化之前提高了 2.34 倍,并且在此基础上,从核性能提高了 8.03 倍。在针对体系架构做出优化后,从核性能比 Intel CPU 的串行执行性能提高了 1.18 倍。在太湖之光上,OpenFOAM 的多节点并行版本在 256 个核组上的加速比为 184.9,获得了 72.2%的强扩展效率。PCG 是一个用来求解线性代数方程的通用算法,本文的优化也可以应用到其他求解器中。

此外,本文介绍的移植手段和优化方法可以为其他复杂的 C++ 程序在神威太湖之光上的应用提供借鉴,我们将硬件相关的模型编入库中,以作为一个通用的线性代数求解模块调用。

目前,OpenFOAM 可以利用神威太湖之光的计算性能,但针对多节点上的运行只是做了一些简单的优化,我们将继续优化 MPI 通信模块来提高 OpenFOAM 的并行性能;此外,我们还将使用更为底层的编程方法来深度优化单节点性能,并使用类似的策略来开发和优化 OpenFOAM 的其他模块。

致谢 感谢国家并行计算机工程技术研究中心的刘鑫、钱宏老师在本次研究中提供的大力支持;同时也要感谢上海交通大学高性能计算中心助理工程师王一超和在读硕士生许志耿在“太湖之光”超级计算机上所做的调研工作。

参 考 文 献

- [1] ANDERSON J D, WENDT J. Computational fluid dynamics [M]. New York: McGraw-Hill, 1995.
- [2] ALONAZI A A. Design and optimization of openfoam-based CFD applications for modern hybrid and heterogeneous HPC platforms[D]. King Abdullah University of Science and Technology, 2014.
- [3] WELLER H G, TABOR G, JASAK H, et al. A tensorial approach to computational continuum mechanics using object-oriented techniques[J]. Computers in Physics, 1998, 12(6): 620-631.
- [4] DONGARRA J. Report on the Sunway TaihuLight System [OL]. <http://www.netlib.org/utk/people/JackDongarra/PAPERS/sunway-report-2016.pdf>.
- [5] FU H, LIAO J, YANG J, et al. The Sunway TaihuLight super-computer: system and applications[J]. Science China Information Sciences, 2016, 59(7): 072001.
- [6] ZHENG F, ZHANG K, WU G M, et al. Architecture Techniques of Many-Core Processor for Energy-Efficient in High Performance Computing[J]. Chinese Journal of Computers, 2014, 37(10): 2176-2186. (in Chinese)
- [7] 郑方, 张昆, 邬贵明, 等. 面向高性能计算的众核处理器结构级高效能技术[J]. 计算机学报, 2014, 37(10): 2176-2186.
- [7] BELL N, GARLAND M. Implementing sparse matrix-vector

- multiplication on throughput-oriented processors[C]//Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis. ACM, 2009:18.
- [8] HARRIS M. Optimizing parallel reduction in CUDA [J]. NVIDIA Developer Technology, 2007, 2(4):511-519.
- [9] KLÖCKNER A. Iterative CUDA [OL]. <http://mathematician.de/software/iterative-cuda>.
- [10] THIBAUT J C, SENOCAK I. CUDA implementation of a Navier-Stokes solver on multi-GPU desktop platforms for incompressible flows[C]//Proceedings of the 47th AIAA Aerospace Sciences Meeting. 2009:1-15.
- [11] TÖLKE J. Implementation of a Lattice Boltzmann kernel using the Compute Unified Device Architecture developed by nVIDIA [J]. Computing and Visualization in Science, 2010, 13(1):29-39.
- [12] KRAWEZIK G P, POOLE G. Accelerating the ANSYS direct sparse solver with GPUs[C]//Proc. Symposium on Application Accelerators in High Performance Computing (SAAHPC). NC-SA, Urbana-Champaign, 2009.
- [13] COMBEST D P, DAY J. Cufflink: a library for linking numerical methods based on cuda c++ with openfoam[J/OL]. <http://cufflink-library.googlecode.com>.
- [14] YING Z. Research on Acceleration of Openfoam Based on GPU [D]. Shanghai: Shanghai Jiao Tong University, 2012. (in Chinese)
应智. 基于 GPU 的 OpenFOAM 并行加速研究[D]. 上海: 上海交通大学, 2012.
- [15] HE X, ZHOU M Z, LIU X. Design and Implementation of Multi-level Heterogenous Parallel Algorithm of 3D Acoustic Wave Equation Forwarded[J]. Computer Applications and Software, 2014, 31(1):264-267. (in Chinese)
何香, 周明忠, 刘鑫. 三维声波方程正演多级异构并行算法设计与实现[J]. 计算机应用与软件, 2014, 31(1):264-267.
- [16] XU J C, GUO S Z, HUANG Y Z, et al. Access Optimization Technique for Mathematical Library of Slave Processors on Heterogeneous Many-core Architectures [J]. Computer Science, 2014, 41(6):12-17. (in Chinese)
许瑾晨, 郭绍忠, 黄永忠, 等. 面向异构众核从核的数学函数库访问优化方法[J]. 计算机科学, 2014, 41(6):12-17.
- (上接第 54 页)
- Hefei: University of Science and Technology of China, 2016. (in Chinese)
胡福松. 基于脉搏波速度和中心动脉压的大动脉功能无创评估方法研究[D]. 合肥: 中国科学技术大学, 2016.
- [8] SUN W, TANG N, JIANG G P. Study of Characteristic Point Identification and Preprocessing Method for Pulse Wave Signals [J]. Journal of Biomedical Engineering, 2015, 32(1):197-201. (in Chinese)
孙薇, 唐宁, 江贵平. 脉搏波信号特征点识别与预处理方法研究[J]. 生物医学工程学杂志, 2015, 32(1):197-201.
- [9] JI Z, LIU X. Study on feature points recognition of pulse wave based on waveform feature and wavelet[J]. Chinese Journal of Scientific Instrument, 2016, 37(2):379-386. (in Chinese)
季忠, 刘旭. 基于波形特征和小波的脉搏波特征点识别研究[J]. 仪器仪表学报, 2016, 37(2):379-386.
- [10] 郑君里. 信号与系统[M]. 北京: 高等教育出版社, 2000:117-176.
- [11] ZHENG G, HUANG Q, YAN G, et al. Pulse waveform key point recognition by wavelet transform for central aortic blood pressure estimation[J]. Journal of Information & Computational Science, 2012, 9(1):25-33.
- [12] ZHAO Z, ZHENG G, PANG Y, et al. Study on Pulse Wave Signal Noise Reduction and Feature Point Identification[J]. Journal of Convergence Information Technology, 2013, 8(9):953-960.
- [13] ZHAO H, LI D Z, CHEN X C, et al. Sinus Bradycardia Detection Method Based on Photoplethysmography for Wearable Computing[J]. Computer Science, 2015, 42(10):25-30. (in Chinese)
赵海, 李大舟, 陈星池, 等. 基于脉搏波的人体窦性心率过缓检测方法[J]. 计算机科学, 2015, 42(10):25-30.
- [14] PELTOKANGAS M, VEHKAOJA A, VERHO J, et al. Age dependence of arterial pulse wave parameters extracted from dynamic blood pressure and blood volume pulse waves[J]. IEEE Journal of Biomedical & Health Informatics, 2015, 21(1):142-149.
- [15] KARAMANOGLU M, O'ROURKE M F, AVOLIO A P, et al. An analysis of the relationship between central aortic and peripheral upper limb pressure waves in man[J]. European Heart Journal, 1993, 14(2):160-167.
- [16] CHEB C H, NEVO E, FETICS B, et al. Estimation of Central Aortic Pressure Waveform by Mathematical Transformation of Radial Tonometry Pressure Validation of Generalized Transfer Function[J]. Circulation, 1997, 95(7):1827-1836.
- [17] GAO M, ROSE W C, FETICS B, et al. A Simple Adaptive Transfer Function for Deriving the Central Blood Pressure Waveform from a Radial Blood Pressure Waveform[J]. Scientific Reports, 2016, 6:33230.
- [18] HUOTARI M, MÄÄTTÄ K, RÖNING J. Photoplethysmographic measurements of arterial and aortic pulse waveform characteristics [J]. Finnish Journal of Ehealth & Ewelfare, 2015, 7(2-3):83-87.
- [19] WANG M Z, HE J X, ZHANG S T. The Choice of Best Multi-wavelet Base in Image Compression [J]. Computer Science, 2002, 29(s2):84-86. (in Chinese)
王梦哲, 何甲兴, 张淑婷. 基于图像压缩的最佳多小波基选择[J]. 计算机科学, 2002, 29(s2):84-86.
- [20] TANG X, SHU L, ZHENG W. Premature Best Signal Recognition Algorithm Based on Wavelet Transform and Rough Set[J]. Computer Science, 2015(b11):32-35. (in Chinese)
唐孝, 舒兰, 郑伟. 基于小波变换和粗糙集的早搏信号识别算法[J]. 计算机科学, 2015(b11):32-35.
- [21] Kumar A, Singh M. Optimal Selection of Wavelet Function and Decomposition Level for Removal of ECG Signal Artifacts[J]. Journal of Medical Imaging & Health Informatics, 2015, 5(1):138-146.
- [22] ZHANG X, XU L, CHEN K, et al. A New Method for Locating Feature Points in Pulse Wave Using Wavelet Transform[C]//2009 WRI World Congress on Computer Science and Information Engineering. IEEE Press, 2009:367-371.