

开源CFD软件SU2在Linux集群的并行计算性能编译优化研究

杨富军^{1,2}

(1. 上海海计信息技术有限公司; 2. 上海超级计算中心, 上海 201203)

摘要: 近年来斯坦福大学开发的开源CFD软件SU2, 在高性能计算领域得到越来越多的关注和应用。因此, 探究如何提高该软件的并行计算效率变得尤为重要。基于“魔方3”高性能计算集群, 通过实际案例测试, 研究了编译时所使用的编译器、MPI类型和版本、编译优化参数, 以及MPI+OpenMP混合并行模式对SU2求解器计算性能的影响。结果表明, ICC/ICPC编译器和Intel MPI表现更优异; O3级别优化、指定CPU架构、使用MKL库以及混合精度选项组合能够显著提高并行计算效率; 对于多节点并行, 设定合适的OpenMP线程数采用混合并行模式, 相比单纯的MPI并行能够进一步提高计算效率。

关键词: SU2软件; Linux集群; 并行计算; 编译优化

DOI: 10.11907/rjdk.241894

中图分类号: TP391.9

文献标识码: A

开放科学(资源服务)标识码(OSID):

文章编号: 1672-7800(2025)009-0106-06



Study on Performance Improvement of Open Source CFD Code SU2 on Linux Cluster through Compilation Optimization

YANG Fujun^{1,2}

(1. Shanghai Mass-Computing Information Technology Co., Ltd; 2. Shanghai Supercomputer Center, Shanghai 201203, China)

Abstract: In recent years, SU2, an open-source CFD software developed by Stanford University, receives increasing attention and application in the field of high-performance computing (HPC). Therefore, it is important to explore how to improve the parallel computing efficiency of the software. Based on HPC cluster named “Magic Cube III”, the study investigates the impact of the type of compiler and MPI, compilation optimization parameters, and combination of MPI and OpenMP on the parallel computing efficiency of the SU2 by solving DLR-F6 model. The results indicate that the ICC/ICPC compiler and Intel MPI perform better; O3 level optimization, specifying CPU architecture, using MKL libraries, and a combination of mixed precision options can significantly improve parallel computing efficiency; For multi-node computing, hybrid parallel mode with proper OpenMP threads setting can further improve computational efficiency than only MPI mode.

Key Words: SU2 software; Linux cluster; parallel computing; compilation optimization

0 引言

计算流体力学(Computational Fluid Dynamics, CFD)作为一门计算科学, 经过半个多世纪的发展, 已经在航空航天、汽车、船舶等诸多工业领域得到成功的应用^[1]。随着应用范围的扩展和应用水平的提高, CFD求解的问题也越来越复杂, 对计算资源的需求也越来越大。个人电脑和小型工作站已满足不了复杂模型的计算需求。更多时候研

究人员需借助高性能计算机来完成大规模计算。因此, CFD也成为高性能计算的重要应用领域之一^[2-3]。

CFD并行计算是利用区域分解思想对计算网格进行划分, 采用MPI(Message Passage Interface)方法调用多核或多计算节点进行节点内或分布式并行计算。为提高并行计算效率, 一方面需降低MPI进程间通信的开销, 另一方面需提高程序的执行效率。每个MPI进程负责一块网格分区的计算, 可以通过采用最小化边缘切割的网格分区和增加每个MPI进程中线程数等方法来减少进程间通信。

收稿日期: 2024-10-21

扫描二维码阅读全文:

作者简介: 杨富军(1979-), 男, 硕士, 上海海计信息技术有限公司工程师, 研究方向为高性能计算及其在CAE仿真计算中的应用。



此外, SIMD(Single Instruction Multiple Data)向量化技术能够有效提高程序执行效率。SIMD 向量化技术通过一条指令同时对一组数据进行相同的操作, 从而实现数据级并行, 并结合特定芯片向量指令集实现程序执行效率的提升。

SU2 是由美国斯坦福大学航空航天学院开发的一款 CFD 求解器^[4-6], 因其程序架构清晰, 获得了广泛的应用^[7-11]。随着 SU2 软件应用规模越来越大, 其并行计算性能也越来越受到关注。SU2 开发者 Economon T.D. 等从代码层面对计算性能进行不断的优化, 包括改进数据结构和访存性能, 优化代码向量化, 通过多重网格和混合并行方式减少通讯等^[12-14]。但对于应用层面的编译优化, 相关研究鲜有报道。实际应用中, 需根据不同的高性能计算集群环境, 选择合适的编译器、MPI 类型和版本, 以及相应的编译优化参数, 来提高 SU2 求解器的并行计算效率。因此, 本文基于上海超级计算中心“魔方-3”高性能计算集群, 综合考虑编译器、MPI 和优化参数等多重影响因素, 通过 DLR-F6 模型并行计算实测对比, 探究各因素对 SU2 求解器并行计算效率的影响, 以期得出更优的编译方案, 从而提高 SU2 求解器的计算效率, 为其大规模并行计算提供一定参考^[15]。

1 计算集群软硬件环境

1.1 硬件配置

SU2 算例测试在上海超级计算中心“魔方-3”高性能计算集群上进行。“魔方-3”理论峰值计算速度为 3.3 PFlops, 全机共有 1 120 台双路刀片式计算节点、35 840 个 CPU 核心。该集群配置裸容量超过 8PB 的分布式并行存储系统, 节点之间采用带宽 100Gb 的 Intel Omni-Path 高速网络互联。计算节点每个处理器含有 1 个 NUMA 节点, 每个 NUMA 节点有 16 个核心, 共 32 个核心, 具体配置见表 1。

Table 1 Configuration of computing nodes in “MagicCube III”
表 1 “魔方-3”集群计算节点配置

节点类型	CPU 型号	CPU 个数	CPU 主频	单节点内存
			/GHz	/GB
计算节点 (32cores)	Intel Xeon 6142(16 cores)	2	2.6	192

1.2 软件环境

“魔方-3”集群所有计算节点的操作系统均为 CentOS Linux release 7.5.1804, 系统自带 GCC 编译器, 版本号为 4.8.5。采用 IBM Spectrum LSF 10.1 作业管理系统。

2 SU2 源码编译介绍

2.1 编译环境要求

本文使用 SU2-7.5.1 版本源代码。该软件基于 C++ 编

写, 支持 GNU GCC/G++ 编译器、Intel ICC/ICPC 编译器和 Apple LLVM(Clang)编译器。各编译器版本要求为: GCC>=v4.7、Clang>=v3.0 或 Intel C++>=v12.0^[16]。此外, SU2 需使用 Python3 运行编译脚本。编译支持并行计算的 SU2 求解器, 并具备 MPI 工具。SU2 支持使用 OpenMPI^[17]、MPICH^[18]、IntelMPI^[19]等 MPI 工具。

2.2 编译优化选项

SU2 编译选项较多, 支持设置不同的编译器、线性代数和 MPI 类型等选项进行编译。本研究基于 SU2 官方编译优化说明, 聚焦于求解器经优化编译后的性能表现, 暂不考虑编译过程的时间、并行计算以及内存使用等方面的优化。所采用的编译优化选项如表 2 所示。其中编译器方面, 所涉及的优化选项数目众多, 经过组合后更是数目庞大。本文重点选取 0-3 不同优化级别的组合优化选项、循环展开选项、浮点计算优化选项以及向量化选项。线性代数方面, 相关选项也会对求解器的性能产生重要影响。本文对比了 MKL 和 OpenBLAS 库的性能差异, 以及是否采用混合精度^[13]对 SU2 求解器性能的影响。MPI 类型方面, SU2 支持单纯 MPI 和 MPI+OpenMP 的混合并行模式。通过配置相应的编译选项, 对 OpenMPI、MPICH 和 IntelMPI 三种 MPI 类型和 MPI+OpenMP 并行模式进行研究。

Table 2 Compilation optimization options of SU2
表 2 SU2 编译优化选项

优化方向	对应选项	说明
编译器	--optimization=level	指定优化编译级别 level: 0-3, 对应 O0-O3
	-funroll-loops	循环展开, 可以减少循环的次数
	-ffast-math (GCC)	允许违反 IEEE 规则, 提高浮点运算速度
	-fp-model fast=2 (ICC)	
线性代数	-march=?? -mtune=??	指定代码编译的 CPU 架构
	-Denable-mkl	支持 Intel MKL 线性代数库
	-Denable-openblas	支持 OpenBLAS 线性代数库
MPI	-Denable-mixedprec=true	支持混合精度
	-Dwith-mpi	支持 MPI (OpenMPI、MPICH 和 IntelMPI)
	-Dwith-omp	支持 OpenMP

3 研究方案设计

3.1 研究思路

本研究将使用 GCC 和 ICC 两款编译器, 与 OpenMPI、MPICH 和 IntelMPI 三款 MPI 工具组合, 并选取不同的优化选项来编译 SU2 求解器。通过实际案例计算性能的测试对比, 分析不同编译优化选项对求解器性能的影响。

对于编译器和 MPI 工具的选择, ICC 和 IntelMPI 直接选用 oneAPI 2023 套件, 而 GCC、OpenMPI 和 MPICH 则通过实际案例测试对比不同版本对求解器性能的影响, 进而确定相对最优的版本。随后, 以上述选定的编译器和 MPI 工具编译出的 SU2 作为基准求解器, 通过实际案例测试对比

分析其他编译选项的优化效果。最后,将效果显著的优化选项进行组合,编译出进一步优化的SU2求解器,并与基准求解器进行对比,计算效率的提升效果。测试数据取自5次运行中耗时最短的结果,且跨节点计算时所用节点均处于同一交换机。

3.2 测试算例

本文用于并行计算测试的算例模型为DLR-F6翼身组合体巡航构型。该模型是第二届和第三届AIAA阻力预测研讨会所采用的标准算例之一^[20]。该测试算例网格数约525万。模型的边界条件设置为马赫数0.75、攻角 0.49° 、SA湍流模型,采用稳态求解,迭代步数设为200步。对于编译优化选项的对比测试统一采用16核并行计算,并使用相同计算节点。由于本研究中测试算例主要用于对比SU2求解器的计算效率,故不考虑计算结果的精度。测试过程中通过调整松弛因子,减缓收敛速度,以保证实测计算都能达到指定迭代步数。

4 测试结果与分析

4.1 编译器和MPI的影响

目前,高性能计算领域常用MPI工具主要包含OpenMPI、MPICH和IntelMPI。其中,OpenMPI和MPICH为开源MPI工具,可以通过不同的编译器对其不同的版本源代码进行编译。为确定本研究使用的OpenMPI和MPICH,以及编译器的版本,首先使用不同版本的GCC/G++分别编译开源MPI的主要版本,然后以对应版本的GCC/G++和MPI编译出SU2求解器,并基于实际案例对其性能进行测试和对比分析。IntelMPI由于无需编译,故不纳入对比范围。

图1和图2展示了采用不同版本GCC/G++编译器分别与OpenMPI和MPICH组合编译的SU2求解器在实际案例中的性能对比测试结果。

由图1和图2可以看出,通过升级GCC/G++版本,计算耗时显著减少,所编译的求解器性能得到进一步提升。这

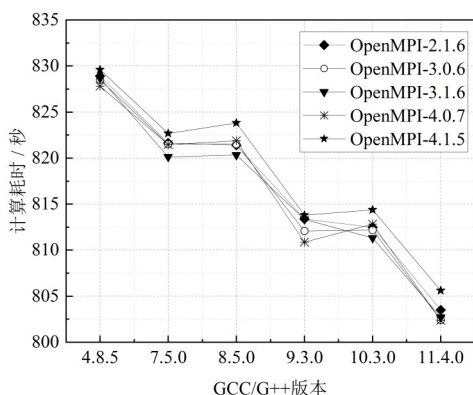


Fig. 1 The impact of OpenMPI and GCC/G++ versions on SU2 performance

图1 OpenMPI与GCC/G++版本对SU2性能的影响

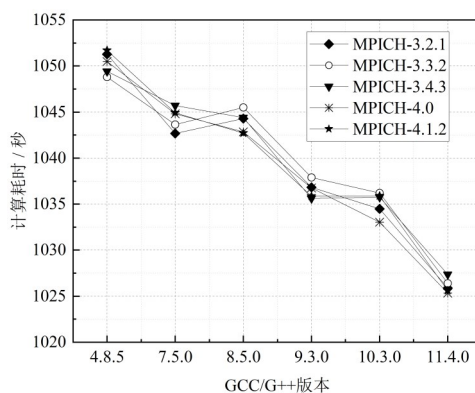


Fig. 2 The impact of MPICH and GCC/G++ versions on SU2 performance

图2 MPICH与GCC/G++版本对SU2性能的影响

主要是因为高版本编译器采用了更先进的优化技术,使得在所选版本编译器中,高版本编译的程序执行效率更高。在GCC/G++版本相同的情况下,不同版本的OpenMPI和MPICH所对应的实测结果相差很小,基本在0.5%以内。根据这一结果,最终选择使用GCC-11.4.0版本作为编译器,MPI则选择在对应该GCC版本下性能略有优势的OpenMPI 4.0.7版本和MPICH 4.0版本。

4.2 基准求解器对比

根据以上确定的GCC、OpenMPI和MPICH版本,对两款编译器和三款MPI工具进行组合,在仅使用--optimization=3优化选项的情况下,编译对应的SU2,作为测试分析的基准求解器,并通过测试算例进行实测。基准求解器的实测性能对比如表3所示。从编译器的角度来看,ICC/ICPC编译的求解器优于GCC/G++编译的求解器;在MPI方面,IntelMPI性能最优,其次为OpenMPI,最后为MPICH。此外,对于同类型的MPI,基于ICC/ICPC编译器SU2能获得更好的计算性能。这主要得益于Intel对其编译器ICC/ICPC以及MPI在Intel处理器上的针对性优化。而OpenMPI和MPICH则更注重开源实现和跨平台支持。

Table 3 Computing performance comparison of SU2 benchmark solvers

编译器类型和版本	MPI类型和版本	计算耗时/s
GCC/G++-11.4.0	OpenMPI-4.0.7	810.6
ICC/ICPC-2021.10.0	OpenMPI-4.0.7	691.4
GCC/G++-11.4.0	MPICH-4.0	1 036.3
ICC/ICPC-2021.10.0	MPICH-4.0	866.7
ICC/ICPC-2021.10.0	IntelMPI-2021.10	675.5

4.3 编译器优化选项

对比不同编译器优化级别对SU2性能的影响,编译过程对选项--optimization=level分别设定0、1、2的level值,并使用与基准求解器相同的编译器和MPI组合编译SU2求解器,将实测结果与相对应的基准求解器结果进行对比,得出相对于基准求解器的性能差异百分比,结果如表4

所示。正数表示性能提升,负数表示性能下降。

Table 4 The impact of compiler optimization level on the computing performance of SU2 solver

优化参数	MPI 与编译器类型				
	OpenMPI_	OpenMPI_	MPICH_	MPICH_	IntelMPI_
	GCC/G++	ICC/ICPC	GCC/G++	ICC/ICPC	ICC/ICPC
--optimization=0	-419.3	-682.0	-324.4	-546.0	-694.2
--optimization=1	-11.5	-34.5	-7.2	-28.0	-40.6
--optimization=2	-2.6	-0.4	-1.6	-0.3	-0.1

实验结果表明,0级和1级优化主要是针对编译所需要的时间和空间,2级和3级优化则增加了寄存器和指令方面的优化;2级优化效果略低于3级,GCC/G++编译器降幅约2%,ICC/ICPC编译器降幅小于0.5%;1级选项较3级选项降幅明显,完全无优化的0级选项编译的求解器性能则数倍的下降。这一现象的原因在于,更高级别的优化虽然增加了编译的时间和空间开销,但却显著提升了程序的执行效率。此外,从表中可以看出,ICC/ICPC编译器在使用1级和0级选项时较GCC/G++编译器下降幅度更大。

在--optimization=3选项基础上,分别添加循环展开-funroll-loops^[21]、浮点优化-ffast-math(GCC/G++)^[21]/-fp-model fast=2 (ICC/ ICPC)^[22]和向量化-march=skylake -mtune=skylake选项^[16],编译得到对应的SU2求解器,并通过实测与相对应的基准求解器对比3个选项的优化效果。与基准求解器性能差异的百分比结果,如表5所示。

Table 5 The impact of compilation optimization options on the computing performance of SU2 solver

优化参数	MPI 与编译器类型				
	OpenMPI_	OpenMPI_	MPICH_	MPICH_	IntelMPI_
	GCC/G++	ICC/ICPC	GCC/G++	ICC/ICPC	ICC/ICPC
-funroll-loops	0.2	0.5	0.3	0.2	0.1
-ffast-math(GCC)					
-fp-model fast=2 (ICC)	8.3	0.1	6.4	0.0	0.2
-march=skylake	1.9	3.4	1.6	2.7	17.2
-mtune=skylake					

由表5可知,添加-funroll-loops选项编译的求解器性能较基准求解器略有提高,但幅度均小于0.5%,说明编译器的2级和3级优化选项,会使得编译器自身进行代码优化,此时手动循环展开对编译的优化效果已不明显。从浮点优化选项-ffast-math(GCC/G++)/-fp-model fast=2 (ICC/ICPC)结果来看,基于GCC/G++编译器的求解器性能分别提升8.32%(OpenMPI)和6.37%(MPICH),而基于ICC/ICPC编译器的求解器性能几乎没有变化。浮点优化选项能够提高浮点运算的速度,但超出IEEE标准,可能会导致精度降低甚至出现数学错误。因此,需在确保计算结果准确性的情况下使用。SU2中一些数值方案支持向量化,需通过

-march和-mtune参数指定CPU架构。根据CPU型号,指定skylake架构。基于OpenMPI和MPICH的求解器性能有1%至4%的小幅提升。而IntelMPI提升幅度达17%,说明在同样指定CPU类型的情况下,同为Intel体系架构下的程序和芯片匹配性更好,能够充分利用寄存器长度,程序执行指令向量化效果明显。

4.4 线性代数优化选项

在流体结构相互作用问题中使用高阶有限元求解器或径向基函数(RBF)插值,SU2建议通过选项-Denable-mkl或-Denable-openblas链接BLAS库进行编译^[23-25]。此外,SU2支持使用-Denable_mixedprec=true混合精度选项来加速线性求解器和预处理器。为考察这三个选项对SU2求解器性能的影响,同样在--optimization=3选项基础上,分别添加这3个选项,编译得到对应的SU2求解器,并通过实测与基准求解器对比,结果如表6所示。

Table 6 The impact of linear algebra compilation options on the computing performance of SU2 solver

优化参数	MPI 与编译器类型				
	OpenMPI_	OpenMPI_	MPICH_	MPICH_	IntelMPI_
	GCC/G++	ICC/ICPC	GCC/G++	ICC/ICPC	ICC/ICPC
-Denable-openblas	0.2	0.5	0.0	0.2	0.1
-Denable-mkl	6.2	14.5	4.1	13.7	13.1
-Denable-mixed-prec=true	17.3	18.9	24.1	24.1	16.6

由表6可知,OpenBLAS库对求解器计算性能的提升效果有限,提升幅度均未超过1%。相比之下,Intel公司基于Intel处理器在性能和多线程方面做了高度优化的MKL库,其提升明显,对于GCC/G++编译的求解器提升约为4%至6%,而对于ICC/ICPC编译的求解器的提升则达到约14%。使用混合精度选项后,线性求解器和预处理器计算则采用单精度,其余部分仍保持双精度。这种策略既提高了计算效率,又维持了结果精度,从而对程序性能带来了更为显著的优化,OpenMPI和IntelMPI的求解器性能提升在16%~19%之间,而MPICH的求解器性能提升约为24%。

4.5 混合MPI

SU2-7.5.1支持混合并行模式,编译时添加-Dwith-omp选项来实现。通过MPI+OpenMP的组合的并行计算模式,同核数下可以减少CFD模型分区,即MPI进程数,从而减小进程间通讯、区域边缘重复计算和负载均衡等方面的开销^{[13][25]}。但在一个MPI进程内,多个OpenMP线程共享数据,也会造成数据访问的冲突,影响计算效率。为研究混合并行的优化效果,在--optimization=3选项基础上,均使用ICC/ICPC编译支持混合并行的SU2求解器。图3对比了同样使用128核情况下,3种MPI混合并行求解器在不同线程数下的计算耗时。

由图3可知,3种MPI工具均在每MPI进程4线程时计

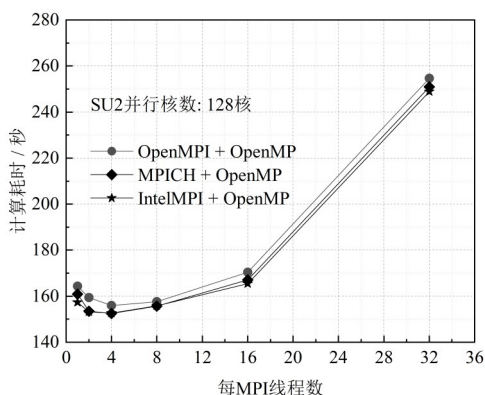


Fig. 3 The computation time varies with the number of threads in each MPI process (2 sockets 32 cores per node)

图3 计算耗时随每MPI进程中线程数的变化(双路32核节点)

算效率最佳,其次是2线程和8线程,效率均优于单纯使用MPI。每MPI进程16线程时效率略低,说明OpenMP造成的数据访问冲突已产生负面作用。根据节点架构,每MPI进程32线程时已跨2个NUMA节点,内存访问效率大大降低,计算效率也受此影响大幅下降^[26-27]。实验结果表明,一个NUMA节点应至少分配一个MPI进程,避免线程数据跨NUMA节点访问。从整体来看,IntelMPI的计算效率最高,MPICH紧随其后,且两者的表现均优于OpenMPI。

4.6 优化求解器对比

基于上述测试结果,在基准求解器编译选项的基础上,同时增加CPU架构、MKL库和混合精度3个优化效果显著的选项,编译对应的SU2优化求解器,并通过实际算例进行测试。由于本文未对CFD结果进行对比,故暂不考虑使用对于GCC/G++效果较好的浮点优化选项进行测试。基准求解器和优化求解器性能对比,如图4所示。

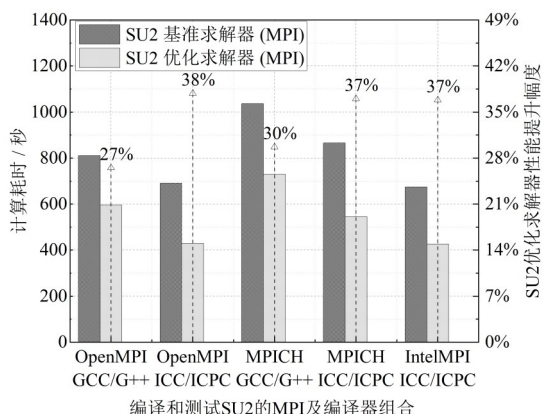


Fig. 4 Performance improvement of optimized SU2

图4 优化后SU2性能提升情况

由图4可知,优化选项组合后,SU2的性能得到了进一步提升,且提升幅度基本呈现各优化选项效果的叠加。基于GCC/G++优化编译的SU2求解器性能分别提升约27%

(OpenMPI)和30%(MPICH)。而基于ICC/ICPC优化编译的SU2求解器性能则提升超过37%。其中,使用IntelMPI的计算耗时最短,而OpenMPI的表现则与之接近。

5 结语

在上海超级计算中心“魔方3”集群上,基于3种MPI工具(OpenMPI、MPICH和IntelMPI),通过实测对比了不同编译选项对SU2求解器的并行计算性能影响。并基于此,选择了优化效果显著的编译选项进行组合,从而进一步优化了SU2求解器。实验结论如下:

(1)Intel芯片架构下,基于ICC/ICPC和IntelMPI的SU2并行计算性能更优。

(2)编译器选项中,应使用3级优化,并通过参数指定CPU架构。此外,浮点优化参数对于使用GCC/G++编译器优化效果明显,但需在确保结果精度的前提下使用。

(3)线性代数优化选项中,使用MKL库和混合精度选项均有明显提升,前者提升幅度26%至30%,后者提升约37%。

(4)对于支持混合并行模式的SU2求解器,每MPI进程2、4和8线程时均优于单纯使用MPI,其中,4线程时计算效率最佳。

(5)求解器的性能提升是编译选项优化效果的叠加,通过增加CPU架构、MKL库和混合精度选项,相较于基准求解器,性能提升最高可达38%。

参考文献:

- [1] WANG P Y, DING J H. Application and practice of high performance computing in industry and engineering[J]. E-science Technology & Application, 2012,3(6):3-11.
王普勇,丁峻宏. 高性能计算在工业工程领域的应用[J]. 科研信息化技术与应用, 2012,3(6):3-11.
- [2] ZHANG L P, DENG X G, HE L, et al. The opportunity and grand challenges in computational fluid dynamics by exascale computing[J]. Acta Aerodynamica Sinica, 2016,34(4):405-417.
张来平,邓小刚,何磊,等. E级计算给CFD带来的机遇与挑战[J]. 空气动力学学报, 2016,34(4):405-417.
- [3] GU B B, WU H, CHI X B, et al. Analysis of development of high performance computing applications[J]. E-science Technology & Application, 2014,5(4):82-91.
顾蓓蓓,武虹,迟学斌,等. 国内外高性能计算应用发展概况分析[J]. 科研信息化技术与应用, 2014,5(4):82-91.
- [4] PALACIOS F, COLONNO M R, ARANAKE A C. Stanford University unstructured (SU2): an open-source integrated computational environment for multiphysics simulation and design[C]// Texas: 51st AIAA Aerospace Sciences Meeting, 2013.
- [5] ECONOMON T D, PALACIOS F, COPELAND S R, et al. SU2: an open-source suite for multiphysics simulation and design[J]. AIAA Journal, 2015,54(3):828-846.
- [6] ECONOMON T D, PALACIOS F, ALONSO J J. Unsteady continuous adjoint approach for aerodynamic design on dynamic meshes[J]. AIAA Jour-

- nal, 2015, 53(9):2437-2453.
- [7] CUI Q, QIAN X H, LIU J M. Research on blunt body tip vortex based on open source CFD software SU2[J]. Journal of Shenyang Aerospace University, 2020, 37(3):24-32.
崔青, 钱晓辉, 刘剑明. 基于开源计算流体力学软件SU2的钝体翼尖涡研究[J]. 沈阳航空航天大学学报, 2020, 37(3):24-32.
- [8] PALACIOS F, ECONOMON T D, WENDORFF A D, et al. Large-scale aircraft design using SU2 [C]//Florida: 53rd AIAA Aerospace Sciences Meeting, 2015.
- [9] MISHRA A A, MUKHOPADHAYA J, IACCARINO G. Uncertainty estimation module for turbulence model predictions in SU2[J]. AIAA Journal, 2019, 57(3):1066-1077.
- [10] KEEP J A, VITALE S, PINI M, et al. Preliminary verification of the open-source CFD solver SU2 for radial-inflow turbine applications[J]. Energy Procedia, 2017, 129: 1071-1077.
- [11] VITALE S, PINI M, COLONNA P. Multistage turbomachinery design using the discrete adjoint method within the open-source software SU2 [J]. Journal of Propulsion and Power, 2020, 36(3):465-478.
- [12] ECONOMON T D, MUDIGERE D, BANSAL G, et al. Performance optimizations for scalable implicit rans calculations with SU2[J]. Computers and Fluids, 2016, 129: 146-158.
- [13] GOMES P, ECONOMON T D, PALACIOS R. Sustainable high-performance optimizations in SU2 [C]// Online: AIAA Scitech 2021 Forum, 2021.
- [14] ECONOMON T D, PALACIOS F, ALONSO J J. Towards high-performance optimizations of the unstructured open-source SU2 suite[C]//Florida: AIAA Infotech @ Aerospace, 2015.
- [15] YANG F J. Introduction to the "MagicCube-3" supercomputer system [J]. Development & Application of High Performance Computing, 2019, 67(3):45-49.
杨富军. 魔方-3超级计算机系统介绍[J]. 高性能计算发展与应用, 2019, 67(3):45-49.
- [16] SU2 Community. Build SU2 on Linux/MacOS[EB/OL]. https://su2code.github.io/docs_v7/Build-SU2-Linux-MacOS/.
- [17] Open MPI. open source high performance computing [EB/OL]. <http://www.open-mpi.org>.
- [18] MPICH. High-performance portable MPI [EB/OL]. <https://www.mpich.org/>.
- [19] Intel® MPI Library [EB/OL]. <https://www.intel.com/content/www/us/en/developer/tools/oneapi/mapi-library.html>.
英特尔并行计算工具库[EB/OL]. <https://www.intel.com/content/www/us/en/developer/tools/oneapi/mapi-library.html>.
- [20] LAFLIN K R, KLAUSMEYER S M, ZICKUHR T. Summary of data from the second AIAA CFD drag prediction workshop [C]// Nevada: 42nd AIAA Aerospace Sciences Meeting and Exhibit, 2004.
- [21] Optimize Options - Using the GNU Compiler Collection (GCC) [EB/OL]. <https://gcc.gnu.org/onlinedocs/gcc-4.1.1/gcc/Optimize-Options.html>.
GNU 编译器(GCC)优化选项[EB/OL]. <https://gcc.gnu.org/onlinedocs/gcc-4.1.1/gcc/Optimize-Options.html>.
- [22] Porting Guide for ICC Users to DPCPP or ICX [EB/OL]. <https://www.intel.cn/content/www/cn/zh/developer/articles/guide/porting-guide-for-icc-users-to-dpcpp-or-icx.html>.
ICC 编译器迁移至 DPCPP 或 ICX 编译器用户指南[EB/OL]. <https://www.intel.cn/content/www/cn/zh/developer/articles/guide/porting-guide-for-icc-users-to-dpcpp-or-icx.html>.
- [23] ABERGO L, MORELLI M, GUARDONE A. Aerodynamic shape optimization based on discrete adjoint and RBF [J]. Journal of Computational Physics, 2023, 477:1-25.
- [24] ZANELLA A, ABERGO L, CACCIA F, et al. Towards an open-source framework for fluid - structure interaction using SU2, MBDyn and preCICE [J]. Journal of Computational and Applied Mathematics, 2023, 429:1-12.
- [25] GOMES P C. A high-performance open-source framework for multiphysics simulation and adjoint-based shape and topology optimization [D]. London: Imperial College London, 2021.
- [26] JIN H, JESPERSEN D, MEHROTRA P, et al. High performance computing using MPI and OpenMP on multi-core parallel systems [J]. Parallel Computing, 2011, 37(9):562-575.
- [27] LI B, HUANG D Q, JIA J F, et al. Research on optimization of heterogeneous stencil computing based on CPU and GPU [J]. Computer Engineering, 2023, 49(4):131-137.
李博, 黄东强, 贾金芳, 等. 基于CPU与GPU的异构模板计算优化研究[J]. 计算机工程, 2023, 49(4):131-137.

(责任编辑:陈维捷)