GRAPES_CUACE 大气化学耦合模式并行优化

叶跃进[1] 陈德训[1] 胡江凯[2] 马 欣[2] 张小曳[3]
(江南计算技术研究所 江苏 无锡 214083)[1] (中国气象局数值预报中心 北京 100081)[2]
(中国气象科学研究院 北京 100081)[3]

摘 要 文中主要介绍了数值天气预报模式 GRAPES_MESO(4.0 版本)与大气化学模式 CUACE 在线耦合形成的 GRAPES_CUACE 大气化学耦合模型在不同版本的 x86 体系结构下的并行优化算法的研究与分析。借鉴目前国内外主流的并行优化设计方法,结合 GRAPES_MESO 系统本身的程序架构和并行框架,针对不同版本 x86 体系架构做了相应的并行化改造。运用 gprof 工具和戳桩计时等方法,测试得到的程序热点模块主要有 3 部分:IO、通信和物理过程。对 IO 模块主要的优化方法为:1)由离散读写改为连续读写;2)开辟缓冲区由稀疏访存改为连续访存;3)异步 IO。对通信部分采用两种方式:1)由细粒度改为粗粒度通信;2)采用时间复杂度更低的集合通信。对 IO 与通信模块优化结果分析可得:IO 模块优化后的耗时占比由原来的 43.7%降至 1.41%,比重大幅度降低,最优部分性能提升了 317 倍,因此,该方法极大地提升了 IO 模块运行效率。此外,对物理过程进行优化采用的主要方法是:1)多层循环计算过程由离散改为连续;2)通信机制循环外移;3)数据复用以减少计算冗余;4)缩减栈变量空间等。这些优化方法使计算性能提高了 22%,进一步提高了程序的并行效率和模式的强可扩展性。

关键词 异步 IO,粗粒度,连续访存,集合通信
中图法分类号 TP302.7 **文献标识码** A

Parallel Design and Optimization of GRAPES_CUACE On-line Coupled Air Quality Mode

YE Yue-jin[1] CHEN De-xun[1] HU Jiang-kai[2] MA Xin[2] ZHANG Xiao-ye[3]
(Jiangnan Institute of Computing Technology, Wuxi, Jiangsu 214083, China)[1]
(Numerical Weather Prediction Center of CMA, Beijing 100081, China)[2]
(Chinese Academy of Meteorological Sciences, Beijing 100081, China)[3]

Abstract This article mainly introduced the research and analysis of the parallel optimization algorithm of the meteorological particulate_meso dust aerosol coupling model under different versions of the x86 architecture. Drawing on the current mainstream parallel optimization design methods at home and abroad, combined with the GRAPES_MESO system's own program architecture and parallel framework, corresponding parallelization transformation was implemented for different versions of x86 architecture. Using the gprof tool and poke pile timing, the test hotspot module has three main parts: IO, communication and physical process. The main optimization methods for the IO module are: 1) continuous reading and writing by discrete reading and writing; 2) opening buffer from sparse memory access to continuous memory access; 3) asynchronous IO. The following methods are adopted for the communication part: 1) the fine-grained communication is changed from fine-grained to coarse-grained; 2) the aggregate communication with lower time complexity is adopted. Analysis of optimization results for IO and communication modules show that the time-consuming ratio of IO module optimization decreased from 43.7% to 1.41%. The proportion is greatly reduced, and the optimal performance is improved by 317 times. Therefore, the method described in this paper greatly improves the operating efficiency of the IO module. In addition, the main optimization methods used to optimize the physical process are as follows: 1) the multi-layer cyclic calculation process is changed from discrete to continuous; 2) the communication mechanism is cyclically shifted; 3) the data is reused to reduce computational redundancy; 4) the stack variable space is reduced. The computational performance is increased by 22%, which further improves the parallel efficiency of the program and the strong scalability of the model.

Keywords Asynchronous IO, Coarse-grained, Continuous memory access, Aggregate communicatio

---

本文受国家重点研发计划(2016YFC0203300),国家重大专项基金(2016YFA0602202,2017YFB0202603)资助。

叶跃进 男,硕士生,工程师,主要研究领域为并行算法与应用,E-mail:ye_ddr@foxmail.com;**胡江凯** 男,硕士,高级工程师,主要研究领域为气象数值预报系统,E-mail:hujk@cma.cn(通信作者)。

1　引言

GRAPES_CUACE 是在国内科学家自主研发的中尺度天气数值模式系统 GRAPES_MESO 和大气化学模式 CUACE(CMA Unified Atmospheric Chemistry Environmen)的基础上开发的在线耦合的气象-化学模式系统。

GRAPES 全球/区域同化预报系统是中国气象局组织且由科学家自主研发的新一代数值预报系统[1]。GRAPES 是全球与区域一体化模式，可作为 GRAPES_GFS 全球模式应用，也可作为 GRAPES_MESO 应用。其中，GRAPES_MESO 是 GRAPES 数值预报基本体系的重要组成部分，其可应用范围很广，从业务预报到理论、模拟研究，其中包括：理想试验(平衡流、密度流、地形重力波、变形流等 10 多种)、理想模拟(Large Eddy Simulation，GRAPES_LES 模式)、台风预报与模拟研究(GRAPES_TYM，GRAPES_TCM)、海-气耦合模拟研究(GRAPES_ECOM)等[2]。

CUACE 是由中国气象科学研究院研发的大气化学模式[3]，该模型包含了气溶胶的源、输送、干、湿沉降、云中和云下清除等详细过程，并计算了气溶胶和云的相互作用，以及沙尘气溶胶模块[3-5]气相化学机制采用 RAMD，气溶胶化学机制采用 CAM，化学热动力平衡采用 isorropia。该模式服务于国家气象中心环境气象预报业务，为雾霾预报提供多种产品[6]。

GRAPES_CUACE 模式被应用于多个研究。王宏等[7-8]基于 GRAPES_CUACE 在线耦合模式建立了新一代沙尘天气预报系统，并被应用于沙尘天气预报研究中。GRAPES_CUACE 也应用于污染减排控制和污染来源追溯[9]和污染物与气象要素间相互反馈研究中[10-12]。WANG[13]等利用该模式探究了东亚地区沙尘气溶胶的辐射强迫效应对长波辐射的影响，结果表明约 1/2 的地表负短波辐射和 1/3 的 TOA 负短波辐射被沙尘气溶胶的正长波辐射所抵消。

随着 GRAPES_MESO 模式和各模块的递进以及气象服务需求的日益增加，现有的 GRAPES_CUACE 版本已经升级到与 GRAPES_MESO 4.0 版本在线耦合状态，面临业务应用的切实需求，需要对耦合模块的并行计算性能进行优化，并将其纳入整体的编译体系中，适配 GRAPES_MESO 4.0 版本的并行框架。升级后的 GRAPES_CUACE 系统不仅能提供更适用的雾霾天气预报业务，同时使整体架构更加协调一致，有利于模式系统的可持续发展。

本文的主要研究目标是 GRAPES_CUACE 模式的并行优化。第 2 节先分析模式在不同版本的 X86 集群中性能的分布特点；第 3 节结合第 2 节性能分析特点，借鉴目前国内外主流的并行优化设计方法，结合 GRAPES_MESO 系统本身程序架构和并行框架，研究并实现系统的整合与优化；第 4 节对比分析实验结果；最后总结全文。

2　模式分析

研究 GRAPES_MESO v4.0 模式现有的程序结构与并行模式，将 CUACE 模块纳入 GRAPES_MESO 模式现有的并行框架，并在此基础上优化系统，以提高模式的并行效率。通过结合主流的并行优化方法与 GRAPES_MESO 系统程序结构，分别从以下几点对模式进行了分析。

2.1　IO 模块

由于新老机器在 IO 上性能有所差异，本文通过两种不同版本的 x86 集群对 GRAPES_CUACE 模式程序做并行分析与优化。两者在 IO 模块上的耗时占总运行时间的比例均超过 40%，因此缩短运行时间能很大程度的提高模式系统的运行效率，这也为模式的其他模块制定优化算法奠定了基础。以下是 x86 不同版本体系结构下 IO 部分函数耗时分布表(见表 1、表 2)，且测试数据均取单次运行平均值。

表 1　集群 IO 模块耗时分布表

曙光旧版 x86(2.16.1)				
程序模块	64 核心	128 核心	256 核心	512 核心
read_grapes_input_data(I)	28.37	26.74	30.12	33.34
read_tracer_data(I)	209.63	209.82	211.16	210.79
read_grapes_boundry_data(I)	0.62	0.73	1.12	1.19
post_output(O)	13.32	34.36	60.45	136.12
aero_output(O)	4.12	5.13	6.26	9.61
tracer_output(O)	109.12	113.45	126.75	145.36

表 2　集群 IO 模块耗时分布表

曙光新版 x86(3.10.0)				
程序模块	64 核心	128 核心	256 核心	512 核心
read_grapes_input_data(I)	279.63	280.30	280.97	281.38
read_tracer_data(I)	1799.21	1798.83	1796.94	1782.46
read_grapes_boundry_data(I)	11.79	12.21	12.39	13.61
post_output(O)	0.72	0.67	0.66	0.68
aero_output(O)	0.43	0.39	0.44	0.41
tracer_output(O)	22.78	22.45	21.56	21.96

GRAPES_MESO 模式由于本身的程序结构特性导致了它在读模块和写模块上性能差异较大。通过程序测试分析发现上一代 x86 集群环境下 IO 模块读操作的效率要明显高于写操作的效率，而新一代 x86 集群环境下写操作效率高于读操作。导致这种差异的因素有很多，可能是受到文件系统对内存读写调配不同的影响，或者是由于不同版本的编译器对读写优化能力与方式上存在差异，亦或是受到其他软硬环境各系统参数调配影响，包括：网络带宽、计算节点主频、L1 缓存组数、缓存线长度等。本文只针对程序测试结果做分析与优化而不对软硬环境差异做深入探讨。详细数据分析如图 1—图 6(横坐标表示运行进程数；纵坐标表示运行时间，单位为秒)所示。

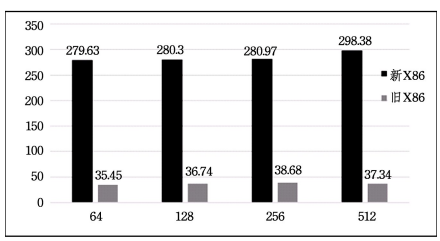

图 1　read_grapes_input_data

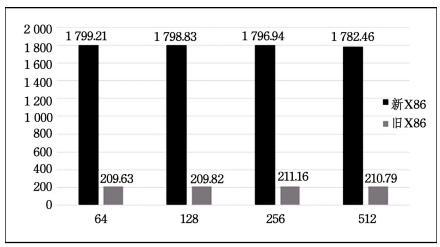

图 2　read_tracer_data

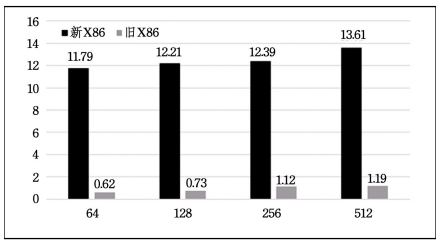

图 3 read_grapes_boundry_data

图 1—图 3 给出读操作在不同版本集群下的对比，而图 4—图 6 表示写操作的对比情况。显然，读操作在新版 x86 下性能极差，而写操作在旧版 x86 下性能差，运行时间随进程规模增大而增加。将分别从读操作（IO 输入模块）与写操作（IO 输出模块）详细分析模式 IO 的性能并设计相应的优化方案。

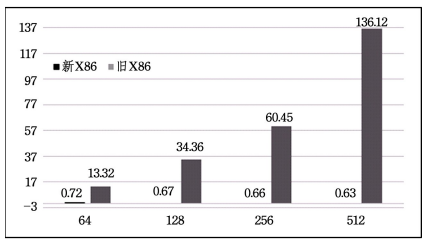

图 4 post_output

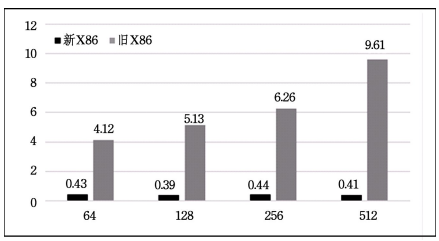

图 5 aero_output

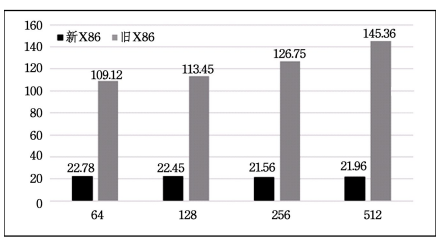

图 6 tracer_output

2.1.1 IO 输入

通过对模式性能的测试分析可得出，模式在新版 x86 集群上运行时间约达到 2000s，占 IO 总运行时长的 90%以上。因此在新版 x86 集群下，IO 输入部分是优化工作的重点。以下对 IO 输入部分的分析主要是以新版 x86 集群相关测试数据作为依赖。

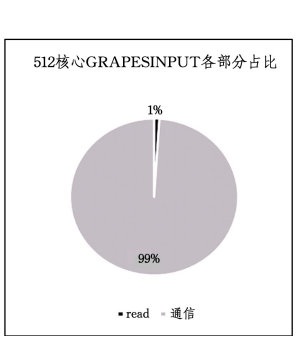

图 7 读取 grapesinput 各部分耗时占比

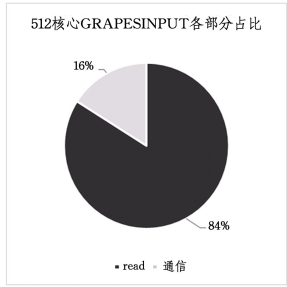

图 8 读 grapes 边界数据各部分耗时占比

模式的 IO 输入过程主要由两部分组成：1）主进程获取数据信息；2）主进程按任务将数据分块并依次分配给其余各进程。其中，前者为实际 IO 过程，后者是通信过程。通过测试对比模式中不同数据的读操作和通信过程的耗时情况，结论都是读数据过程的运行时间占比远高于通信过程，如图 7—图 8 所示。通过进一步测试分析，我们最终得到了可能导致性能变差两个因素。

1）read 采用跳读方式。该模式读取数据时采用稀疏的存储方法（见图 10）。其中 globbuf 数组存储方式在内存中连续性映射表现依次为：i 层是最小连续内存块，也是与 *cache line* 最直接相关的连续存储空间；其次是 k 层；最后是 J 层。若优先遍历 k 层循环，这使得系统访存时需不断重复跳读取数据方式体系结构的性能瓶颈，容易增加系统，如图 11 所示。因此，为了避免使用跳读方式，改用顺序读取方法，使随机访存转变为连续访存，这样能大大提高系统的访存性能，保证了程序运行效率。

```
Globbuf=0.
IF(OnMonitor)THEN
    do k=kms,kme
  READ(fid)((globbuf(i,k,j),i=ids,ide),j=jds,
jde)
  eddo
ENDIF
```

图 9 读 grapes 初始数据-read 过程原始 code

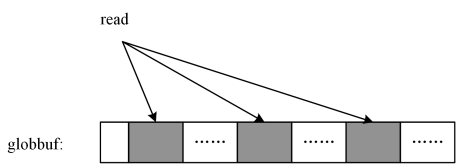

图 10 系统获取数据信息存入 globbuf 过程示意图

2）使用结构体成员变量参与读数据过程。系统在执行 read 操作时是按固定的字段长度来获取数据并将数据存储到物理内存块上。模式在获取边界信息时使用结构体的成员变量作为存储对象，如图 11 所示。虽然结构体数组 gird 访存顺序是按最连续的 i 层起依次访问，但由于成员变量在结构体的内存块中并不连续，依然有随机访存的特性，如图 12 所示。因此在读数据过程中应尽量避免使用结构体成员变量作为存储单元存储数据信息，而改用一块连续缓冲单元作为介质先获取数据再循环赋值给结构体成员变量。

```
IF(OnMonitor)THEN
  READ(fid)(((grid%q_b(i,k,j,1),i=1,&
    max(ide,jde)),k=kms,kme),j=1,grid%spec
_bdy_width)
ENDIF
```

图 11 读 grapes 边界数据-read 过程原始 code 图

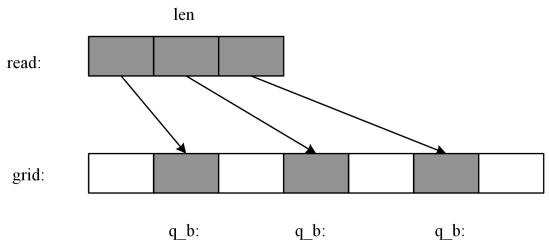

图 12　系统获取边界数据存入 grid 过程示意图

2.1.2　IO 输出

模式的 IO 输出过程是嵌入在计算过程中的，并由代理控制，在该模式 48 小时模拟预报实验中标准输出机制为每 30 min 出一次结果。本文通过对新旧版集群下 512 进程规模 IO 输出做对比实验，得知在旧版集群下输出一次结果需耗时 139 s，而在新版集群下输出一次结果平均仅需 0.5 s。因此在旧版 x86 集群下，对 IO 输出的优化显得尤为关键。（以下实验结果分析是基于旧版 x86 集群相关测试数据作为依赖）。

模式的 IO 输出过程主要由两部分组成：1）主进程获取其他进程数据并依次装入固定内存单元；2）由主进程写二进制输出文件。其中，前者为通信过程，后者为实际 IO。通过模式测试对比，我们发现参与测试进程规模越大，IO 过程耗时明显增加（如图 13－图 14 所示）。因此，导致通信耗时显著增加的两个因素为：

1）细粒度通信。使通信次数增多造成 MPI 通信机制反复启动增加系统额外开销。

2）阻塞的高时间复杂度通信——“点对点”式的全局通信方法是通过主任务进程循环接收其余各进程数据，主进程在接收其中一个进程的消息时，其他进程消息需排队等待，随着进程规模增加，需要等待的进程数增加，通信时间也增加。因此，需要增大通信粒度并改变通信算法以减少由上述两点造成的通信开销。此外，模式的写操作也是离散方式，写在一定程序上也能影响到程序的运行效率，在程序设计过程中应尽量避免直接的离散读写文件。

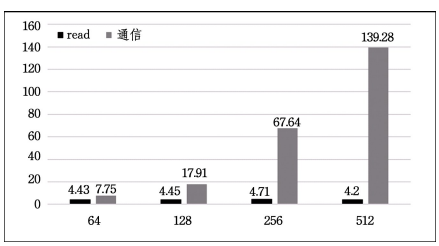

图 13　写 post 文件各部分不同进程规模耗时分布

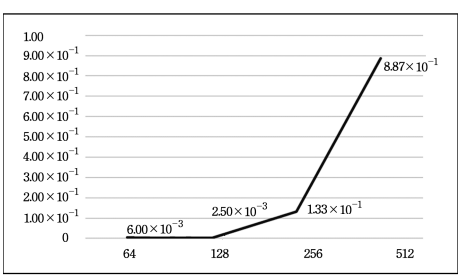

图 14　patch2global 通信函数不同进程规模耗时分布

2.2　计算过程

影响程序计算性能的原因有很多：计算访存不连续、循环计算中掺杂许多细粒度通信、计算冗余、栈空间冗余、循环空转、负载不均衡等。这些因素都能影响并行程序的运行效率，通过对模式程序进行测试分析发现，主要有以下几个方面影响模式运行效率。

1）多层循环计算访存不连续：多层循环计算时访存不连续容易导致 cache 命中率低，影响内存访问效率，导致整体运行时间增加（见图 15）。

```
do j=jts,jte
do k=kts,kte
do i=its,ite
   do b=1,num_band
   do nn=1,aero_nsize
   dtaual(i,k,j,b)=dtaual(i,k,j,b)+ntaual(i,k,j,nn,b)
asytau(i,k,j,b)=asytau(i,k,j,b)+ntaual(i,k,j,nn,b)&* monitor(i,k,j,
                b,nn)
   enddo
   enddo
enddo
enddo
enddo
```

图 15　访存不连续

2）循环迭代中含有细粒度 MPI 通信：例如在 x，y 方向动力传输过程中（算法结构如图 16 所示）。最外层循环内有通信函数 reg_updatehalo（见图 17）必然导致通信次数增加，从而增加许多额外开销。因此需要改进通信机制，以避免由于频繁通信造成运行时间增加。

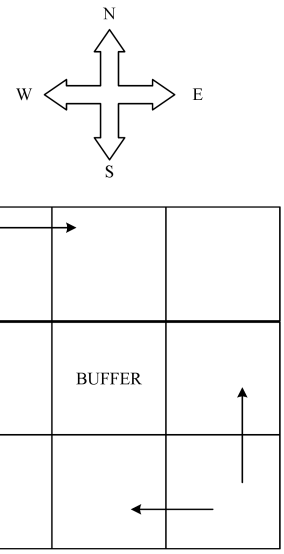

图 16　reg_updatehalo 数据走向图

```
DO kk=2,num_tracer
   do j=…
      do…
   enddo !j
      …
reg_updatehalo(varlist)
      …
   do j=jts,jte
      do…
         …
      enddo !j
ENDDO
```

图 17　多层循环内嵌通信

3)计算冗余:核心函数计算过程中,对不同物理量求值存在相同的赋值操作。重复操作导致系统指令流冗余不仅影响计算结果输出效率也使得程序模块化受制约。因此,需要把相同的赋值操作存入缓冲区,保证在计算过程中数据能复用,以提高 cache 空间的利用率。

4)调用函数传入参数栈空间冗余:模式在计算过程中的函数调用存在传入参数值空间冗余。例如在调用 CAMAE-ARSOL 函数是传入参数 QV3D 时,实际参与计算的数组边界是:(its:ite,kts:kte,jts:jte)。而变量传入时维度大小是:(ims:ime,*kms*:*kme*,jms:jme)。因此,其计算边界小于维度边界,缩减了传入参数的内存开销,能提高栈空间利用率,这会对程序运行效率提升有所帮助(见图 18、图 19)。

```
REAL, DIMENSION(ims:ime, kms:kme, jms:jme),&.
      INTENT(IN)::                                    P3D,&.
                                                      P8W3D,&.
                                                      Pi3D,&.
                                                      T3D,&.
                                                      QV3D,&.
                                                      QC3D,&.
                                    dz8w,U3D,V3D,W3D,RHO3D,CLDFRA3D
```

图 18 调用函数参数数据冗余

```
j_loop:DO J=jts,jte
  DO K=kts,kte
  DO I=its,ite
    ii=I- its+1
      NK=kte+1-k
      T2D(ii,k)=T3D(I,NK,J)
        fdz(ii,k)=dz8w(i,nk,j)
        SH2D(ii,k)=QV3D(I,NK,J)/(1.+QV3D(I,NK,J))
        SH2D(ii,k)=max(⊙.,SH2D(I,NK,J))
        QC2D(ii,k)=QC3D(I,NK,J)
        QC2D(ii,k)=max(⊙.,QC2D(II,K)
                                                  ! P2D is
        P2D(ii,k)=P3D(I,NK,J)
        fcld2D(ii,k)=CLDFRA3D(I,NK,J)
```

图 19 调用函数参数数据冗余

3 优化方法

3.1 IO 模块

3.1.1 IO 输入

(1)read 由跳读改为顺序读取:读取数据时由离散存储改为连续存储,从最连续层开始依次读数据。先开辟一段连续缓冲区,通过缓冲区按内存连续原则依次从最连续层 i 层开始存储数据,完成后再将缓冲区数据移动至指定内存空间。

(2)read 读取数据时由结构体类型变量改为内存块连续的临时变量:使用结构体类型变量作为 read 数据的存储对象时,其本质是一种离散访存机制,而且对于新版的 x86 体系结构这种读文件方式会导致性能降低。因此,需要先开辟一段连续的临时缓冲区,将 read 数据先存入缓冲区,再将缓冲区中的数据 move 到 grid 结构对应的成员。

3.1.2 IO 输出

(1)由细粒度改为粗粒度通信:将需要输出到文件的变量扩维处理,目的就是增加单次通信量;将 patch2global 通信机制循环外移,使原有数组在内层三维循环通信扩大至四维一次通信,这样不仅扩大了通信粒度,同时也减少了通信次数,并在完成改造后验证数据一致性,确保数据的正确性。

(2)由"点对点"通信改为集合通信:patch2global 通信机制原有算法是各进程数据按顺序依次发送给主进程并由主进程循环接收,其时间复杂度为 O(n)。这种方式会产生进程排队现象,通信时间会随着进程规模的增加而增加。因此,采用 MPI_Gatherv 集合通信方式将会避免此现象,其通信原理是各进程直接将消息同步发送到主进程中指定的内存块上,其时间复杂度为 O(1)。这样能够大大减少通信时间,提高程序的运行效率。通信方式如图 20 所示。

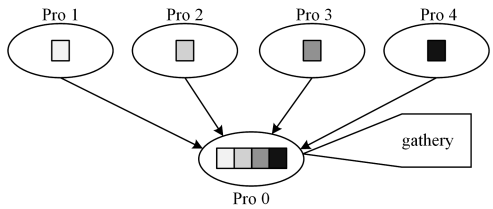

图 20 集合通信原理

(3)write 方式由稀疏改为连续:开辟连续缓冲区,对矩阵做转置处理,写文件时从最连续层开始依次循环遍历。

(4)使用异步 IO 方式:由于该模式原所有进程均参与计算过程,无法单独分离出 IO 进程组,因此需要修改原有进程划分算法。使用异步 IO 机制调整原有进程的任务分块,将所有进程分为 IO 进程组与计算进程组,由 IO 进程组负责模式 IO 输出过程。这样提高了计算和 IO 的重叠率,最大程度降低了 IO 耗时占比。具体算法如图 21 所示。

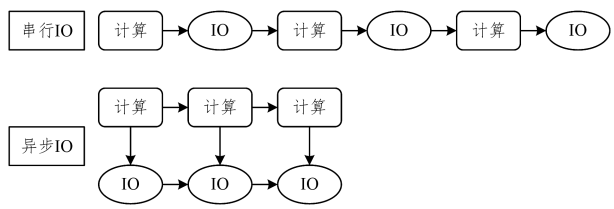

图 21 读边界数据优化前后耗时对比

3.2 计算模块

(1)调整循环层顺序:按数组在内存中连续原则调整循环层次,以最连续层为起始依次调整至最不连续层,以保证数组访存连续来提高 cache 的命中率。

(2)通信机制循环外移:调整循环中数组的通信粒度,并由细粒度改为粗粒度,接着将通信机制移至循环最外层外,尽力使循环内部减少通信机制使用率,完成优化后再还原原始数据。

(3)数据复用以减少计算冗余:在积分求解过程中,不同物理量的求值过程存在相同的求解过程,将其简化。把相同的赋值操作存入一块临时缓冲区中,以便下次直接使用,无需再求解。保证数据复用,提高 cache 空间的利用率。

(4)缩减调用函数栈变量空间:传入参数实际使用空间小于栈空间,会造成栈空间冗余。在函数调用传参时只传入实际参与的计算量空间。这样不仅提高了栈空间的利用率,也避免了数组在原函数计算过程访存不连续。例如,在 CAMAEROSOL 函数调用时,将最外层数据原有传入为:(jme:jme),缩小至维度为:(jts:jte),以减小内存开销从而提高栈空间使用率,如图 22 所示。

```
CALL CAMAEROSOL(itimestep,dt,                                    &.
        GSW,XLAT,XLONG,                                          &.
        ALBEDO,t_phy,                                            &.
        moist(ims,kms,jms,P_QV),                                 &.
        moist(ims,kms,jms,P_QC),                                 &.
        p_phy,p8w,dz8w,pi_phy,CLDFRA,                            &.
        u_phy,v_phy,w,rho_phy,                                   &.
```

↓

```
CALL CAMAEROSOL(itimestep,dt,                                    &.
        GSW(its:ite,jts:jte),                                    &.
        XLAT(its:ite,jts:jte),                                   &.
        XLONG(its:ite,jts:jte),                                  &.
        ALBEDO(its:ite,jts:jte),                                 &.
        t_phy(its:ite,kts:kte,jts:jte),                          &.
        moist(its:ite,kts:kte+1,jts:jte,P_QV),                   &.
        moist(its:ite,kts:kte+1,jts:jte,P_QC),                   &.
        p_phy(its:ite,kts:kte,jts:jte),                          &.
        p8w(its:ite,kts:kte,jts:jte),                            &.
```

图 22　修改调用函数栈空间

4　实验结果与分析

4.1　IO 模块优化效果

(1)IO 输入优化效果：实验是基于模式 48 小时模拟预报获取优化前后的性能对比图。从图 23、图 24 可以看出，优化前后性能差异很大，优化效果极好。其中，在读 grapesinput 模块时 512 进程规模下的运行时间由 282.38 s 缩短至 2.81 s，性能提升了 100 倍；在读 tracer 模块优化后 512 进程规模运行时间由 1782.46 s 缩短至 5.62 s，性能提升了 317 倍。由此可以看出，程序 IO 在读写顺序上与存储空间连续性上的要求非常苛刻，通过修改读文件顺序与存储顺序后性能得到极大的提高。

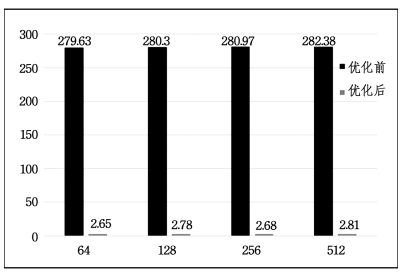

图 23　读 grapesinput 数据优化前后耗时对比

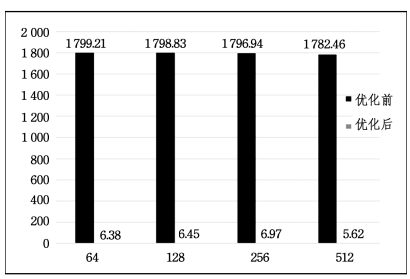

图 24　读 tracer 数据优化前后耗时对比

(2)IO 输出优化效果：IO 输出模块在不同版本 x86 集群上性能差异较大。实验结果表明：在新版 x86 平台优化后总耗时为 1.2 s 左右，而优化前为 0.5 s，这是由于新平台在做异步 IO 过程中计算进程组发送消息至 IO 进程组时涉及数据索引号“点对点”传输(计算进程需要告诉 IO 进程关于进程数据内存大小以便 IO 进程定制性分配)，临时缓冲区数据拷贝(为确保原有数据结构不受破坏)等额外开销，导致优化后反而比优化前更慢。但在旧版 x86 平台上 512 进程规模优化前 IO 输出一次耗时 145.73 s，优化后为 2.35 s，IO 输出效率提升了 56 倍，优化效果明显，且会随着进程规模的增大而凸显出来，如图 25 所示。

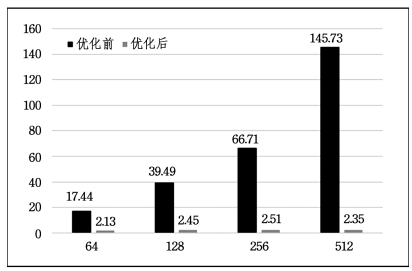

图 25　IO 输出优化前后耗时对比

(3)IO 整体优化效果：IO 整体优化效果如图表所示，从图 26、图 27 可以看出，原始 IO 模块运行时间占总运行时间的43.7%，经优化后降到 1.41%，512 进程规模总时间由原来的4862 s 降至 2789 s，由此说来通过本文在 IO 模块中所采用的优化方法使得其性能得到极大提升。

表 3　模式预报 48 小时优化前后 IO 耗时分布

(单位：s)

进程规模	优化前	优化后
64	16251.85	14110.91
128	10001.15	7982.74
256	6536.94	4534.64
512	4862.78	2789.63

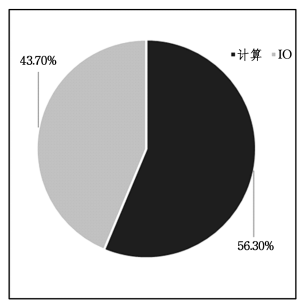

图 26　优化前 IO 与计算占比

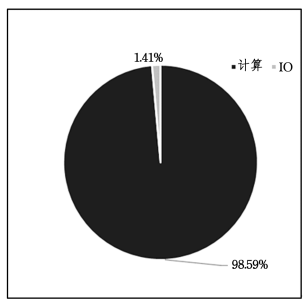

图 27　优化后 IO 与计算占比

4.2　计算过程优化效果

从表 4 中可以看出，512 进程规模下平均一次叠代步耗时从 1.29 s 降至 1.01 s，性能只提升 22%。这是由于该模式计算热点零散，分布到主要核心循环段时耗时小，导致了性能提升受限。因此，未来将会针对这部分在程序结构和算法上展开深入研究，并制定出优化方案。

表 4 计算过程单次叠代步优化前后耗时分布

(单位:s)

进程规模	优化前	优化后
64	6.72	6.68
128	3.74	3.51
256	2.15	1.89
512	1.29	1.01

4.3 可扩展性分析

影响程序可扩展性的因素有很多,从图 27 可以看出,在 512 进程规模时模式整体加速比由原来的 3.34 上升到5.78,加速比提升明显,并且在不同进程规模下整体加速比趋于线性。由于模式在 IO 模块上的优化使得 IO 占比非常小,主要运行时间在于多进程并行的计算任务上,保证各进程在计算任务上能够达到或者接近负载均衡,促进了进程之间任务流水的同步,减缓通信过程中的延时,进而提高程序的可扩展性;另一方面对模式的通信机制采用集合操作能够大大降低时间复杂度,并且优化效果会随着进程规模的增大而凸显,这使得模式的强可扩展性得到进一步提升。

表 5 优化前后总耗时分布

(单位:s)

进程规模	优化前	优化后
64	16251.85	13827.07
128	10001.15	7680.53
256	6536.94	4241.64
512	4862.78	2390.71

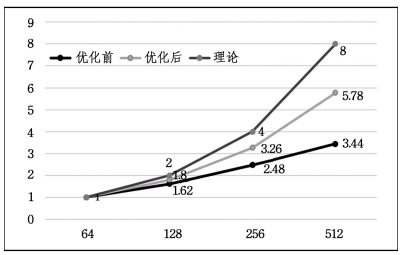

图 28 优化前后总加速比对比

结束语 本文针对不同版本 x86 体系架构做相应的并行优化改造,并结合程序结构采用多种并行优化手段分别对 IO 模块和计算模块做优化处理,对 IO 模块主要的优化手段是:1)读写次序由离散改连续;2)追求访存连续性原则;3)追求隐式 IO 原则。优化后的 IO 占比大幅度降低,这大大提高了程序性能。既对通信部分做了相应优化,缩短了通信次数与通信时间,也对计算过程的结构做了调优处理,包括多层循环计算过程由离散改连续,消除了计算冗余并缩减了栈变量空间等,进一步提高了并行效率。由于本文对程序计算过程尚处在调优阶段,并未深入研究。后续将从分任务调度、超线程优化、L1 缓存优化、缓存分块等方面对计算模块中的核心循环段以及高热点的矩阵计算部分做针对性测试分析与优化。

参 考 文 献

[1] 陈德辉,薛纪善,杨学胜,等. GRAPES 新一代全球/区域尺度统一数值预报模式总体设计研究[J]. 科学通报,2008,53:2396-2407.

[2] 张涵斌,陈静,等. GRAPES 区域集合预报应用研究[J]. 气象期刊,2014,40(9).

[3] GONG S L,ZHANG X Y. CUACE/Dust — An integrated system of observation and modeling systems for operational dust forecasting in Asia[J]. Atomospheric Chemisty and Physics,2008,8:2333-2340.

[4] ZHANG X Y. Characterization of soil dust aerosol in China and its transport/distribution during 2001 ACE-Asia. 1. Network Observations [J]. Journal of Geophysical Research, 2003, 108(9):4261.

[5] ZHOU C H,GONG S L,ZHANG X Y,et al. Development and evaluation of an operational SDS forecasting system for East Asia: CUACE/Dust [J]. Atomospheric Chemisty and Physics, 2008,8(4):787-798.

[6] 李曼,张载勇,李淑娟,等. CUACE 系统在乌鲁木齐空气质量预报中的效果检验[J]. 沙漠与绿洲气象,2014,8(5):63-68.

[7] WANG H,GONG S L,ZHANG H L,et al. A new-generation sand and dust storm forecasting system GRAPES_CUACE/Dust:Model development,verification and numerical simulation [J]. Sci Bull.

[8] 王宏,龚山陵,张红亮,等. 新一代沙尘天气预报系统 GRAPES_CUACE/Dust:模式建立、检验和数值模拟[J]. 科学通报,2009,54:3878-3891.

[9] AN X Q,ZHAI S X,JIN M,et al. Tracking influential haze source areas in North China using an adjoint model,GRAPES-CUACE [J]. Geosci. Model Dev. Discuss. ,2015,8:7313-7345.

[10] WANG H,SHI G Y,LI W,et al. The impacts of optical properties onradiative forcing due to dust aerosol[J]. Adv Atmos Sci , 2006,23:431-441.

[11] WANG H,ZHANG X Y,GONG S L,et al. Radiative feedback of dust aerosolson the East Asian dust storms[J]. J Geophys Res,2010,115:D23214.

[12] WANG H,ZHAO T L,ZHANG X Y,et al. Dust direct radiative affects onthe earth-atmosphere system over East Asia: Early spring cooling and late spring warming [J]. Chinese Science Bull,2011,56:1020-1030.

[13] WANG H,SHI G Y,ZHU J,et al. Case study of longwave contribution to dust radiative effects over East Asia [J]. Chinese Science Bull,2013,58:3673-3681.