GRAPES 区域模式的输入输出分析和优化*

杨 斌[1,2],王敬宇[3],刘卫国[1,2],蔡蕙伊[2],于 翡[4,5],邓莲堂[4,5],黄丽萍[4,5]
(1.山东大学 软件学院,山东 济南 250101;2.国家超级计算无锡中心,江苏 无锡 214072;
3.国家并行计算机工程技术研究中心,北京 100086;4.中国气象局地球系统数值预报中心,北京 100081;
5.灾害天气国家重点实验室,北京 100081)

摘 要:新一代全球/区域多尺度统一的同化与数值预报系统 Global/Regional Assimilation and PreEdiction System(GRAPES)是中国气象局(China Meteorological Administration,CMA)自主研发的数值天气预报软件。随着对模式分辨率和预测时效性要求的提高,GRAPES 的输入输出(I/O)性能成为了一个重要的瓶颈。分析了 GRAPES 区域模式的 I/O 行为,提出并设计实现了一个高性能 I/O 框架。该框架采用二进制编码以及多 I/O 通道技术实现了灵活可配置的输出方式。同时,通过非堵塞通信的方式实现了异步 I/O,隐藏了 I/O 与通信的开销。工作在曙光"派"超级计算机上进行了测试,结果显示该框架不仅可以提高 I/O 性能达到 10 倍以上,也可以减少性能抖动带来的性能不确定性问题。

关键词:I/O 优化;异步 I/O;GRAPES;CMA-MESO;区域模式
中图分类号:TP391 **文献标识码**:A DOI:10.16157/j.issn.0258-7998.212422

中文引用格式:杨斌,王敬宇,刘卫国,等. GRAPES 区域模式的输入输出分析和优化[J].电子技术应用,2022,48(1):39-45,52.
英文引用格式:Yang Bin,Wang Jingyu,Liu Weiguo,et al. Input/Output analysis and optimization for GRAPES regional model[J]. Application of Electronic Technique,2022,48(1):39-45,52.

Input/Output analysis and optimization for GRAPES regional model

Yang Bin[1,2],Wang Jingyu[3],Liu Weiguo[1,2],Cai Huiyi[2],Yu Fei[4,5],Deng Liantang[4,5],Huang Liping[4,5]
(1.School of Software,Shandong University,Jinan 250101,China;2.National Supercomputing Center in Wuxi,Wuxi 214072,China;
3.National Research Center of Parallel Computer Engineering & Technology,Beijing 100086,China;
4.Numerical Weather Prediction Center of CMA,Beijing 100081,China;5.State Key Laboratory of Severe Weather,Beijing 100081,China)

Abstract:The new generation Global/Regional Assimilation and PreEdiction System(GRAPES) is a homegrown numerical weather prediction software developed by China Meteorological Administration(CMA). As the requirements for model resolution and prediction timeliness increase, the Input/Output(I/O) performance of GRAPES becomes a critical performance bottleneck. This paper performs a deep analysis of I/O behavior for the GRAPES regional model,and proposes, designs and implements a high-performance I/O framework. This framework achieves a flexible and configurable output method through binary encoding and multiple I/O channels. At the same time, asynchronous I/O is included by non-blocking communication, which hides the I/O and communication overhead. The framework has been tested on the Sugon Pai supercomputer, and the results show that the framework can not only improve I/O performance by up to over ten times but also reduce the performance uncertainty caused by performance jitter.
Key words:I/O optimization;asynchronous I/O;GRAPES;CMA-MESO;regional model

0 引言

天气预报与人民财产和生命安全休戚相关。近年来高影响天气事件(例如郑州暴雨)的发生频度和强度持续增加,严重影响经济社会发展,威胁人民生命财产安全。对高影响天气的精准预报已经成为国际热点,以及国家防灾减灾决策中的关键和迫切需求。自 20 世纪 80 年代起,数值预报已成为国际天气预报的主流发展趋势,天气形势预报时效目前达到甚至超过 7 天,而制作更精细的数值预报,提高天气预报准确率的关键是进一步提高数值预报的精确度[1]。超高分辨率模拟和高频观测资料的快速应用是当前提升数值预报精确度的关键,也是极其重要又极具挑战的计算应用。全球/区域多尺度统一的同化与数值预报系统(Global/Regional Assimilation and PreEdiction System,GRAPES)[2]应运而生,GRAPES 是

*基金项目:国家重点研发计划(2020YFB0204800);无锡市太湖人才计划创新领军人才项目(融合人工智能技术的新一代精细化区域气候预测系统研制)

我国历时超过二十年自主研发，以多尺度通用非精力动力框架为核心开发建立的可插拔式的新一代通用数值分析同化与预报系统。目前 GRAPES 已成为我国中期数值天气预报的业务化模式，不仅能够预报大尺度形势场，而且在要素预报(如降水量、风、湿、压等)准确性也在逐步提高，逐渐在预报业务一线发挥着越来越大的作用[3-5]。

随着分辨率和时效性的需求提高，改进 GRAPES 的计算效率，提高其扩展性的需求也越来越迫切。在大规模超级计算机和集群系统快速发展的过程中，可扩展性的并行算法研究受到了广泛重视，模式的计算可扩展性得到了快速的发展，出现了许多针对 GRAPES 的优化工作[6-10]。这些工作使用了新的计算方法或者新的并行算法，提高了 GRAPES 在新设备、新集群上的计算扩展性，使得 GRAPES 可以充分发挥新设备带来的高算力。

与计算相比，GRAPES 输入输出(I/O)的性能提升则相对发展缓慢，且面临多重困难。一方面，应用本身 I/O 访问十分复杂，网络和存储资源的冲突导致性能容易发生波动；另一方面，I/O 性能受制于并行文件系统的系统参数配置。因此，随着计算规模的不断增大以及网格分辨率需求的不断提高，GRAPES 的 I/O 开销逐渐成为不可忽略的一部分，I/O 性能也成为制约 GRAPES 可扩展性和整体计算效率的关键因素之一。

实际上不仅是 GRAPES，世界上其他优秀的数值模式的 I/O 性能也随着计算规模的增大而被诟病。为此，国内外的学者开始针对数值模式的 I/O 问题提出了不少解决方案。部分工作主要通过引入第三方 I/O 中间件来改善原始数值模式的 I/O 性能，其核心思想是通过将原始的串行 I/O 替换为并行 I/O 来提高 I/O 性能。Dennis[11]等人针对地球系统模式(Community Earth System Model，CESM)[12]的 I/O 问题，研发了并行 I/O 库 PIO。PIO 使用并行 I/O 的接口取代了 NetCDF[13]库提供的串行 I/O 接口，提高了 I/O 扩展性。PnetCDF[14]也是一种 NetCDF 库并行化的实现，但是相比 PIO，它无法重新划分 I/O 数据域，导致无法充分利用存储带宽。季旭等人[15]则专注于全球海洋环流模式(LASG/IAP Climate system Ocean Model version 2，LICOM2)[16]的 I/O 优化，引入了 Adaptive I/O System(ADIOS)[17]I/O 中间件，并取得了较好的优化效果。Zou 等人[18]分析了数值天气预报系统 GRAPES 的 I/O 访问模式，并且引入了 MPI-IO 和 ADIOS 两种并行 I/O 中间件来优化原始的串行 I/O。

另一类的工作则是在并行 I/O 的基础上，发掘计算与 I/O 的重叠机会，进一步优化了数值模式的 I/O 性能。Wang 等人[19]基于 PnetCDF 的接口特性，引入了 I/O 转发技术，尽可能将 I/O 和计算过程重叠，从而减少了应用的 I/O 开销。陈璟锟等人[20]则基于 Acharya 等人[21]提出的主-从架构，设计了一种重叠存储的优化技术，并将其推广到了 WRF(The Weather Research and Forecast model)[22]、ROMS(The Regional Oceanic Modeling System)[23]等常见大气与海洋模式中。这类技术的核心思想都是通过通信来替换 I/O 操作，从而减少计算流程中的 I/O 开销。廖嘉文等人[24]则将该方法进一步推广到了 FVCOM(Finite-Volume Community Ocean Model)[25]海洋模式中。

第一类工作主要通过引入 I/O 中间件来优化数值模式应用的 I/O 性能，其优势是对应用代码的改动较小，方便应用的后续移植和扩展。但是应用仍然缺乏异步输出能力[24]，计算需要等待 I/O 结束后才可以继续进行，在复杂的超算场景下，仍然会成为性能瓶颈。第二类工作主要通过进程间的同步通信去替换原先的 I/O 操作，使用通信时间来替代 I/O 时间。一般来说网络带宽会远大于 I/O 带宽，所以这些工作可以较为明显地提高 I/O 性能。但是仍然存在一些不足，在一些负载繁忙的场景下，由于接收进程无法即时接收数据，仍然会导致通信开销过大，性能提升不明显。

目前，针对 GRAPES 的 I/O 优化工作主要仍是通过引入 I/O 中间件的方式的来实现并行 I/O。在这之中，Zou 等人[18]的优化工作已经受到了 GRAPES 开发人员的广泛认可，且被集成到了 GRAPES 中。但是，随着规模的不断扩大和精度的逐步增加，GRAPES 的 I/O 仍然会成为性能瓶颈。与此同时，科研人员也对 GRAPES 的 I/O提出了新的需求，例如某项研究，研究人员对于 GRAPES 的输出需求仅包含少数的几个变量，如：位势高度、风力、方向等。但是在实际使用中，研究人员不得不等待所有预定的变量输出完毕才可以开展后续的分析。这样不仅增加了额外的 I/O 开销，也增加了存储开销，大大浪费了宝贵的资源。与此同时，也有按变量变频率输出的需求。进一步，研究人员希望一部分变量以某种固定的频率输出，而另一部分变量则按照另一种频次输出。这些新的需求，也对 GRAPES 模式当前的 I/O 方案设计和性能持续优化带来了新的挑战。

本文在现有研究的基础上，针对目前 GRAPES 复杂的 I/O 需求，从应用层出发提出了一个灵活可配置的高性能 I/O 框架。该框架通过灵活可配置的变量输出，提高了 GRAPES 的 I/O 灵活性。与此同时，该框架提供了高效的异步 I/O 优化方案，通过 MPI 异步通信的方法，完全掩盖了 I/O 及通信的时间。在曙光“派”超级计算机进行了测试，发现该 I/O 框架不仅可以提高 10 倍以上的 I/O 性能，也可以减少性能抖动带来的不确定性问题。与此同时，本文的工作已经在中国气象局基于 GRAPES 区域模式开发的作为业务系统运行的中尺度天气数值预报系统中得到实际应用。该系统自 2021 年 10 月起，正式命名为中国气象局中尺度天气数值预报系统(CMA-MESO)。

1 GRAPES 区域模式的 I/O 模块分析

GRAPES 模式分为全球和区域模式两个版本，两个版本的 I/O 模块基本一致，故本文的研究以 GRAPES 区

域模式为例开展，下文使用 GRAPES_MESO 来指代 GRAPES 区域模式。

GRAPES_MESO 的 I/O 模块包含两部分。一部分初始化中配置文件和数据文件的读入；另一部分是预报过程中的状态量文件输出。状态量文件的输出也分为两类：一类是二进制数据文件；另一类是描述文件，通常以扩展名.ctl 结尾，包含对二进制数据的文件位置及数据信息完整描述，包括网格划分、空间分辨率、时间分辨率、经纬度信息、垂直层次、变量描述等信息。由于初始化中的文件读取仅需要进行一次，而输出文件中的描述文件十分小，且不必每次都输出，因此本文的研究重点是输出二进制数据文件。图 1 展示了 GRAPES_MESO 的主要流程。

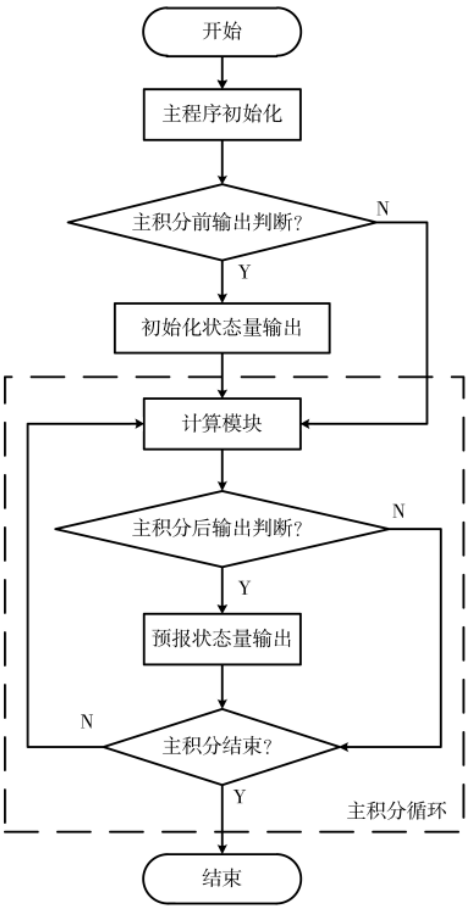

图 1 GRAPES_MESO 的主要流程示意图

程序的一开始会进行初始化，读取一些配置文件和输入数据。然后会进入到主积分模块 module_integrate.F 中。主积分模块主要包含一个主积分循环，循环由计算和 I/O 两部分组成。在主积分循环中，首先进入计算模块进行计算，每次计算结束后，会进行一次输出的判定。当判定满足输出要求，就会调用并行 I/O 模块进行预报状态量的数据输出。GRAPES_MESO 的并行 I/O 模块主要是通过 MPI-IO 来实现的。当需要进行 I/O 时，所有的计算进程会暂停计算，然后通过 MPI_Allgather 收集数据，选择其中一部分的计算进程进行 I/O 输出。GRAPES_MESO 提供了两种方式，一种是水平划分进程，这样由每行的头进程收集整理数据并进行输出。另一种是垂直划分进程，这样由每列的头进程收集整理数据并进行输出。当完成 I/O 输出后，所有的进程才会进入到下一轮的计算中去。

本文的测试平台是中国气象局的曙光“派”超级计算集群，曙光“派”的基本配置信息如表 1 所示，主要包含了计算节点上 CPU 的核心数、主频，集群的高速互联网络信息，操作系统，文件系统，以及使用的 MPI 版本信息。

表 1 曙光“派”集群基本配置信息

参数	内容
单 CPU 核心数	32
主频/GHz	3.3
网络系统	Omni-Path
操作系统	Red Hat 4.8.5-16
文件系统	ParaStor
MPI 版本	Intelmpi2017.2.174

表 2 则描述了本文测试使用的 GRAPES_MESO 版本和具体算例情况，包含预报区域经纬度、积分步长、网格数等信息。首先对原始的 GRAPES_MESO 进行了测试分析，发现其无法满足当前的需求。从功能上讲，原始 GRAPES_MESO 仅可以支持模式面或者等压面的变量按固定频率进行并行 I/O 输出，无法满足上文提到的科研人员一些灵活的输出要求。从性能上讲，每次进行 I/O 输出需要数十秒，而计算每一步仅需要上百毫秒，在需要频繁输出的场景下，I/O 开销不可忽略。

表 2 算例介绍

参数	内容
课题名称	GRAPES_MESO V4.0
算例详细信息	水平分辨率 3 km，积分迭代步数为 8 640，每 120 步输出一次；预报区域东经 70°~145.03°，北纬 15°~65.13°；垂直分层数为 50
积分步长/s	30
网格数	2501×1671×50

2 灵活可配置的高性能 I/O 框架的设计与实现

为了提高 GRAPES_MESO I/O 模块的灵活性，用以面对多样的 I/O 需求，同时持续提升 I/O 性能，本文从 GRAPES_MESO 的应用本身出发，结合其数据特征，将整个 GRAPES_MESO 的 I/O 模块进行了模块化的改造，构造了一个相对通用的高性能 I/O 框架。

图 2 展示了采用高性能 I/O 框架优化后的主积分循环的内部流程，流程内所有的进程根据其功能划分为计算进程以及 I/O 进程。图的左侧代表计算进程负责的流程，除了负责计算以外剩下的是高性能 I/O 框架的输出

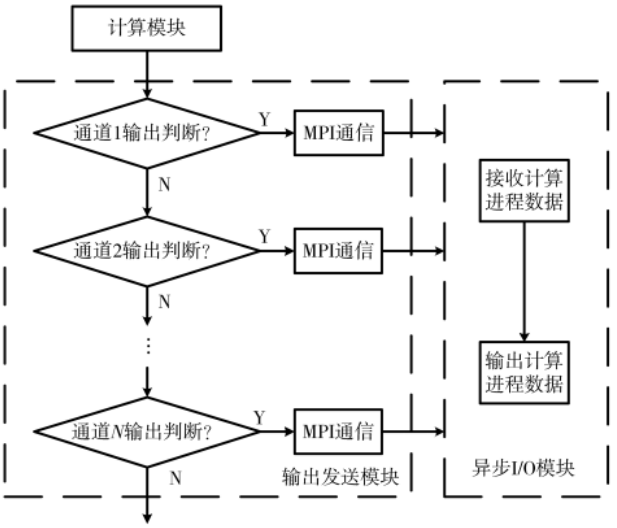

图 2　优化后的主积分循环内部流程示意图

发送模块，主要包括判断是否需要输出和发送 I/O 数据。右侧则表示由 I/O 进程组成的高性能 I/O 框架的异步 I/O 模块，负责接收计算进程发送过来的数据并输出。

I/O 通道(Channel)概念的引入，将原本复杂的 I/O 模块划分成针对不同 I/O 需求的子模块。例如：通道 1 负责标准的模式面变量的输出，通道 2 负责标准的等压面变量的输出，通道 3 负责对某些特殊变量进行可变频率的输出。各 I/O 通道相互独立，可通过配置文件单独控制。在每一轮的计算结束后，会依次判断当前迭代步数是否满足输出要求。若满足要求，则通过异步 MPI 通信将数据发送给异步 I/O 模块进行 I/O 输出，自身则无需等待，可直接进入下一轮的迭代计算。

优化后的架构可以很好地隐藏原有的 I/O 开销，相比于陈璟锟等人[20]通过 MPI 同步通信进行数据传输的方法，本架构也可以隐藏掉一部分通信的开销，可以更好地实现 I/O 性能优化的目标。与此同时，将原始 I/O 模块划分为若干子模块，引入 I/O 通道的设计，也便于后续增加新的功能来面对新的 I/O 需求，大大提高了灵活性。

2.1 进程的划分及映射

为了实现计算和 I/O 的分离，首先要做的是进程的划分。从所有的进程中选择一部分进程作为 I/O 进程，另一部分则仍为计算进程。为此，本文选择在 GRAPES_MESO 的一开始增加一个通信子分裂函数。该函数将 MPI_COMM_WORLD 全局通信子划分为两个子通信子，包含计算进程的计算通信子以及 I/O 进程的 I/O 通信子。此外，提供了两种方法来实现通信子的分裂，并且可选择手动分配或自适应配置。

图 3 展示了 I/O 框架提供的两种划分方式。布局 1 称作“独立分布”，表示计算进程和 I/O 进程在物理上尽可能地独立分布，一般根据进程的 rank 选择最后的若干进程作为 I/O 进程。布局 2 称作“共享分布”，表示计

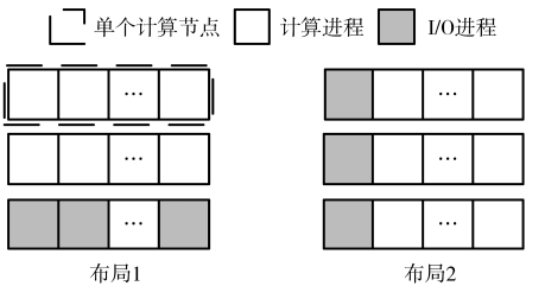

图 3　进程划分布局

算进程和 I/O 进程共享使用物理节点，一般选取同一个物理节点上的少数进程作为 I/O 进程。布局 2 在一些特定场景下可以获得更好的通信性能，具体解释可以见章节 3.1 单步 I/O 性能测试。

计算进程到 I/O 进程的映射则通过映射表来确定。分裂函数在划分通信子的同时，相应地会构建计算进程到 I/O 进程的映射表。因为 GRAPES_MESO 中的变量是以多维数组的形式散布到所有计算进程中进行存储的，所以每个计算进程的数据量都比较平均。为了保证 I/O 进程的负载均衡，构建映射表时也需要尽可能地保证每个 I/O 进程服务相同数量的计算进程。此外，为了保证 I/O 进程便于对接收到的数据进行按变量的重组，同一个 I/O 进程服务的所有计算进程需要满足变量空间上的局部性。

2.2 可配置变量输出的设计

GRAPES_MESO 本身的计算涉及了大量的预报变量，而用户往往只需要其中的一部分。在原始的 GRAPES_MESO 的 I/O 模块中，用户需要等到所有的变量输出完毕后，才可以通过后处理提取出自己想要的数据进行下一步分析。这无疑大大增加了开销，也浪费了宝贵的资源。为了解决这个问题，也有用户通过人工修改数值模式的源码，使 GRAPES 仅输出对应的变量。但是，当应用场景发生变化，用户需求发生变化时，又不得不再次去修改代码，对模式程序进行重新编译和验证正确性，这对于用户和模式应用来说，无疑是繁琐的。

为了解决这一问题，本文的 I/O 框架提供了可配置调整变量输出的功能。用户通过 namelist 配置文件，选择一个 I/O 通道，在其中输入自己想要输出的变量，设定其对应的输出频次即可。当满足要求时，对应的 I/O 通道会将要输出的数据传输给异步 I/O 模块进行输出。I/O 通道的主要流程可以分为初始化及数据发送。

2.2.1 初始化

I/O 通道的初始化主要目的是读取配置文件，确定要输出的变量的具体信息及输出频次。当有较多的变量需要输出时，存储这部分的变量也会是一个很大的开销。因此设计了专门的算法来降低存储开销，算法通过构建一个输出编码来存储指定输出的所有变量。

构建输出编码的具体流程如图 4 所示。首先将所有

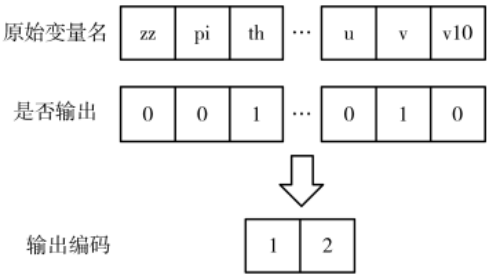

图 4　输出编码的构建示意图

的变量进行排序(任意约定的顺序),接着根据配置文件确定要输出的具体变量,标记要输出的变量为 1,其余的为 0。然后每 31 个变量为一组,按二进制构成一个整数,不足 31 位的左侧补 0,最后求出输出编码。输出编码是一组整数,决定了该 I/O 通道要输出的变量。采用此方法可以用较小的开销来存储所有要输出的变量。与此同时,用户不再需要修改程序,可以仅通过配置文件的方式,灵活地指定要数值模式输出哪些变量。

2.2.2 数据发送

I/O 通道在正式发送输出数据前,需要对输出的变量进行打包,将要发送的数据先合并到一个发送缓冲区中。通过打包,可以将小消息合并为大消息,将多次发送减少为一次发送。采用这种策略可以极大程度地减少多次通信带来的开销。

此外,由于 I/O 进程的数量会远小于计算进程,因此整个异步 I/O 模块的内存是有限的,必须对内存的水位(内存占用)加以限制。因此,I/O 通道需要获取当前异步 I/O 模块的内存水位信息 X,然后与要发送的数据的数据量 D 进行比较,如果 $X \geqslant D$,那么调用 MPI_ISEND 异步发送数据;如果 $X<D$,那么需要进入等待状态,直到 $X \geqslant D$,才可以发送数据。

发送的数据也分为两个部分,一个是输出编码,另一个是实际需要输出的数据。输出编码可以帮助异步 I/O 模块估计剩余的内存以及提取输出数据的变量信息。

2.3 异步 I/O 模块的设计

异步 I/O 模块由分裂函数划分的 I/O 进程组成,负责接收计算进程发送过来的数据,通过数据重组后进行并行 I/O 输出。图 5 展示了异步 I/O 模块的具体流程示意图。

首先是内存水位的计算。由于异步 I/O 模块中的通信均采用了非堵塞的异步通信方式,因此需要严格控制其内存的水位。当水位过高时,I/O 进程需要暂停接收计算进程传过来的数据,等待之前的数据输出完毕,水位下降后再重新接收计算进程传过来的数据。为此,异步 I/O 模块在一开始会获取当前已使用的内存情况 X_{use},然后计算空闲的内存情况 X_{free}。此外,为了保证计算的容错,X_{free} 会按照 $0.9X_{total}-X_{use}$ 来计算,其中 X_{total} 代表全部的内存大小,即 90%的内存占用已经代表满载。获取 X_{free} 后,根据服务计算进程的数量,计算水位标记 X,然

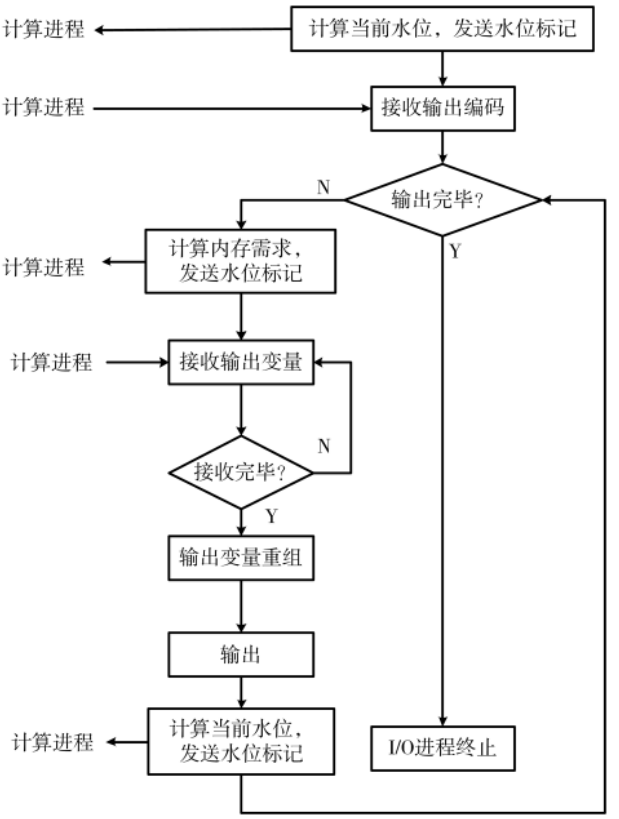

图 5　异步 I/O 模块流程示意图

后发送给对应的计算进程。

第二步是接收输出编码。输出编码接收完毕后需要进行解码操作来获取要输出的变量。同时,根据解码后的变量信息,结合变量的网格划分以及对应接收计算进程的数量来估算即将要接收的数据的大小,重新计算内存水位 X 后发送给计算进程。此外,输出编码也作为终结标记使用。此时由计算进程发送终结标记,I/O 进程获取终结标记后进入终止状态。

然后是通过 MPI 异步通信接收计算进程发送的数据。由于采用了异步通信,因此需要等到数据接收完成后才可进行下一步的重组操作。重组操作负责对接收区的数据进行变量重组,其目的是保证最终的输出是按照变量进行的。

图 6 展示了某一个变量的数据重组示意图。假设变量 A 是一个二维变量(多维的变量可以划分为若干二维的处理),其全局大小为(1:X,1:Y),被切分成 36 份散布到 36 个计算进程中。异步 I/O 模块共有 4 个 I/O 进程,分别从对应的计算进程中接收数据。由于接收的数据为打包后的数据,因此每个 I/O 进程需要提取变量 A 的数据,然后依照其在原始二维数组 A(1:X,1:Y)的位置进行重构。例如,对于图 6 左上角的 0 号 I/O 进程来说,需要组合出子变量 $A_0(1:x_1,1:y_1)$。同理,对于 I/O 进程 2、3 和 4 来说,也需要组合出子变量 $A_1(x_1:X,1:y_1)$、$A_2(1:x_1,y_1:Y)$ 以及 $A_3(x_1:X,y1:Y)$。直到所有的变量都组合完毕后,再

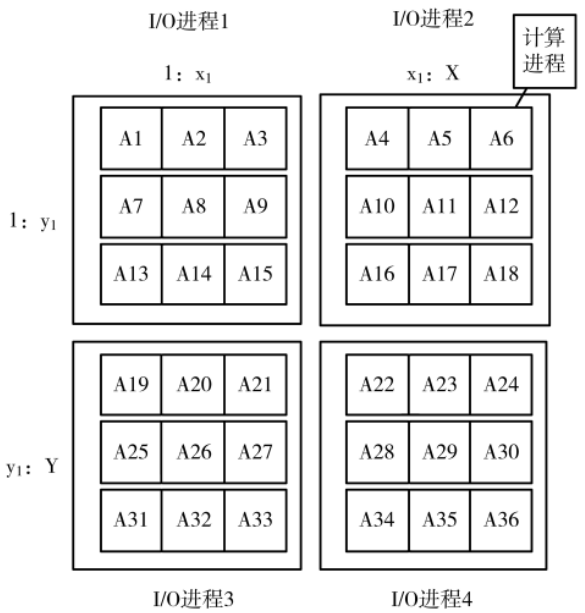

图 6　变量数据重组示意图

调用 MPI-IO 进行并行输出。

3　测试与分析

3.1　单步 I/O 性能测试

本测试主要用来测试该 I/O 框架在单次 I/O 中带来的性能提升，测试使用的算例如表 2 所示。在本次测试中，共采用了 1 024 个进程，其中 I/O 进程为 60 个。测试程序一共运行了 864 步积分，每 288 步输出一次等压面的变量，加上积分前的输出，一共需要进行 4 轮 I/O 输出。测试分别比较了数值模式应用采用原始 MPI-IO 的性能、采用共享分布方式的高性能 I/O 框架的性能、采用独立分布方式的高性能 I/O 框架的性能。图 7 展示了实验的结果，其中横轴表示进行的 I/O 轮数，每轮 I/O 包含 3 个性能结果，分别代表采用的不同 I/O 方式，纵轴为每轮 I/O 过程的时间。

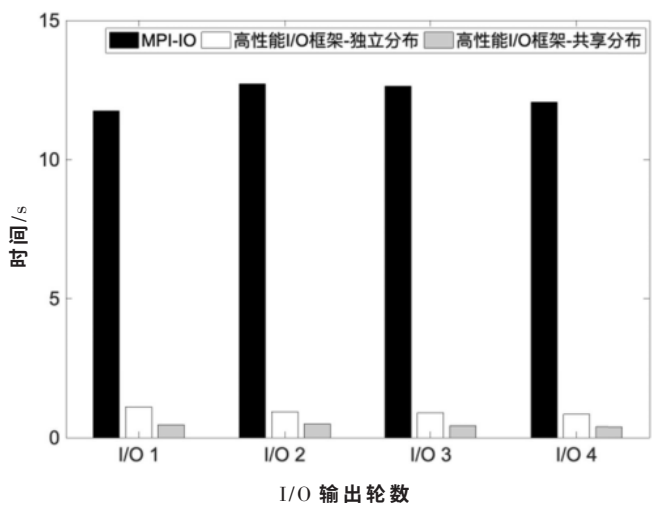

图 7　单步 I/O 性能比较图

从图 7 中可以看到，高性能 I/O 框架较原版 MPI-IO 有明显的性能提升，平均超过 10 倍。对于独立分布和共享分布来说，共享分布的性能明显优于独立分布。这是因为共享分布的情况下，有些通信操作可以在节点内部完成，而不用通过外围的交换网络进行通信，这可以帮助提高通信效率。

3.2　应用整体性能测试

本测试主要用来测试该 I/O 框架在数值模式应用需要频繁输出的场景下的整体表现，测试使用的算例如表 2 所示。此外具体的参数配置如下：测试程序的规模为 1 024 个进程，其中 60 个为 I/O 进程，一共运行 1 000 步积分，采用了三个 I/O 通道进行不同频率的 I/O 输出。通道 1 输出模式面变量，从 0 步开始，每 20 步输出一次，1 000 步结束；通道 2 输出等压面变量，从 10 步开始，每 20 步输出一次，990 步结束；通道 3 输出一些定制变量，其中一部分变量从 0 步开始输出，每 5 步输出一次，500 步结束；另一部分从 500 步开始输出，每 2 步输出一次，1 000 步结束。同时，修改了原版 GRAPES_MESO 的源码，使其在使用 MPI-IO 的情况下达到相同的输出要求。

图 8 展示了数值模式应用分别采用 MPI-IO 和高性能 I/O 框架的方式在多次运行下的情况。可以看到，在密集 I/O 的场景下，由于异步 I/O 可以很好地隐藏 I/O 和通信开销，因此相比于原来的 MPI-IO 可以获得明显的性能提升。此外，也发现采用了异步 I/O 后可以明显地降低性能的抖动。可能的原因是基于 MPI-IO 的直接存储访问有较长的数据访问路径导致性能干扰多，影响程度较大。特别是当系统繁忙时，会出现明显的性能干扰。而采用异步 I/O 的方式可以隐藏这一部分的开销，从而减少了应用性能的抖动。

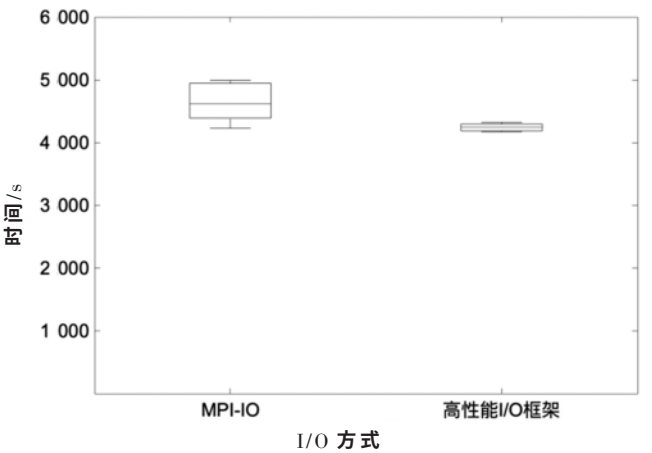

图 8　总运行时间对比图

3.3　准确性测试

本测试主要用来证明高性能 I/O 框架的设计是独立于计算流程之外的，其输出的预报变量与原版 GRAPES_MESO 输出的预报变量完全相同。比较了采用和未采用

高性能 I/O 框架前后预报变量的变化情况。表 3 列举了一些预报变量的误差对比，可以看到高性能 I/O 框架不会对预报精度造成任何影响，可以达到二进制一致性。同时，也通过后处理可视化分析进一步验证成功了数据的准确性。

表 3 预报变量对比示例

变量名	平均误差	误差方差
h	0.0	0.0
u	0.0	0.0
v	0.0	0.0

4 结论

为了应对数值模式日益增长的 I/O 需求，本文针对 GRAPES_MESO 设计了一个灵活可配置高性能 I/O 框架，且已经在 CMA-MESO 中得到业务应用。该 I/O 框架设计了单独的 I/O 进程，使得应用的计算和 I/O 过程完全分离，计算进程无需等待 I/O 结束即可进行下一轮计算。相较于原先采用的 MPI-IO 的方法，该框架不仅可以提升 10 倍以上的性能，也可以降低性能抖动。与此同时，该 I/O 框架通过水位控制策略保证了独立 I/O 模块的安全。此外，该框架还引入了二进制编码以及多 I/O 通道的设计，实现了可配置的变量输出，大大提高了应用 I/O 的灵活性，避免了无效变量输出导致的资源浪费。

虽然该框架是基于 GRAPES_MESO 的 I/O 模块实现的，但是采用的一些思想、方法也可以应用到其他的数值模式中去，例如：多 I/O 通道的设计，基于 MPI 非堵塞通信的异步 I/O，可配置变量的输出等。下一步计划将该 I/O 框架应用到其他的数值模式中去。

参考文献

[1] 陈德辉，薛纪善.数值天气预报业务模式现状与展望[J].气象学报，2004(5)：623-633.
[2] 黄丽萍，陈德辉，邓莲堂，等.GRAPES_Meso V4.0 主要技术改进和预报效果检验[J].应用气象学报，2017，28(1)：25-37.
[3] 于翡，黄丽萍，邓莲堂，等.GRAPES-MESO 模式不同空间分辨率对中国夏季降水预报的影响分析[J].大气科学，2018，42(5)：1146-1156.
[4] 叶成志，欧阳里程，李象玉，等.GRAPES 中尺度模式对 2005 年长江流域重大灾害性降水天气过程预报性能的检验分析[J].热带气象学报，2006(4)：393-399.
[5] 徐双柱，张兵，谌伟.GRAPES 模式对长江流域天气预报的检验分析[J].气象，2007(11)：65-71.
[6] 刘钊.基于国产高性能计算机的 GRAPES 性能优化研究[D].上海：上海交通大学，2014.
[7] 王卓薇，许先斌，赵武清，等.基于 GPU 的 GRAPES 模型并行加速及性能优化[J].计算机研究与发展，2013，50(2)：401-411.
[8] 王为，张悠慧，姚骏，等.基于线性阵列处理器的 GRAPES 核心代码优化[J].计算机学报，2013，36(10)：2053-2061.
[9] 郭妙.基于 GPGPU 系统的 GRAPES-GLOBAL 长波辐射过程并行设计与优化[D].北京：中国气象科学研究院，2012.
[10] 刘永柱，张林，金之雁.GRAPES 全球切线性和伴随模式的调优[J].应用气象学报，2017，28(1)：62-71.
[11] DENNIS J M，EDWARDS J，LOY R，et al.An application-level parallel I/O library for Earth system models[J].International Journal of High Performance Computing Applications，2012，26(1)：43-53.
[12] HURRELL J W，HOLLAND M M，GENT P R，et al.The community earth system model：a framework for collaborative research[J].Bulletin of the American Meteorological Society，2013，94(9)：1339-1360.
[13] REW R，DAVIS G.NetCDF：an interface for scientific data access[J].IEEE Computer Graphics and Applications，1990，10(4)：76-82.
[14] LI J，LIAO W，CHOUDHARY A，et al.Parallel netCDF：a high-performance scientific I/O interface[C]//SC′03：Proceedings of the 2003 ACM/IEEE Conference on Supercomputing.IEEE，2003：39.
[15] 季旭，武海平，邹寅隆，等.面向 LICOM2 的并行 I/O 优化[J].科研信息化技术与应用，2014(5)：37-48.
[16] LIU H，LIN P，YU Y，et al.LASG/IAP climate system ocean model version 2：LICOM2[M]//Flexible Global Ocean-Atmosphere-Land System Model.Berlin. Springer，2014：15-26.
[17] LOFSTEAD J F，KLASKY S，SCHWAN K，et al.Flexible IO and integration for scientific codes through the adaptable IO system(ADIOS)[C]//Proceedings of the 6th International Workshop on Challenges of Large Applications in Distributed Environments，2008：15-24.
[18] ZOU Y，XUE W，LIU S.A case study of large-scale parallel I/O analysis and optimization for numerical weather prediction system[J].Future Generation Computer Systems，2014，37：378-389.
[19] WANG W，HUANG X，FU H，et al.CFIO：a Fast I/O library for climate models[C]//2013 12th IEEE International Conference on Trust，Security and Privacy in Computing and Communications.IEEE，2013：911-918.
[20] 陈璟锟，杜云飞.地球科学大规模并行应用的重叠存储优化[J].计算机研究与发展，2019，56(4)：116-123.
[21] ACHARYA A，UYSAL M，BENNETT R，et al.Tuning the performance of I/O-intensive parallel applications[C]//Proceedings of the Fourth Workshop on I/O in Parallel and Distributed Systems：Part of the Federated Computing Research Conference，1996：15-27.
[22] DONE M，DAVIS A，WEISMAN L.The next generation：explicit forecasts of convection using weather research and forecasting(WRF) model[J].Atmospheric Science Letters，

(下转第 52 页)

表 3 优化前后的多线程批处理指令返回值情况

线程号	指令	隔离前	隔离后
线程 0	Get T1	N/A	N/A
	Get T2	N/A	N/A
	redisGetReply	Kv1，Tv2	Tv1，Tv2
线程 1	Get K1	N/A	N/A
	Get K2	N/A	N/A
	redisGetReply	Kv3，Kv4	Kv1，Kv2
线程 2	Get K3	N/A	N/A
	Get K4	N/A	N/A
	redisGetReply	Kv2，Tv1	Kv3，Kv4

每次实验结果几乎各不相同，这是由于指令和结果集的内存混淆所致。当使用本文设计的 Redis 客户端进行测试时，线程0、1、2 分别得到了 Tv1、Tv2、Kv1、Kv2、Kv3、Kv4 的返回值，与各自的指令提交顺序完全一致。

5 结论

以 Redis 为代表的内存数据库由于高性能、低延迟、非结构化存储等特性，已成为当今大数据时代的宠儿。如何用好这些数据库也成为了大家关注的重点。本文以 Redis 客户端中应用最为广泛的 Hiredis 为例开展了深入分析，发现其在高并发高性能应用中存在一些问题，包括：面向大型指令时的高内存开销问题，在复杂情景下的内存混淆问题等。为了解决这些问题，本文设计并实现了一个高性能高可用的 Redis 客户端，并且借助于真实的应用场景做了深入验证。

虽然本文的工作主要以 Beacon 为例展开，但是也具有通用性，同样也可推广到其他使用 Redis 的业务中去，例如常见的日志处理系统、电商交易平台以及社交文娱业务等。总之，在面向大数据的高性能处理情景时，本文的工作可保证系统的高性能、高并发、低开销。下一步计划将该客户端应用到其他基于 Redis 服务的高性能系统中去。

参考文献

[1] GYŐRÖDI C，GYŐRÖDI R，PECHERLE G，et al.A comparative study：MongoDB vs.MySQL[C]//2015 13th International Conference on Engineering of Modern Electric Systems (EMES).IEEE，2015：1-6.

[2] NAQVI S N Z，YFANTIDOU S，ZIM?NYI E.Time series databases and influxdb[J].Studienarbeit，Université Libre de Bruxelles，2017，12.

[3] DIVYA M S，GOYAL S K.ElasticSearch：an advanced and quick search technique to handle voluminous data[J].Compusoft，2013，2(6)：171.

[4] WU X，LONG X，WANG L.Optimizing event polling for network-intensive applications：a case study on redis[C]//2013 International Conference on Parallel and Distributed Systems.IEEE，2013：687-692.

[5] TANG W，LU Y，XIAO N，et al.Accelerating redis with RDMA over InfiniBand[C]//International Conference on Data Mining and Big Data.Springer，Cham，2017：472-483.

[6] MITCHELL C，GENG Y，LI J.Using one-sided RDMA reads to build a fast，CPU-effcient key-value store[C]//2013 USENIX Annual Technical Conference，2013：103-114.

[7] KALIA A，KAMINSKY M，ANDERSEN D G.Using RDMA efficiently for key-value services[C]//Proceedings of the 2014 ACM Conference on SIGCOMM，2014：295-306.

[8] WANG Y，MENG X，ZHANG L，et al.C-hint：an effective and reliable cache management for rdma-accelerated key-value stores[C]//Proceedings of the ACM Symposium on Cloud Computing，2014：1-13.

[9] LIU Q，YUAN H.A high performance memory key-value database based on Redis[J].J. Comput.，2019，14(3)：170-183.

[10] ZHANG J，JIA Y.Redis rehash optimization based on machine learning[C]//Journal of Physics：Conference Series.IOP Publishing，2020，1453(1)：012048.

[11] BLANKSTEIN A，SEN S，FREEDMAN M J.Hyperbolic caching：flexible caching for web applications[C]//2017 USENIX Annual Technical Conference，2017：499-511.

[12] PAN C，LUO Y，WANG X，et al.pRedis：Penalty and locality aware memory allocation in Redis[C]//Proceedings of the

(下转第 58 页)

(上接第 45 页)

2004(5)：110-117.

[23] SHCHEPETKIN A F，MCWILLIAMS J C.The regional oceanic modeling system(ROMS)：a split-explicit，free-surface，topography-following-coordinate oceanic model[J].Ocean Modelling，2005，9(4)：347-404.

[24] 廖嘉文，陈璟锟，颜辉，等.有限体积近岸海洋模式 FVCOM 的并行 I/O 优化[C]//CCF HPC China，2021：13-20.

[25] QI J，CHEN C，BEARDSLEY R C，et al.An unstructured-grid finite-volume surface wave model(FVCOM-SWAVE)：implementation，validations and applications[J].Ocean Modelling，2009，28(1)：153-166.

(收稿日期：2021-12-04)

作者简介：

杨斌(1992-)，男，博士研究生，工程师，主要研究方向：文件系统、智能存储。

王敬宇(1970-)，男，硕士，高级工程师，主要研究方向：高性能计算软件调试、大规模并行调试。

刘卫国(1975-)，通信作者，男，博士，教授，主要研究方向：高性能计算、大数据处理与分析，E-mail：weiguo.liu@sdu.edu.cn。

版权声明