基于 OpenACC 的高性能计算并行优化研究与应用

顾文静,孙 晨,王 彬
(国家气象信息中心 高性能计算室,北京 100081)

摘 要:针对 GPU 加速时存在的编码复杂性、移植性差导致开发维护效率低下的缺陷,利用基于 OpenACC 指导命令的加速技术对传统的串行代码进行改写,从而达到提高开发效率、简化代码的目的。以 GRAPES 全球模式长波辐射过程为研究对象,首先通过编译选项对程序性能进行初步优化,再根据其数据依赖和访存特性,对数据和循环结构进行预处理并添加 OpenACC 指导命令实现循环级并行。实验结果表明,长波辐射过程并行计算结果正确,在不改变原有代码结构的基础上即可获得 4 ~6 倍的加速比,优化性能可比拟相同计算能力的 Intel 集群,虽然较 GPU 加速仍有差距,但大大增强了代码的可读性和可移植性,且随着编译器和硬件技术的发展,OpenACC 有着广阔的发展空间。

关键词:神威·太湖之光;OpenACC;GRAPES 模式;长波辐射过程
中图分类号:TP301 **文献标识码**:A **文章编号**:1673-629X(2018)04-0065-06
doi:10.3969/j.issn.1673-629X.2018.04.014

Research and Application of Parallel Optimization in High Performance Computing Based on OpenACC

GU Wen-jing, SUN Chen, WANG Bin
(High Performance Computing Division, National Meteorological Information Center, Beijing 100081, China)

Abstract: For the inefficiency of development and maintenance caused by complex coding and poor portability in GPU acceleration, we make use of the acceleration technology based on the OpenACC to rewrite the traditional serial code for improving the development efficiency and simplifying the code. In this paper, taking the long wave radiation in GRAPES model as research object, the preliminary optimization of procedure performance is carried on by compiler options first, and then the data and loop structure is preprocessed with adding OpenACC instruction to implement the parallel of loop according to the data dependence and memory accessing feature. The experiments indicate that the parallel computing of long wave radiation is correct with the acceleration of 4 to 6 times on basis of the original non-parallel code structure. The optimal performance can be compared to the Inter cluster in same computing power. Although still lower than GPU acceleration, the readability and portability of the code are greatly enhanced. With the development of the compiler and hardware technology, the OpenACC has a broad space for development.

Key words: Sunway Tauhu Light System; OpenACC; GRAPES model; long wave radiation

0 引 言

近年来,面对浩如烟海的数据,CPU 已力不从心,并行计算技术日趋成熟,世界领先的超级计算机都装备大量的 GPU 加速器或众核处理器[1]。然而,要在 GPU 硬件上实现加速需要通过对底层的 API 进行编程来实现,程序编写复杂、难度大,且容易形成高度依赖特定设备的代码。因此,一种基于指令制导计算的编程模型 OpenACC 应运而生[2]。OpenACC 的编程机制是在源程序中添加少量编译标识,编译器根据作者的意图自动产生低级语言代码,无需学习新的编程语言和加速器硬件知识。只需添加少量编译标识,不破坏原码,开发容易,既可并行执行又可恢复串行执行;硬件更新时,不需要修改代码,重新编译即可。OpenACC 标准制定时就考虑了当前及将来多种加速器产品,同一份代码可以在多种加速器设备上编译、运行、无成本切换硬件平台[1]。

GRAPES(global/regional assimilation and prediction system)是中国气象局自主研发的静力/非静力多尺度通用数值预报系统,该系统是气象与气候研究的基础和核心。对于一个大型、先进的数值预报系统而

收稿日期:2017-05-04 **修回日期**:2017-09-08 **网络出版时间**:2017-12-05
基金项目:国家重点研发计划项目(2016YFA0602102);公益性行业专项(气象)科研专项(GYHY201306062);中国气象局局校合作项目
作者简介:顾文静(1984-),女,硕士,工程师,研究方向为高性能计算机应用、系统开发。

言,并行计算已成为其必需的结构特征[3]。

GRAPES 系统的核心部分包括模式动力框架和物理过程,动力框架中计算时间最长的为半拉格朗日计算和求解 Helmholtz 大型线性代数方程组,物理过程中耗时较大的为辐射和微物理过程,其中辐射过程计算耗时占到 GRAPES 模式的 40%,所以提升辐射过程的性能和计算效率,对整个 GRAPES 模式的性能提升具有重要科研价值和实际意义[4-5]。

以 GRAPES 模式物理过程中的长波辐射为加速研究对象,依托国家超级计算无锡中心的"神威·太湖之光"高性能计算机系统,设计一种基于 OpenACC 的加速算法,以实现利用更精简的指令来优化代码。

1 "神威·太湖之光"概况

2016 年 6 月 20 日,新一期全球超级计算机 500 强榜单在德国法兰克福举办的国际超算大会(ISC)上公布,"神威·太湖之光"超级计算机荣登榜首。"神威·太湖之光"由国家并行计算机工程技术研究中心研制,是全球首台运行速度超过 10 亿亿次/秒的超级计算机,峰值性能高达 12.54 亿亿次/秒,持续性能达到 9.3 亿亿次/秒。该系统部署在国家超级计算无锡中心,系统由 40 个运算机柜和 8 个网络机柜组成。打开柜门,4 个由 32 个节点板组成的超节点分布其中,每个节点板上包含 4 个节点卡,每个节点卡由两个装有"申威 26010"众核处理器和 32 GB 的 DDR3 内存节点组成,一台机柜就有 1 024 块处理器。节点之间通过基于 PCI-E3.0(外设部件互联标准扩展 3.0 版)的神威高速网络进行互联。

"神威·太湖之光"系统全部采用"申威 26010"众核处理器,架构如图 1 所示。

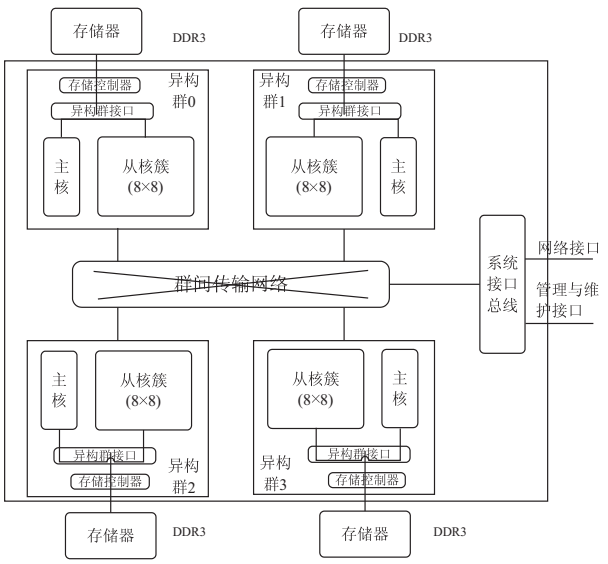

图 1 "申威 26010"处理器架构

该芯片主频 1.45 GHz,由 4 个核组组成,每个核组包含一个主核和一个从核簇,每个从核簇由 64 个从核组成,共 260 个处理器核心。同一个核组内的主核和 64 个从核共享主存,每个从核有独立的、容量为 64 KB 的高速缓存(SPM),支持 DMA(direct memory access)的方式在主存和 SPM 之间传输数据。主核作为处理器核心,可以进行通信、IO、计算等操作,同一核组内的 64 个从核作为加速计算部件,用来加速主核代码中的计算密集部分。

"神威·太湖之光"的冷却系统采用计算节点板上全封闭式循环水冷技术和定制化的液体水冷单元,功耗比达到每瓦特 60.51 亿次运算。

2 OpenACC * 编程模型

OpenACC 是由 OpenACC 组织(www.openacc-standard.org)于 2011 年推出的众核加速编程语言,以编译指示的方式提供众核编程所需的语言功能,其主要目的是降低众核编程的难度。用于 C/C++和 Fortran 的 OpenACC API 和指令负责把底层的 GPU 任务交给编译的同时,又提供跨系统、跨主机 CPU 和加速设备的支持[6]。

"神威·太湖之光"计算机系统中的 OpenACC * 语言是在 OpenACC2.0 文本的基础上,针对申威 26010 众核处理器结构特点进行适当的精简和扩充而来的。

2.1 执行模型

OpenACC * 程序的执行模型是在 host(主处理器)的指导下,host 和 device(加速设备)协作的加速执行模型,如图 2 所示。

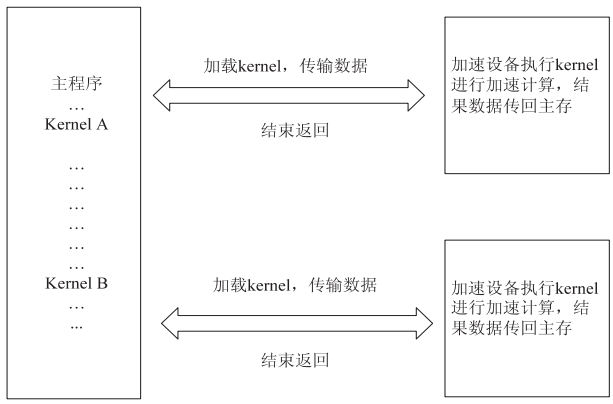

图 2 OpenACC * 执行模型

程序首先在 host 上启动运行,以一个主线程串行执行,或者通过使用 OpenMP 或 MPI 等编程接口以多个主线程并行执行,计算密集区域则在主线程的控制下作为 kernel(加速任务)被加载到加速设备上执行。kernel 的执行过程包括:在设备内存上分配所需数据空间;加载 kernel 代码(包含 kernel 参数)至 device;kernel 将所需数据从主存传输至设备内存;等待数据

传输完成;device 进行计算并将结果传送回主存;释放设备上的数据空间等。大多数情况下,host 可以加载一系列 kernels,并在加速设备上逐个执行[1]。

OpenACC * 支持三级并行机制:gang、worker、vector。gang 是粗粒度并行,在加速设备上可以启动一定数量的 gang。worker 是细粒度并行,每个 gang 内包含有一定数量的 worker。vector 并行是在 worker 内通过 SIMD 或向量操作的指令级并行[1]。

在"申威 26010"中,一个运算控制核心(简称主核)仅控制一个运算核心阵列(加速设备,简称从核阵列)的运行,每个运算核心阵列内有 64 个运算核心(简称从核),每个运算核心可以运行一个加速线程。默认情况下,64 个加速线程被组织成 64 个 gang、每个 gang 内一个 worker、worker 内可以 vector 并行的逻辑视图。通常情况下,尽量用满 64 个加速线程可以获得较好的运行性能。

2.2　存储模型

在 CUDA 和 OpenCL 等低层次加速器编程语言中,主机和加速器存储器分离的概念非常明确,内存间移动数据的语句占据用户大部分代码[7]。

"申威 26010"中从核和主核共享内存,从核可直接访问主存空间,并在从核内提供加速线程私有的高速缓冲(local data memory,LDM),加速计算需要存放到 LDM 的数据由从核控制传输。存储模型如图 3 所示。

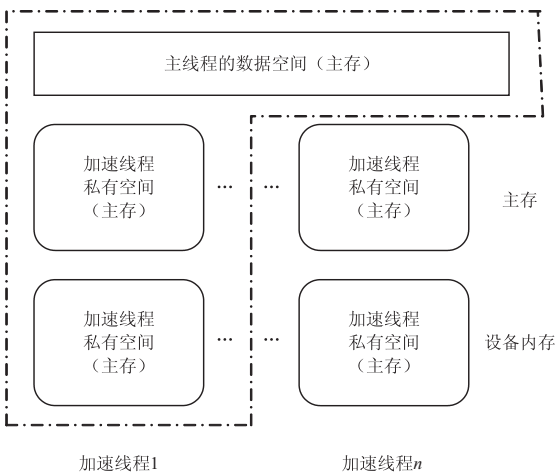

图 3　OpenACC * 存储模型

OpenACC 标准中的数据管理功能将不产生实际的作用,但直接访问主存往往带来性能损失,因此需要充分利用从核中的高速缓冲 SPM,提升数据访问效率。在"神威 · 太湖之光"中,OpenACC * 对 OpenACC 标准所做的主要功能延伸和语法扩展就是为了解决共享内存架构下片内高速存储空间的使用问题。

"申威 26010"中,从一个加速线程的角度来看,其可见的数据空间有三种。

(1)主线程数据空间:位于主存,主线程的内存空间对其创建的加速线程直接可见,加速线程可以直接访问相应的数据;对于主线程创建的多个加速线程而言,这部分空间是共享的;程序中在加速区外定义的变量,均位于该空间内。

(2)加速线程私有空间:位于主存,每个加速线程有独立的私有空间,程序中使用 private、firstprivate 子句修饰的变量将存放于该空间内。

(3)加速线程本地空间:位于设备内存,每个加速线程有独立的本地空间(LDM),本地空间的访问性能是三种空间中最高的。程序中使用 local、copy 等数据子句修饰的变量将由编译系统控制全部或局部存放于 LDM 内。主存与本地空间的数据交互由加速线程控制。

3　长波辐射过程 OpenACC 优化方案

GRAPES 模式的辐射模型采用欧洲中期天气预报中心(ECMWF)的长短波辐射方案,该方案将整个大气层划分成水平方向和垂直方向的三维网格,水平方向为横跨地球表面的二维网格,垂直方向则表示大气的层数[8-9]。数值天气预报模式是一种非线性化离散计算模式,其计算量巨大[10],最初的都是通过 CPU 计算实现,GPU 出现之后,虽然可以把该算法移植到 GPU 上并行执行,但编程难度大且易出错。因此,文中设计一种基于 OPENACC 加速方法,以实现通过简化的编程算法和简洁的 Fortran 语言代码完成对长波辐射过程并行处理过程。

3.1　辐射过程的构成

辐射过程是 GRAPES 系统中一个重要的物理过程,包含云结构的描述和辐射算法(RRTM 模块)两部分。其中云对调解辐射平衡起着至关重要的作用[11],该系统在云产生器的基础上利用 McICA 进行辐射计算。辐射过程首先调用 mcica_subcol_lw 函数实现云结构的描述,之后调用 RRTM 模块进行辐射计算。RRTM 模块初始化函数 rrtmg_lw_init 读取数据文件中的输入数据;然后调用 rrtmg_lw_rad 函数计算辐射传输的过程;最后调用辐射模型子函数 rrtm_lw,rrtm_lw 子函数一次计算一个垂直列的辐射传输值[9]。它包含以下几个基本步骤:inatm:准备大气剖面;cldprmc:计算云层光学深度;setcoef:计算这种大气剖面下辐射传输算法所需的各种量;taumol:计算气体光学深度和普朗克分数;rtrnmc:计算晴空和云层的辐射传输值。其中,taumol 中的 16 个子函数 taugb,计算了全部 16 个长波波域谱带的气态光学厚度和普朗克(Planck)函数。

辐射过程的函数包结构如图 4 所示。

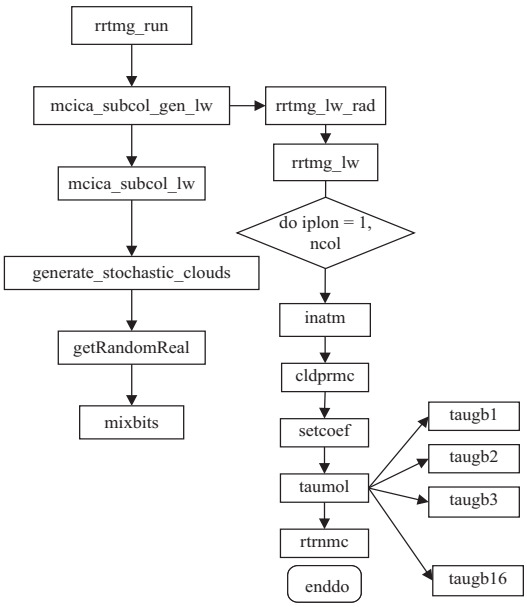

图 4 长波辐射过程的函数包结构

3.2 并行优化方案

并行化的唯一目的是充分利用硬件资源来提高程序运行速度,缩短运行时间。程序中最耗时间的是循环,计算并行化的目标是将循环迭代步分散到多个不同的线程上执行,这些线程运行在多个加速器核心,从而将计算任务由 CPU 转移到加速器上,减轻 CPU 的负担[1]。

结合程序代码的分析,确定需要众核加速的代码段,并根据源代码数据结构进行预处理,对相关数组和循环进行调整,以适合 OpenACC 加速。优化的基本思想是确定将程序中的标量和关键数组尽可能多地放到局存(LDM)中,并设法提高程序中数据的传输效率,将需要使用众核加速的循环的前后使用 OpenACC 加速编译指示进行标注,其中 PARALLEL 表明该部分代码是需要加速执行的并行代码;LOOP 表示下面紧跟着的循环需要在加速线程间进行并行划分。

3.2.1 RRTM 模块优化方案

rrtm_lw 函数包含一层循环,涵盖 inatm、cldprmc、setcoef、taumol 和 rtrnmc 五个子函数调用,每个子函数内部有多个 do 循环分块,考虑最外层的 iplon 循环阈值为[1,90],直接进行众核并行,从核使用率仅为 41%,没有充分发挥核组计算能力。且函数间调用关系复杂,如果在该层循环外层添加 OpenACC 加速指示语句,循环内部的所有函数将全部被调配到从核运行,即使添加 routine 函数将部分从核函数的栈数组放到主存,但由于子函数中变量众多,对从核空间需求较高,最终仍难避免栈的容量超出局存(LDM),造成模式运行出错。结合辐射过程的计算模型为单柱计算模型,核心迭代针对每个柱进行多个物理过程计算,分析到各子函数间存在数据依赖关系,为保证数据正确性且增加从核运算并行度,考虑对 rrtmg_lw 函数进行拆分并进行循环内移。内移工作从 rrtmg_lw 函数调用的函数 inatm 开始,依次对 cldprmc、setcoef、taumol 和 rtrnmc 子函数进行循环内移,代码结构示意如图 5 所示。

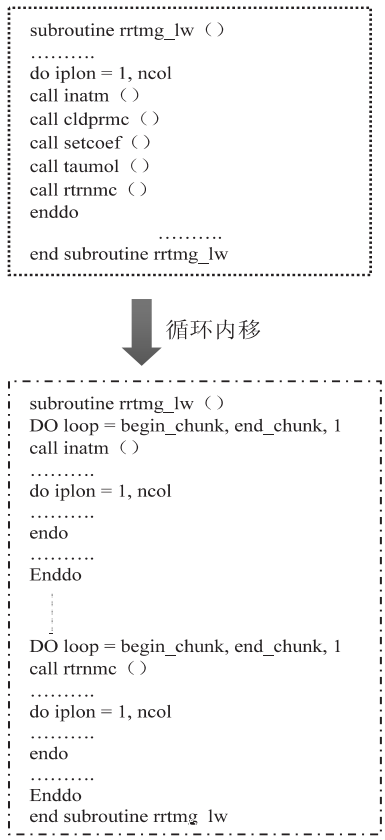

图 5 循环内移代码结构示意图

3.2.2 mcica_subcol_gen_lw 函数优化方案

经测试分析,函数中各进程负载不均衡,通过插桩确定耗时主要集中在两个分块,一块是随机数生成、另一块是数组计算。随机数部分调配到从核运行,并行生成 0~1 之间的随机数,效率大幅提升。数组计算部分包含 6 个循环,考虑到各循环结构和各层阈值均不一致,且没有紧嵌套关系,不适合进行循环合并,考虑直接添加 OpenACC 加速指示语句。下面取其中三个做实例分析。

(1)标量、数组拷贝。

```
do nm=1, nlay
do km=1, ncol
reicmcl(km,nm)=rei(km,nm)
relqmcl(km,nm)=rel(km,nm)
pmid(km,nm)=play(km,nm)*1.e2_rb
enddo
enddo
```

本段核心段代码中循环体结构为二维循环,外层循环阈值为[1,61],内层循环阈值为[1,90],计算数组大小,LDM 空间可以满足,考虑直接进行众核加速,

OpenACC 编译指示语句如下:

```
! $ ACC PARALLEL LOOP local(nm,km)
! $ ACC copyin(rei,rel,play) copyout(reicmcl,relqmcl,pmid)
! $ ACC annotate(dimension(rei(ncol,nlay),rel(ncol,nlay),play(ncol,nlay),reicmcl(ncol,nlay),relqmcl(ncol,nlay),pmid(ncol,nlay))) collapse(2)
……//源代码
! $ ACC END PARALLEL LOOP
```

其中,local(nm,km)表明将 nm、km 两个标量放在每个加速线程的 LDM 中,其性质是加速线程私有的,运算过程中不做数据传输。程序中访问的数组有 6 个,rei、rel、play、reicmcl、relqmcl、pmid,其中 rei、rel、play 是只读的,reicmcl、relqmcl、pmid 是只写的,对于只读的数据只需要拷入(copyin),只写的数据只需要拷出(copyout)。当动态数组在加速计算区中以数组的形式使用时,编译器往往无法判断出该动态数组所指向数组的维度信息,进而无法在设备存储器中申请适当大小的空间并拷贝数据。dimension 暗示中所指出的数组维度信息帮助编译器在加速计算区中申请设备空间并拷贝数据。collapse(2)表明 nm 和 km 两层循环合并,有利于提高并行度。

(2)数组转置。

```
do i=1, ncol
do isubcol=1, nsubcol
do ilev=2,nlay
if(CDF(isubcol, i, ilev-1)>1._rb-cldf(i,ilev-1))
then
CDF(isubcol,i,ilev)=CDF(isubcol,i,ilev-1)
else
CDF(isubcol,i,ilev)=CDF(isubcol,i,ilev) *
(1._rb-cldf(i,ilev-1))
endif
enddo
enddo
enddo
```

本段源码核心段是矩阵乘运算代码段,由于数组 CDF 的访问与 ilev 相关,因此 ilev-循环不能分块(tile)处理,因此将 ilev 循环调到最内层,循环的调整导致 CDF 数组的访问不连续。不连续的数据访问会导致数据传输效率较低,而 swapout 的作用是对 CDF 数组的原始数据按照指定的维度顺序进行转置,使用转置后的数据代替原始 CDF 数组的访问,这样可以使不连续的数据访问变成连续的数据访问。

调整后,ilev-循环对应 CDF 数组的第三维,isubcol-循环对应 CDF 数组的第一维,i-循环对应 CDF 数组的第二维,OpenACC 编译指示语句如下:

```
! $ ACC PARALLEL LOOP swap(CDF(dimension order:3,1,2)) copyin(cldf) annotate(dimension(cldf(ncol,nlay))) collapse(2)
……//源代码
! $ ACC END PARALLEL LOOP
```

(3)循环分块。

本段核心代码中,数组拷贝需要的 LDM 空间过大,处理器无法满足需求,考虑对循环进行分块处理,求解过程如下:

clwp、ciwp 和 tauc 数组的数据类型均为双精度浮点,数据大小为 8 字节,计算该循环使用三个数组的大小,其值为(90 * 61+90 * 61+140 * 90 * 61) * 8 = 6 236 640字节,远超出局存大小(65 536 字节),须分块后才能装入从核。

①计算最外层循环(ilev-循环)的分块大小。令该层的分块大小为 1,这时三个数组数据块的总大小为(90 * 1+90 * 1+140 * 90 * 1) * 8 = 102 240 字节,仍超过了局存大小。表明对数组该维的最小分块仍会使局存溢出,需要对下一维进行划分,取该层的分块值为 1[12]。

②计算次外层循环(i-循环)的分块大小。先令该层的分块大小为 1,这时三个数组的第 2 维按块大小 1 划分,三个数组数据块总大小为(1 * 1+1 * 1+140 * 1 * 1) * 8 = 1 136 字节,远小于局存大小。为充分利用存储空间,应增大分块值。当该层的分块值增大到 58 时,三个数组数据块总大小为(58 * 1+58 * 1+140 * 58 * 1) * 8 = 65 888,大于局存空间,达到临界值。所以该层分块大小确定为 57,算法结束[12]。

该算法得到的循环分块方案为(1,57,0),即最外层按块大小 1 划分,次外层按块大小 57 划分,最内层不划分[12],OpenACC 代码如下:

```
! $ ACC PARALLEL LOOPcopyin(clwp,ciwp,tauc) copyout(clwp_stoch(*,*,i),ciwp_stoch(*,*,i),tauc_stoch(*,*,i))
do ilev=1,nlay
! $ ACC LOOP tile(57)
do i=1, ncol
do isubcol=1, nsubcol
.....
clwp_stoch(isubcol,i,ilev)=clwp(i,ilev)
ciwp_stoch(isubcol,i,ilev)=ciwp(i,ilev)
tauc_stoch(isubcol,i,ilev)=tauc(isubcol,i,ilev)
.....
enddo
enddo
enddo
! $ ACC END PARALLEL LOOP
```

3.2.3　使用 ldm 数学函数库

在 OpenACC 加速指导语句的基础上,使用 ldm 数

学函数库。经测试,ldm 数学函数精度与默认库保持一致,计算性能提升 30% ~80% 不等,进一步提升了 GRAPES 模式整体运算效率。

3.2.4 结果与分析

实验采用分辨率为 0.5 °×0.5 °的 GRAPES 全球模式长波辐射方案,积分步长为 36,优化选项使用 -O2。由于 cldprmc、setcoef 耗时较少,没有进行优化。加速后,对比输出的全球长波辐射通量与模式输出值,经验证误差,相对误差的量级在 10^{-5} ~ 10^{-4}之间,在允许范围内。优化结果对比如表 1 所示。

表 1 优化结果对比

函数名称	主核运行时间	主从并行运行时间	主从并行加速比
mcica_subcol_gen_lw	64.14	11.06	5.8
inatm	30.48	6.12	5
taumol	20.97	5.24	4
rtrnmc	28.65	6.23	4.6

4 结束语

随着编译器的进一步优化和硬件技术的发展,OpenACC 加速与 CUDA 等在底层技术实现上的差距将越来越小,而且支持 OpenACC 加速指令转换将不仅仅针对 CUDA 设备,还包括其他更多厂商的硬件加速设备,从而能极大地提高 OpenACC 加速指令的普适性[13]。

文中以 GRAPES 模式长波辐射模块为加速对象,目前只应用了为数不多的 OpenACC 指令,就能得到较高的加速比,优化还有很大的提升空间。下一步将继续深入研究源代码数据结构,加强数据预处理,减少从核访主存操作,充分利用从核中高速缓存 SPM,提升数据访问效率;同时尝试多种 OpenACC 加速指令组和使用,进一步提升加速效果。

参考文献:

[1] 何沧平. OpenACC 并行编程实战[M]. 北京:机械工业出版社,2017.

[2] 曾文权,胡玉贵,何拥军,等. 一种基于 OPENACC 的 GPU 加速实现高斯模糊算法[J]. 计算机技术与发展,2013,23(7):147-150.

[3] 伍湘君,金之雁,陈德辉,等. 新一代数值预报模式 GRAPES 的并行计算方案设计与实现[J]. 计算机研究与发展,2007,44(3):510-515.

[4] 伍湘君,金之雁,黄丽萍,等. GRAPES 模式软件框架与实现[J]. 应用气象学报,2005,16(4):539-546.

[5] 陈德辉,沈学顺. 新一代数值预报系统 GRAPES 研究进展[J]. 应用气象学报,2006,17(6):773-777.

[6] 郭 妙,金之雁,周 斌. 基于通用图形处理器的 GRAPES 长波辐射并行方案[J]. 应用气象学报,2012,23(3):348-354.

[7] 刘文志. 并行编程方法与优化实践[M]. 北京:机械工业出版社,2015.

[8] LI L,XUE W,RANJAN R,et al. A scalable Helmholtz solver in GRAPES over large-scale multicore cluster[J]. Concurrency and Computation:Practice and Experience,2013,25(12):1722-1737.

[9] 郑 芳,许先斌,向冬冬,等. 基于 GPU 的 GRAPES 数值预报系统中 RRTM 模块的并行化研究[J]. 计算机科学,2012,39(6A):370-374.

[10] 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.

[11] 荆现文,张 华. McICA 云-辐射方案在国家气候中心全球气候模式中的应用与评估[J]. 大气科学,2012,36(5):945-958.

[12] 李雁冰,赵荣彩,赵 博,等. 面向异构多核处理器的循环分块[J]. 计算机工程与设计,2015,36(1):168-173.

[13] FARBER R. Parallel programming with OpenACC[M]. Massachusetts:Morgan Kaufmann Publishers,2015.

(上接第 64 页)

method for multi-task sparse learning problem[C]//Proceedings of the 2009 ninth IEEE international conference on data mining. Washington DC,USA:IEEE Computer Society,2009:746-751.

[11] ROSS D A,LIM J,LIN R S,et al. Incremental learning for robust visual tracking[J]. International Journal of Computer Vision,2008,77(1-3):125-141.

[12] BAO C,WU Y,LING H. Real time robust L1 tracker using accelerated proximal gradient approach[C]//IEEE conference on computer vision & pattern recognition. Washington DC,USA:IEEE Computer Society,2012:1830-1837.

[13] KWON J,LEE K M. Visual tracking decomposition[C]//IEEE conference on computer vision and pattern recognition. Washington DC,USA:IEEE Computer Society,2010:1269-1279.

[14] BABENKO B,YANG M,BELONGIE S. Visual tracking with online multiple instance learning[C]//IEEE conference on pattern analysis and machine intelligence. Washington DC,USA:IEEE Computer Society,2009:983-990.

[15] ADAM A,RIVLIN E,SHIMSHONI I. Robust fragments-based tracking using the integral histogram[C]//IEEE conference on computer vision and pattern recognition. Washington DC,USA:IEEE Computer Society,2006:798-805.