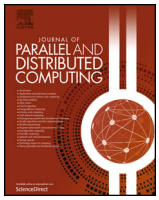
Journal of Parallel and Distributed Computing 202 (2025) 105094



Contents lists available at ScienceDirect

Journal of Parallel and Distributed Computing

journal homepage: www.elsevier.com/locate/jpdc





HIP-RRTMG_SW: Accelerating a shortwave radiative transfer scheme under the heterogeneous-compute interface for portability (HIP) framework

Zhenzhen Wang [a,b], Yuzhu Wang [a,*], Fei Li [a], Jinrong Jiang [c], Xiaocong Wang [d]

[a] School of Information Engineering, China University of Geosciences, Beijing, 100083, China
[b] Dawning Information Industry Co., LTD, Beijing, 100193, China
[c] Computer Network Information Center, Chinese Academy of Sciences, Beijing, 100190, China
[d] Institute of Atmospheric Physics, Chinese Academy of Sciences, Beijing, 100029, China

ARTICLE INFO

Keywords:
High performance computing
HIP
GPU
Shortwave radiation

ABSTRACT

With the development of higher-resolution atmospheric circulation models, the amount of calculation increases polynomially with resolution, and the calculation accuracy of physical processes is increasing rapidly. The traditional parallel computing methods based on multi-core CPUs can no longer meet the requirements of high efficiency and real-time computing performance of climate models. In order to improve the computational efficiency and scalability of the Atmospheric General Circulation Model, it is urgent to study efficient parallel algorithms and performance optimization methods for radiation physical process with massive calculations. In this paper, a heterogeneous multidimensional acceleration algorithm is proposed for the shortwave radiation transfer model (RRTMG_SW) based on HIP. Then, the HIP version of RRTMG_SW is developed, namely HIP-RRTMG_SW. In addition, combined with the "MPI + HIP" hybrid programming model, a multi-GPU implementation of RRTMG_SW is also proposed, and it makes full use of the multi-node, multi-core CPU and multi-GPU computing capability of a heterogeneous high performance computing system. Experimental results show that HIP-RRTMG_SW achieves 7.05× of acceleration in the climate simulation with 0.25° resolution using 16 AMD GPUs on the ORISE supercomputer compared with RRTMG_SW using 128 CPU cores. When using 1024 AMD GPUs, HIP-RRTMG_SW is 83.94× faster than RRTMG_SW with 128 CPU cores, indicating that the proposed multi-GPU acceleration algorithm has strong scalability.

1. Introduction

With the rapid development of the information society, the demand for computer processing ability in various application fields is increasing rapidly. High performance computing (HPC) can easily solve a large number of scientific problems through simulation, modeling and analysis, helping people solve the world's most difficult problems [1]. The development and progress of HPC technology is based on the deep cross-integration of mathematics, computing science, application fields, and other disciplines. Its application is becoming more and more extensive, and it has been used in many scientific research fields including weather forecasts, oil exploration, animation simulation and so on. Graphics processing units (GPU) have powerful floating point arithmetic capability, high memory access bandwidth and a high performance/energy consumption ratio [2]. In the era of pursuing green HPC, many supercomputers have adopted the heterogeneous architecture of CPU/GPU. NVIDIA is the first to provide supercomputing technology on GPUs, and launches the compute unified device architecture (CUDA) [3], which has won the favor of more and more researchers. AMD also provides a set of GPU solutions and has its own heterogeneous-compute interface for portability — HIP. HIP can also convert CUDA applications into applications that can be executed on AMD GPUs [4]. Facing the huge demand and wide application of the HPC industry, researchers in the field of high-performance computing would benefit greatly from mastering GPU programming technologies and their respective heterogeneous programming models.

Generally speaking, various fields that require large-scale numerical simulation calculation and large-scale data analysis and processing can be accelerated by HPC technology. At present, the Earth system model is developing towards the global kilometer level and the 100-meter level in critical areas to provide better simulation results and more detailed spatial information. As the resolution of the model increases, its com-

* Corresponding author.
E-mail addresses: wangzhzhen@gmail.com (Z. Wang), wangyz@cugb.edu.cn (Y. Wang), lifei2021@email.cugb.edu.cn (F. Li), jjr@sccas.cn (J. Jiang), wangxc@lasg.iap.ac.cn (X. Wang).

https://doi.org/10.1016/j.jpdc.2025.105094
Received 24 March 2024; Received in revised form 2 April 2025; Accepted 17 April 2025
Available online 23 April 2025
0743-7315/© 2025 Elsevier Inc. All rights are reserved, including those for text and data mining, AI training, and similar technologies.

putational load will increase exponentially on a large scale, requiring a large number of computational resources. Therefore, the development of high-resolution Earth system models is inseparable from the support of HPC [5–9]. As the most important and complex component model of the Earth system model, the atmospheric general circulation model is mainly composed of a dynamic framework and physical processes. Taking the atmospheric circulation model IAP AGCM4.0 [10] (1.4° resolution version) of the Chinese Academy of Sciences-Earth System Model (CAS-ESM) as an example, the calculation time of the physical process accounts for 56.48% of the total calculation time in the serial calculation. Among them, the radiation and cloud microphysical processes occupy the most of the calculation time of the physical processes [11].

In view of the time-consuming process of radiation physics, this paper mainly focuses on its large-scale, scalable, efficient heterogeneous computing and performance optimization methods, providing a theoretical basis for efficient parallel computing of other physical processes. IAP AGCM4.0, the atmospheric component model of CAS-ESM, primarily adopts the rapid radiative transfer model for the general circulation model (RRTMG) [12] developed by the Atmospheric and Environmental Research (AER) group, as its radiation parameterization scheme. The RRTMG calculates the radiation transport using the correlation K distribution method, dividing the atmosphere into columns in a longitude and latitude grid. The calculation of radiation in the vertical layer is dependent, and the calculation of each horizontal column is completely independent. RRTMG includes both longwave and shortwave radiation, in which longwave radiation mainly calculates the atmospheric longwave flux and cooling rate, and shortwave radiation mainly calculates the solar shortwave flux and heating rate. RRTMG is very suitable for parallel processing because it is a “column physical process” and there is no data correlation in the horizontal direction. To meet the higher resolution development requirements of IAP AGCM4.0 and improve the overall scalability of CAS-ESM, this paper takes the RRTMG shortwave radiation scheme (RRTMG_SW) as the research object and proposes a heterogeneous multidimensional acceleration algorithm of RRTMG_SW based on HIP, namely HIP-RRTMG_SW. The algorithm has been successfully applied in CAS-ESM with different resolutions. In CAS-ESM with 0.25° resolution, HIP-RRTMG_SW was simulated with 16 AMD GPUs and 16 CPU cores for one model day, and the acceleration performance was 7.05× better than original Fortran algorithm with 128 CPU cores. The main contributions of this paper are as follows:

(1) A GPU-based RRTMG_SW algorithm is proposed, implemented in HIP C, with performance optimized through memory access and I/O transmission improvements. Optimizations include reducing global memory usage by storing intermediate data in registers, and reducing unnecessary data transfer, using fixed memory technology and asynchronous data transfer, further optimize I/O operations and improve the utilization efficiency of GPU resources.
(2) A HIP-based 3D acceleration algorithm for RRTMG_SW is proposed, extending horizontal parallelism and optimizing the vertical layer and *g points* or band dimensions. Dimensional optimization is applied to improve efficiency at higher resolutions. This algorithm is successfully applied to the CAS-ESM framework at different resolutions.
(3) A multi-GPU acceleration algorithm for RRTMG_SW is proposed, utilizing the “MPI + HIP” hybrid programming model. The model optimizes load balancing and communication between multi-node, multi-core CPUs and GPUs, effectively reducing computation time and improving resource utilization.

In summary, the work of this paper has supported the large-scale, real-time computation of CAS-ESM with different resolutions. The rest of this article is organized as follows. Section 2 analyzes the current heterogeneous computing of radiation physics processes and the application effect of using HIP framework and AMD GPUs, and determines the research direction. Section 3 describes the solving process of the radiation model equation and heterogeneous ecological environment. Section 4 presents the 1D acceleration algorithm, performance optimization methods, 3D acceleration algorithm and multi-GPU acceleration algorithm for RRTMG_SW under the HIP framework. Section 5 introduces the experimental platform, the acceleration efficiency of the algorithm, and the arithmetic intensity analysis. Section 6 provides an objective evaluation of the work and analyzes the shortcomings of the future work outlook and planning.

2. Related work

In order to use the GPU-accelerated RRTMG longwave radiation scheme (RRTMG_LW) in the mesoscale weather forecast model (WRF), Price et al. [13] first rewrote the Fortran code of RRTMG_LW in C language, then the GPU acceleration of RRTMG_LW was implemented using CUDA C. In performance optimization, 19 optimization schemes are implemented. Regardless of I/O overhead, RRTMG_LW achieves a 127× speedup on Tesla K40 GPU compared to Intel Xeon E5-2603 single-core computing time. To improve vectorization, thread parallelism, and data locality, Michalakes et al. [14] refactored the RRTMG code to achieve a 3× speedup on many-core MIC (Intel’s Many Integrated Core). This is an effective attempt to accelerate RRTMG on the MIC, but the acceleration effect is not very obvious.

Kondratyuk et al. [15] analyzed the performance of LAMMPS, GROMACS, and OpenMM molecular dynamics packages on NVIDIA Volta and AMD Vega20 GPUs, and their work on porting LAMMPS from CUDA to HIP demonstrated the considerable computational advantages of AMD GPUs. Kuznetsov et al. [16] ported the classical dynamics algorithm from CUDA to AMD’s Radeon Open Computing Platform (ROCm) [17], and the results show that converting an application from CUDA to HIP does not require extensive modifications, and the performance on AMD GPUs does not decline. Wang et al. [18] proposed a HIP-based heterogeneous acceleration scheme for the high-resolution ocean circulation model LICOM3. On 384 AMD GPUs, LICOM3-HIP is 42× faster than the same number of CPU cores. Borisov [19] implemented the “Kuznyechik” and “Magma” symmetric encryption algorithms based on CUDA and HIP, respectively, and the results showed that the computing performance on AMD Radeon RX Vega 56 GPU surpassed that of the NVIDIA GeForce GTX 1080Ti and NVIDIA GeForce RTX 2080Ti GPUs. In the same year, a HIP-based gradient descent algorithm [20] was researched and implemented, achieving about 6× performance improvement on one AMD Radeon RX Vega 56 GPU compared to the 8-core CPU version.

Jang et al. [21] proposed a technique to improve the memory efficiency of applications on data-parallel architectures. The method was demonstrated on a set of benchmark kernels, resulting in an 11.4× speedup on AMD GPUs compared to the baseline GPU implementation. Dao et al. [22] proposed two accurate models based on GPU platforms: a sampling-based linear model and a model based on machine learning (ML) techniques. On AMD Radeon HD 6970 GPUs, the model has an error rate of about 8%, and the proposed technique outperforms existing models by a factor of 5× to 6× in terms of accuracy. Blakeney et al. [23] proposed a novel parallel block distillation algorithm to speed up the distillation process of complex DNNs. Experimental results on 4 AMD Geforce RTX 2080Ti GPUs show that the algorithm achieves 3× speedup in VGG distillation and 3.5× speedup in ResNet distillation.

Through a comprehensive review of the existing literature, the following conclusions can be drawn: (1) There have been studies on using GPUs to accelerate the radiation process with significant acceleration effects, proving the necessity of using GPUs to accelerate the radiation process. However, there are still few studies on heterogeneous acceleration algorithms for radiation processes based on HIP. This work particularly emphasizes the potential of HIP in the study of parallel computing and scalable parallel algorithms for shortwave radiation processes, proving its effectiveness in this context. (2) Currently, there are few heterogeneous computing algorithms for radiation physical processes based

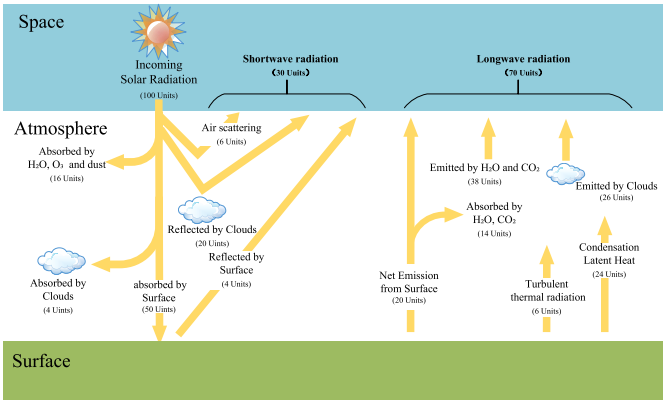


Fig. 1. Radiation physical model.

on multi-GPU. The parallel computing performance and scalability are poor, and the performance of multi-node, multi-core CPU and multi-GPU models in today's mainstream large-scale heterogeneous computing system is not fully utilized. Theoretical research on large-scale scalable heterogeneous parallel algorithms for radiation physical processes is lacking, which cannot meet the requirements of multi-million-core heterogeneous computation for high-resolution earth system models.

According to the characteristics of computation-intensive physical processes of shortwave radiation and the design of efficient heterogeneous computing algorithms based on HIP, this study will focus on the research of heterogeneous acceleration methods and optimization methods based on AMD GPUs, including: a) investigation of HIP's computational characteristics and mapping strategy system structure, with an emphasis on complex memory access levels; b) development of a HIP-based multidimensional acceleration algorithm for the shortwave radiation transmission model; and c) performance optimization methods based on HIP, focusing on reducing global memory usage, optimizing data transfer, and increasing kernel function parallelism to further improve the efficiency of heterogeneous computations in radiative transfer models.

3. Model and HIP framework

3.1. Radiation model

The change of climate is the manifestation of the atmospheric thermal condition, which is essentially the reflection of heat absorption in the air. The quantity and variation of heat in the air are the results of the exchange and transformation of heat among solar radiation, surface radiation, and atmospheric radiation, i.e., the radiation process can strongly influence the energy balance at the top of the atmosphere (TOA) and the surface. Radiation transfer is an important process in climate models, as shown in Fig. 1. Solar radiation is the main source of energy at the Earth's surface with a wide range of wavelengths, with most of the energy concentrated between 0.15 and 4.0 μm. Since the temperature of the solar surface is about 6000 K, according to the Venn displacement law, its maximum radiation capacity corresponds to the wavelength of 0.467 μm, so it is generally called shortwave radiation. The temperature of the ground and the atmosphere is much lower than that of the sun, and the wavelengths of their radiation are mainly between 3 and 120 μm, so the radiation from the ground and the atmosphere is collectively referred to as long-wave radiation.

When solar radiation enters the atmosphere, the interaction between various gas molecules, as well as the absorption, scattering, and reflection of solar radiation by suspended particles in the atmosphere, result in the energy reaching the ground not only being much smaller in quantity than atmospheric radiation energy but also undergoing significant changes in quality, such as spectral components. From the perspective of the radiation direction, it can be divided into direct radiation and scattered radiation, so the total change of radiation intensity is given by the formula:

$$dI_\lambda = -k_\lambda \rho I_\lambda ds + j_\lambda \rho ds. \tag{1}$$

Where the former term is the change in radiation intensity in the direction of direct radiation, and the latter term is the change in radiation intensity in the direction of scattered radiation. Among them, k_λ represents the mass extinction section of the radiation wavelength λ, ρ is the material density, I_λ is the intensity of the incident radiation, ds is the thickness in the direction of propagation, and j_λ is called the source function, which is different from the mass extinction section k_λ unit (the former relates more with the concept of intensity). To facilitate calculation, we define the source function J_λ as follows:

$$J_\lambda \equiv j_\lambda / k_\lambda. \tag{2}$$

Then the radiative transfer equation is:

$$\frac{dI_\lambda}{k_\lambda \rho ds} = -I_\lambda + J_\lambda. \tag{3}$$

To simplify the calculation, it is usually assumed that the medium of electromagnetic wave transmission is plane-parallel, that is, the medium can be divided into a number or infinite number of parallel layers, the radiation properties within each layer is the same, the properties between the layers are different. Therefore, for plane-parallel media, the radiation transmission equation can be written as:

$$-\mu \frac{dI_\lambda}{d\tau_\lambda} = -I_\lambda + J_\lambda. \tag{4}$$

Where θ is the angle between the radiation direction and the stratification direction normal, namely the zenith angle, and $\mu = \cos\theta$. τ is the optical thickness of the atmosphere, $d\tau_\lambda = -k_\lambda \rho ds$.

The difficulty of solving the shortwave radiation equation lies in the multiple scattering part, that is, the source function J_λ. When the electromagnetic wave incident in one direction is scattered by the medium to the other direction, the scattering process generally can be divided into single scattering and multiple scattering processes. The sum of single scattering and multiple scattering is given by the following formula:

$$\begin{aligned} J(\tau,\Omega) = &\frac{\omega}{4\pi} F_0 e^{-\frac{\tau}{\mu_0}} P(\Omega,\Omega_0) + \\ &\frac{\omega}{4\pi} \int_{4\pi} I(\tau,\Omega') P(\Omega,\Omega') d\Omega' + B[T(\tau)]. \end{aligned} \tag{5}$$

In the formula (5), the first term is the source function generated by single scattering, the second term is the source function resulting from multiple scattering, and the last term is the emitted source function. For single scattering, F_0 is the initial value of the flux density of the incident radiation, μ_0 is the zenith angle cosine of the propagation direction Ω_0, and ω is the reflectance of the single scattering. $P(\Omega,\Omega')$ is called the scattering phase function, mainly to describe the intensity distribution ratio of electromagnetic waves scattered from direction Ω' to direction Ω by the medium, and $P(\Omega,\Omega')/4\pi$ is normalized where $P(\Omega,\Omega_0)$ is the proportion of directional Ω_0 electromagnetic waves scattered to directional Ω. For multiple scattering, since the source function of multiple scattering is the sum of radiation from all directions and scattered to direction Ω, the integral over 4π space of direction Ω' is required. $I(\tau,\Omega')$ is the intensity of radiation at τ in the direction of Ω'. $B[T]$ is the Planck function, and represents the brightness of the outgoing radiation emitted by the object at a temperature of T. Its intensity is isotropic independent of the direction of the outgoing radiation.

Considering the scattering and emission source functions, the basic scalar equation of radiative transport in a plane-parallel atmosphere can be written as:

$$-\mu\frac{dI(\tau,\Omega)}{d\tau} = -I(\tau,\Omega) + \frac{\omega}{4\pi}F_0 e^{-\frac{\tau}{\mu_0}}P(\Omega,\Omega_0) + \frac{\omega}{4\pi}\int_{4\pi} I(\tau,\Omega')P(\Omega,\Omega')d\Omega' + B[T(\tau)]. \tag{6}$$

Normally, only numerical or simplified solutions exist for this equation [24,25]. Next, we will use the two-stream approximation to simplify the solution of this equation.

3.2. Model solution

We assume that radiative transmission is independent of azimuth and only related to μ. Without considering emission, it can be simplified as:

$$-\mu\frac{dI(\tau,\mu)}{d\tau} = -I(\tau,\mu) + \frac{\omega}{4\pi}F_0 e^{-\frac{\tau}{\mu_0}}P(\mu,\mu_0) + \frac{\omega}{2}\int_{-1}^{1} I(\tau,\mu')P(\mu,\mu')d\mu'. \tag{7}$$

The scattering phase function $P(\mu,\mu')$ in the formula uses finite N-term Legendre polynomial analysis and Gauss formulas to solve with a finite sum instead of an integral, and the final radiation equation is:

$$-\mu_i\frac{dI(\tau,\mu_i)}{d\tau} = -I(\tau,\mu_i) + \frac{\omega}{4\pi}F_0 e^{-\frac{\tau}{\mu_0}}\sum_{l=0}^{N}(-1)^l C_l P_l(\mu_i)P_l(\mu_0) + \frac{\omega}{2}\sum_{l=0}^{N}C_l P_l(\mu_i)\sum_{j=-n}^{n}a_j P_l(\mu_j)I(\tau,\mu_j). \tag{8}$$

Where $\mu_i(-n,n)$ represents the direction of radiation flow. Both C_l and P_l are coefficients of a Legendre polynomial expansion. The weight value is a_j, and $a_j = \frac{1}{P'_{2n}(\mu_j)}\int_{-1}^{1}\frac{P_{2n}(\mu)}{\mu-\mu_j}d\mu$, μ_j is the zero of the n-order Legendre polynomial $P_{2n}(\mu)$, and are: $a_{-j} = a_j$, $\mu_{-j} = -\mu_j$, $\sum_{j=-n}^{n}a_j = 2$. If we take $n=1$, then we get two radiation flows, namely $j=\pm1$, at this time, $N=1$, and the corresponding Gaussian point and weight value are respectively: $\mu_1 = \frac{1}{\sqrt{3}}$, $a_1 = a_{-1} = 1$. After rearranging the terms and using I^+ to represent $I(\tau,\mu_1)$ and I^- to represent $I(\tau,-\mu_1)$, two simultaneous equations can be derived, namely,

$$\mu_1\frac{dI^+}{d\tau} = I^+ - \omega(1-b)I^+ - \omega b I^- - s^- e^{-\tau/\mu_0}. \tag{9}$$

$$-\mu_1\frac{dI^-}{d\tau} = I^- - \omega(1-b)I^- - \omega b I^+ - s^+ e^{-\tau/\mu_0}. \tag{10}$$

Where g is an asymmetric factor, $g = \frac{C_1}{3} = \frac{1}{2}\int_{-1}^{1}P(cos\phi)cos\phi dcos\phi$, C_1 is a Legendre polynomial system, ϕ is the scattering angle, and $b = \frac{1-g}{2}$. Equations (9) and (10) are two-stream approximate radiative transfer equations. I^+ and I^- are solved by two equations. Through formula $F^{\mp} = 2\pi\int_0^{\mp1} I(\mu)\mu d\mu$, the radiation flux is obtained as follows:

$$F = F^+ - F^-. \tag{11}$$

The radiant heating (or cooling) rate is expressed as:

$$\frac{d\theta}{dt} = -\frac{1}{c_p\rho}\frac{dF}{dz} = \frac{g}{c_p}\frac{dF}{dP}. \tag{12}$$

Where P is the pressure, c_p is the specific heat at P, g is the acceleration of gravity, and ρ is the air density of a layer [26].

The RRTMG model performs a multiple scattering using the two-stream radiative transfer algorithm. In the process of molecular absorption and scattering, the method of correlation K distribution is used to calculate atmospheric transmittance (for further details, refer to [27]). The shortwave part of the RRTMG model has 14 bands (solar flux fraction) and 112 g points (discrete absorption intensity within the band). The RRTMG_SW shortwave radiation algorithm is mainly divided into four functions, which are *inatm_sw*, *cldprmc_sw*, *setcoef_sw* and *spcvmc_sw*. Refer to [28] for detailed descriptions of the first three functions. The fourth function, *spcvmc_sw*, is briefly described here. It mainly calls three sub-functions (*taumol_sw*, *reftra_sw*, and *vrtqdr_sw*), which do the following:

(1) *taumol_sw*, calculate the optical depth and Planck fraction of each spectral band.
(2) *reftra_sw*, calculate the reflectivity and atmospheric transmittance of the second-rate approximation model.
(3) *vrtqdr_sw*, calculate the vertical orthogonal integral in the second-rate approximation model.
(4) *spcvmc_sw*: *taumol_sw*, *reftra_sw* and *vrtqdr_sw* are called to realize the overall solution of the second-rate approximation model.

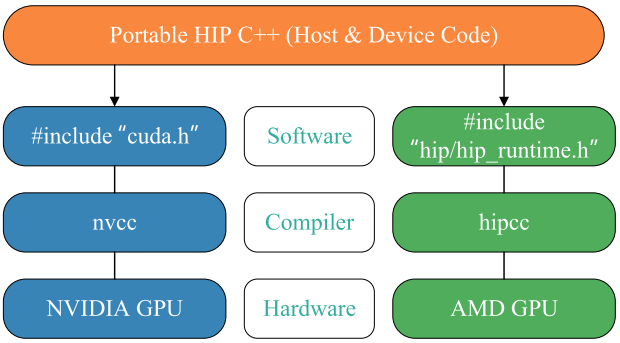


Fig. 2. The comparison of CUDA and HIP at different levels.

3.3. ROCm platform and HIP framework

ROCm is the HPC platform for AMD GPUs. It is a lightweight and modular GPU software development environment that provides a variety of development tools and runtime environments. Each component in ROCm builds a complete platform function from bottom to top according to its function, which is suitable for large-scale application computing, compiler and program runtime component development, enabling developers to quickly and easily develop heterogeneous computing applications on CPU, AMD GPU, NVIDIA GPU and other devices.

ROCm currently supports two heterogeneous parallel programming models, HIP and OpenCL, both of which provide corresponding development components and runtime environment support. HIP is the most important heterogeneous parallel programming model in the ROCm software stack. It defines a programming model that can be compiled and run simultaneously on multiple platforms, including NVIDIA GPUs, AMD GPUs, and so on. The HIPify tool [29] provided by HIP can automatically convert CUDA applications into the HIP kernel language and runtime API. The HIP compiler selects the corresponding platform as the target for a HIP source compilation based on the current system environment, as shown in Fig. 2. Data structure of the HIP and interface model are highly consistent with CUDA, and the syntax structure of the kernel function is basically the same.

4. HIP-based acceleration algorithms

4.1. 1D acceleration algorithm

In the whole process of atmospheric radiation propagation, the radiative transfer algorithm combines longitude and latitude in the vertical direction of a 3-dimensional model into horizontal dimensions at the code level. Therefore, in the atmospheric dimension, the first dimension describes the number of horizontal columns of the atmosphere, and the second dimension is the number of model layers at a certain resolution. In CAS-ESM with 1.4° resolution, the number of horizontal columns in the first dimension is 128×256, and the number of model

layers in the second dimension is 52. Since the physical process of the atmospheric circulation model belongs to "column physical process", the 1D acceleration algorithm of shortwave radiation transmission model is studied. Each stream processor (SP) or HIP core of the AMD GPU is responsible for calculating the radiation flux at each horizontal grid point. The current research work continues the previous development of CUDA algorithms [28,30]. Firstly, the HIPify tool is used to convert CUDA algorithms into HIP C algorithms, and then the converted code is modified in detail according to the unique features of the HIP framework. Algorithm 1 is the HIP C-based shortwave radiation acceleration algorithm.

Algorithm 1 HIP C-based acceleration algorithm of RRTMG_SW with 1D domain decomposition and the implementation of HIP-RRTMG_SW.

```
extern "C" void rrtmg_sw_d(parameters) {
1. Copy input data to GPU device
   // read the atmospheric profile parallel to the plane
   // n is taken as a positive integer
2. hipLaunchKernelGGL(inatm_sw, ncol/(64*n), 64*n, parameters);
   // set the cloud depth for McICA
3. hipLaunchKernelGGL(cldprmc_sw, ncol/(64*n), 64*n, parameters);
   // calculate pressure and temperature indices in radiation
4. hipLaunchKernelGGL(setcoef_sw, ncol/(64*n), 64*n, parameters);
5. Data is transmitted from the device to host;
6. for i ← 1, ncol {
   // calculate the total optical depth of aerosol
7.    if(iaer == 0) {
8.      ztaua(i, nbndsw, nlayers) = 0.0;
9.    }
10.   else {
11.     for lay ← 1, nlayers {
12.       for ib ← 1, nbndsw {
13.         ztaua(i, ib, lay) = taua(i, ib, lay);
14.       }
15.     }
16.   }
17. }
18. Data is transmitted from the host to device;
   // realize the whole solution of the two-stream approximation model
19. hipLaunchKernelGGL(spcvmc_sw, ncol/(64*n), 64*n, parameters){
      // calculate the optical depth and Planck fraction for each spectral band
20.   taumol_sw(parameters);
      // calculate reflectivity and atmospheric transmittance
21.   reftra_sw(parameters);
      // compute vertical orthogonal integrals
22.   vrtdqr_sw(parameters);
23. };
24. Copy results to host;
   // Judge whether atmospheric horizontal profile data is completed
25. if it is not completed goto 1
}
```

Algorithm 1 shows the differences between the HIP C-based shortwave radiation acceleration algorithm and the CUDA C-based acceleration algorithm. (1) The number of threads launched by each kernel, which is a multiple of 32 in the CUDA algorithm, is changed to a multiple of 64. (2) The HIP framework is not friendly to the use of local memory. All small arrays in CUDA are stored in registers or local memory, but they are moved to global memory in HIP. (3) The "restrict" keyword is added to the arrays in the kernel parameter list, which informs the compiler that all operations modifying the content pointed to by this pointer are based solely on this pointer, and that there are no other ways to perform modifications. This helps the compiler optimize the code better and generate more efficient assembly code. (Note that (2) and (3) are not reflected in the pseudo-code of Algorithm 1.)

4.2. Performance optimization

In accelerated computing based on a HIP framework, when the kernel function is started and executed, data transmission and memory access between host and device will be carried out. When these operations are optimized, the computational efficiency of the algorithm can be greatly improved. Therefore, we used two performance optimization methods based on HIP: reducing global memory access and data transfer optimization.

4.2.1. Reduce global memory access

Global memory is the memory for writing operations by both GPU and CPU. The storage space of global memory is large, but the access cost is high, and the channel cannot be used efficiently. (1) Considering the characteristics of HIP-RRTMG_SW and hardware memory, other types of memory can be used to replace the access to global memory, such as changing the three-dimensional intermediate arrays used in the *spcvmc_sw* kernel function's internal calculation process to variables stored in registers, which not only compresses the data dimensions but also effectively utilizes register memory to improve memory access speed. (2) Based on the distinct operational characteristics of the shortwave model, it is possible to optimize the code by eliminating redundant blocks, thereby reducing the consumption of global memory resources. (3) For non-input/output array variables, an on-demand allocation and release strategy can improve memory utilization efficiency.

4.2.2. Data transfer optimization

Firstly, the intermediate data calculated between different kernels are put into the GPU for the calculation to avoid the data transmission operation of calculation back to the CPU and then back to the GPU, thus improving the efficiency of the algorithm. In HIP-RRTMG_SW, we place the portion of the code that calculates the total optical depth of aerosol between *setcoef_sw* and *spcvmc_sw* into *setcoef_sw* to reduce the interaction between the host side and the device and carry out parallel processing for this part. Secondly, we utilize pinned memory technology to optimize the necessary part of the data transfer. The pinned memory data transfer method always resides in random access memory and is not allocated to disk, avoiding the extra transfer overhead between paging memory and the pinned host buffer. Finally, on the basis of pinned memory, HIP streams are used to data transfer with the *spcvmc_sw* kernel function for asynchronous data transfer, realizing the optimization of the data transfer part.

4.3. 3D acceleration algorithm

The acceleration in the horizontal direction is completed. In the vertical direction, the acceleration calculation can also be carried out for the part where there is no data correlation. RRTMG uses the correlation K distribution method to calculate the radiation flux, which decomposes the radiation spectrum into several bands. Due to the rapid variation of absorption lines in the gas molecular bands, each band absorption intensity value is thrown into an accumulative distributed intensity function. To obtain the radiation flux of the bands, this distribution function is discretized in each band by means of g iterative integration. In each band, the discrete absorption intensity is captured by the g points. RRTMG_SW has 14 bands and 112 g points. There is no data correlation in the band and *g points* dimensions, and the accelerated calculation can be carried out. Therefore, the 3D heterogeneous calculation strategy of radiation physical processes is to accelerate the calculation in the horizontal direction, vertical direction and band or *g points* dimensions. Its 3D regional division is shown in Fig. 3.

Data dependence is one of the main problems to be solved in the design of multi-dimensional parallel algorithms. In HIP-RRTMG_SW, the data dependence is unavoidable, and the decoupling with the internal structure of specific kernel functions is the main way of dimensional-raising acceleration. The decoupling method of this algorithm is to

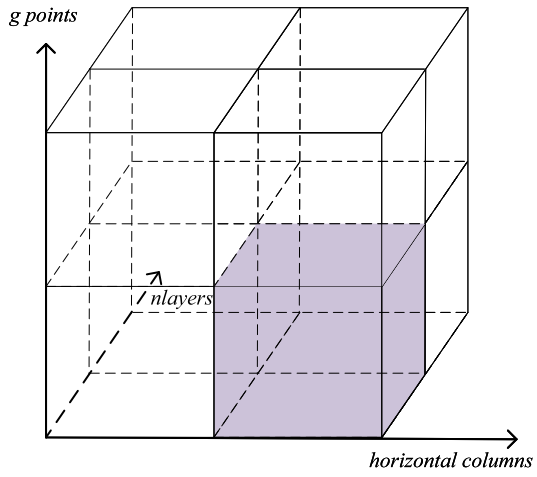


Fig. 3. Schematic diagram of 3D region decomposition.

separate the calculation process by increasing the kernel function. The method achieves the purpose of decoupling the data dependence and improving the calculation performance. Algorithm 2 describes the 3D acceleration algorithm.

Algorithm 2 HIP C-based acceleration algorithm of RRTMG_SW with 3D domain decomposition and the implementation of HIP-RRTMG_SW.

extern "C" void *rrtmg_sw_d*(parameters){
1. Copy input data to GPU device
   // *n* is taken as a positive integer
2. **dim3** *thread1*(64**n*, 4, 1);
3. **dim3** *block1*((*ncol*+(64**n*)-1)/(64**n*), (*nlayers*+4-1)/4, 1);
4. **dim3** *thread2*(32, 64**n*, 4);
5. **dim3** *block2*((*ngptsw*+32-1)/32, (*ncol*+(64**n*)-1)/(64**n*), ((*nlayers*-1)+ 4-1)/4);
6. **dim3** *thread3*(64**n*, 4, 1);
7. **dim3** *block3*((*ncol*+(64**n*)-1)/(64**n*), ((*nlayers*-1)+4-1)/4,1);
8. hipLaunchKernelGGL(*inatm_sw_d1*, *block1*, *thread1*, parameters);
9. hipLaunchKernelGGL(*inatm_sw_d2*, **dim3**(*ceil*(*ncol*/(64.0**n*))), **dim3**(64**n*), parameters);
10. hipLaunchKernelGGL(*inatm_sw_d3*, *block2*, *thread2*, parameters);
11. hipLaunchKernelGGL(*inatm_sw_d4*, *block3*, *thread3*, parameters);
12. hipLaunchKernelGGL(*inatm_sw_d5*, **dim3**(*ceil*(*ncol*/(64.0**n*))), **dim3**(64**n*), parameters);
13. hipLaunchKernelGGL(*cldprmc_sw_d*, *block1*, *thread1*, parameters);
14. hipLaunchKernelGGL(*setcoef_sw_d1*, *block1*, *thread1*, parameters);
15. hipLaunchKernelGGL(*setcoef_sw_d2*, **dim3**(*ceil*(*ncol*/(64.0**n*))), **dim3**(64**n*), parameters);
16. hipLaunchKernelGGL(*setcoef_sw_d3*, *block1*, *thread1*, parameters);
17. hipLaunchKernelGGL(*taumol_sw_d*, *block1*, *thread1*, parameters);
18. hipLaunchKernelGGL(*spcvmc_sw_d1*, *block1*, *thread1*, parameters){
19. *reftra_sw_d*(parameters);
20. };
21. hipLaunchKernelGGL(*spcvmc_sw_d2*, **dim3**(*ceil*(*ncol*/(64.0**n*))), **dim3**(64**n*), parameters){
22. *vrtdqr_sw_d*(parameters);
23. };
24. Copy results to host;
   //Judge whether atmospheric horizontal profile data is completed
25. **if** it is not completed **goto** 1
}

According to the code structure and parallelism, the *inatm_sw* function is divided into *inatm_sw_d1*, *inatm_sw_d2*, *inatm_sw_d3*, *inatm_sw_d4* and *inatm_sw_d5*. The sub-function *inatm_sw_d1* is responsible for initializing part of the array. Since it has data of horizontal and vertical dimensions, and the data calculation of these two dimensions is independent, the 2D parallel calculation is used. The sub-function *inatm_sw_d2* is responsible for assigning values to some 2D and 3D arrays. Like *inatm_sw_d1*, *inatm_sw_d2* can perform 2D parallel computation. The sub-function *inatm_sw_d3* is assigned to the 3D array. Since it has no dependence on data calculation in horizontal, vertical and *g points* dimensions, 3D parallel calculation is used. The independent part of the horizontal and vertical dimension calculation in the original *inatm_sw* function was extracted into *inatm_sw_d4* for 2D parallel computation. Finally, there is *inatm_sw_d5*, which is responsible for the rest of the computation that is dependent and difficult to decouple, so 1D is still used for parallel computation.

Similarly, *cldprmc_sw* and *setcoef_sw* are decoupled according to the function structure. The *taumol_sw* function inside the *spcvmc_sw* function is decomposed for separate two-dimensional parallel processing, because *taumol_sw* calculates the data of 14 bands in shortwave, and the calculation of 14 bands is completely independent in horizontal and vertical dimensions. The remaining *spcvmc_sw* is decoupled into two parts, in which *spcvmc_sw_d1* was 2D accelerated, while *spcvmc_sw_d2* keeps the original 1D acceleration due to the integration of the data of 14 bands and the calculation of the total radiation flux and heating rate. We couple the optimized code to the system for correctness verification and performance tests. Subsequently, we carry out a large number of experiments to test it rigorously.

4.4. Multi-GPU acceleration algorithm

With the improvement of the climate model resolution, the computing performance and video memory capacity of a single GPU can no longer meet the requirements of its fast calculation. So it is necessary to study multi-GPU heterogeneous computing algorithms for high resolution physical processes. In multi-GPU computing, the CPU is required to drive the GPU. In the atmospheric general circulation model with 0.25° resolution, the corresponding number of horizontal grid points is 1152×768. If four GPUs are used, each GPU is responsible for calculating the radiation flux corresponding to 288×768 horizontal grid points. If the global memory is limited, each GPU needs several iterations to complete the corresponding calculation of 288×768 horizontal grid points. As shown in Fig. 4, a GPU can only complete grid point calculation in a chunk at a time, requiring four iterations.

The acceleration algorithm of RRTMG_SW on multi-GPU is as follows:

1) The total grid is divided into sub-grids. Each CPU core is responsible for the calculation of its sub-grid points. Each CPU core starts a GPU and then sends initial data to the GPU.
2) After receiving the initial data, a chunk number of threads on each GPU carry out parallel calculation of radiation flux for grid points in a chunk, that is, each thread is responsible for the radiation flux calculation of one grid point. The result is then passed to the CPU.
3) The CPU receives the calculation results. If the computing tasks assigned to each CPU are not completed within an integral step, the computing tasks corresponding to the remaining grid points continue from step 1). Fig. 5 shows the multi-GPU computing process of RRTMG_SW.

To fully leverage the computing power of multi-node, multi-GPU algorithms in heterogeneous systems, this paper combines the "MPI + HIP" hybrid programming model and proposes a HIP-based multi-node heterogeneous parallel algorithm for RRTMG_SW. MPI is used to perform communication among different nodes and is responsible for coarse-grained parallelism of computing tasks. HIP is used to drive powerful GPU devices to extract inherent fine-grained parallelism in computing tasks. Fig. 6 depicts the "MPI + HIP" hybrid programming model.

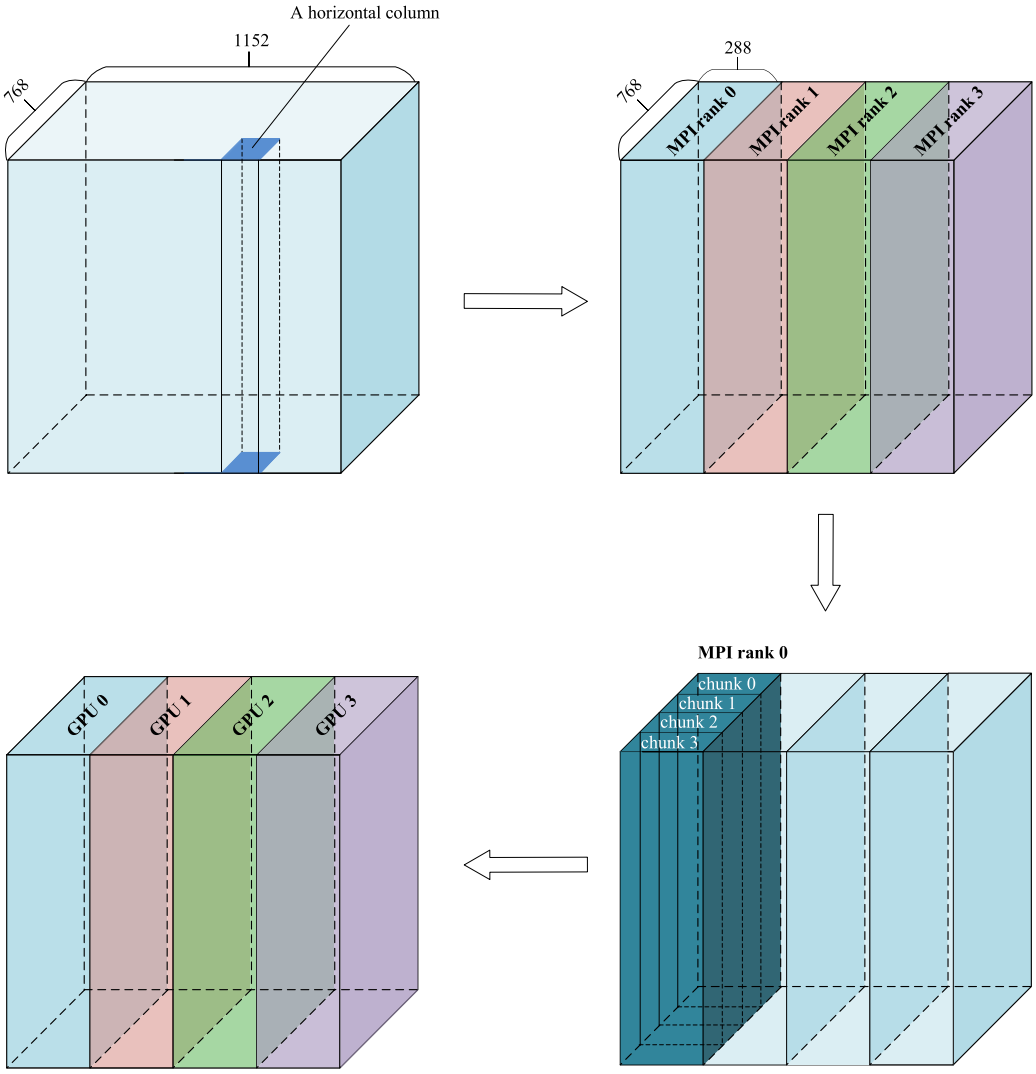


Fig. 4. Data partitioning and grid allocation.

5. Results and discussion

5.1. Experimental environment

The 1.4° resolution and 0.25° resolution experiments were conducted on the ORISE supercomputer at the Computer Network Information Center of the Chinese Academy of Sciences. ORISE is a heterogeneous supercomputing system consisting of 7,000 nodes. Each node is equipped with 4-way 8-core 1.9 GHz x86 CPU processors (32 cores in total), 128 GB DDR4 memory, and 128×PCI-E 3.0 channels. In addition, each node is equipped with four improved GPU cards based on the AMD GFX906 architecture. The performance of these GPU cards has been optimized and has been developed to a level comparable to that of AMD MI60. Each GPU is equipped with 4,096 cores and 16 GB HBM2 video memory. The ORISE system achieves efficient communication between nodes through a high-speed network of 25 GB/s, and adopts a HIP-based backend computing framework (HIP version 3.9.1) to achieve efficient collaborative computing between CPU and GPU.

The CUDA code at 1.4° resolution runs on a parallel supercomputer, which is equipped with nodes featuring Intel(R) Xeon(R) Platinum 8255C CPUs with 16 cores at 2.5 GHz, along with NVIDIA Tesla V100 GPUs. Each GPU has 5120 CUDA cores and 32 GB of HBM2 memory.

5.2. Acceleration performance

5.2.1. Performance comparison between 1D HIP-RRTMG_SW and 1D CUDA-RRTMG_SW

We compared the performance of the 1D HIP version and CUDA version of RRTMG_SW. HIP-RRTMG_SW was executed on the ORISE supercomputer, while CUDA-RRTMG_SW was run on a supercomputer equipped with NVIDIA Tesla V100 GPUs. 1D HIP-RRTMG_SW and CUDA-RRTMG_SW both used the 1.4° resolution, with parallel processing of 1024 horizontal columns each time, that is, *ncol* was 1024. The block size for HIP-RRTMG_SW was 64, while the one for CUDA-RRTMG_SW was 32. Table 1 shows the performance comparison of HIP-RRTMG_SW and CUDA-RRTMG_SW running on four AMD GPUs and four Tesla V100 GPUs, respectively, for a model day.

As can be seen from Table 1, CUDA-RRTMG_SW on Tesla V100 GPUs performs significantly better than HIP-RRTMG_SW, especially in terms of computation speed, where CUDA-RRTM-G_SW demonstrates higher efficiency across several subroutines. However, even though HIP-RRTMG_SW's performance does not surpass CUDA-RRTMG_SW on Tesla V100 GPUs, a comparison of AMD and NVIDIA GPUs hardware reveals that AMD GPUs still offer a better cost-performance ratio in some respects.

First, AMD GPUs generally provide better cost-effectiveness, delivering higher computational performance at a lower purchase cost. With the same budget, using multiple AMD GPUs may result in greater overall

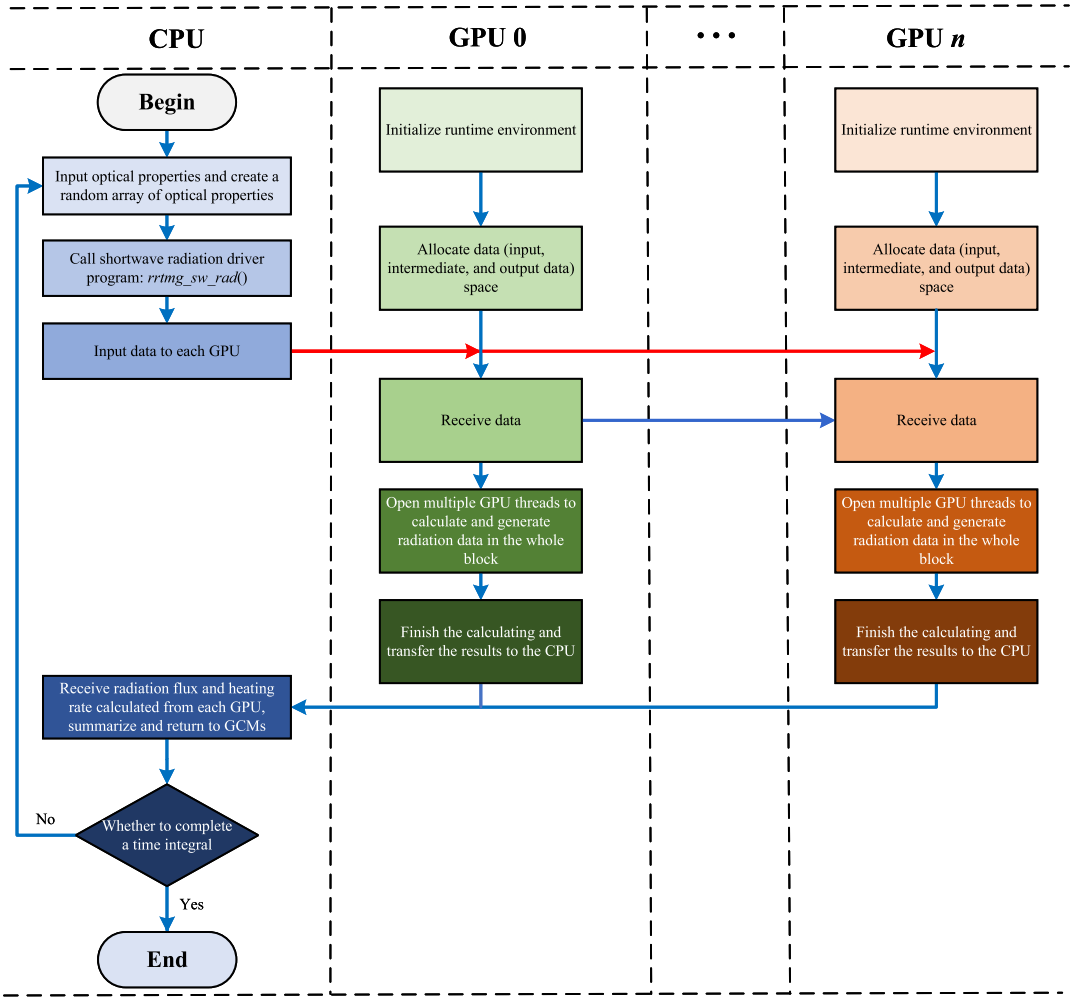


Fig. 5. The multi-GPU computing flow chart of RRTMG_SW.

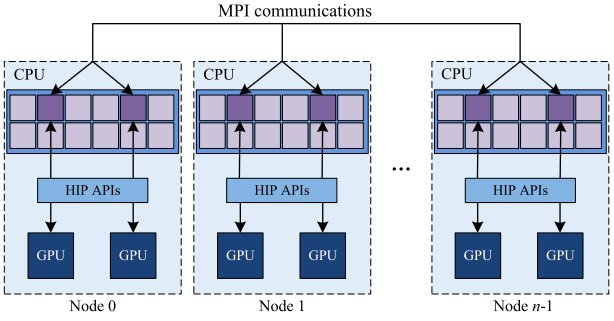


Fig. 6. "MPI + HIP" hybrid programming model.

Table 1
Performance Comparison of 1D CUDA-RRTMG_SW and HIP-RRTMG_SW (*ncol* = 1024).

Subroutines	CUDA-RRTMG_SW (s)	HIP-RRTMG_SW (s)	Speedup
inatm_sw	2.9360	2.3920	1.28
cldprmc_sw	2.7665	1.8292	1.51
setcoef_sw	0.0726	1.8468	0.04
spcvmc_sw	16.9193	24.9515	0.68
Memcpy	0.0669	0.0224	—
rrtmg_sw	22.7613	31.0419	0.73

computational power compared to a single NVIDIA GPU. Furthermore, AMD GPUs excel in energy efficiency, performing high-intensity computations at lower power consumption. Therefore, while HIP-RRTMG_SW did not outperform CUDA-RRTMG_SW on Tesla V100 GPUs, AMD GPUs still demonstrate clear advantages in terms of performance, cost, and energy efficiency, making them particularly suitable for large-scale deployments and efficient computing applications.

Table 2
Performance Comparison of 1D and 3D HIP-RRTMG_SW (*ncol* = 1024).

Subroutines	1D time (s)	3D time (s)	Speedup
inatm_sw	2.9418	0.3701	7.95
cldprmc_sw	1.9377	0.2487	7.79
setcoef_sw	1.8585	0.1417	13.12
spcvmc_sw	25.3433	10.3508	2.45
Memcpy	0.0235	0.1270	—
rrtmg_sw	32.1048	11.2383	2.86

5.2.2. Comparing 1D HIP-RRTMG_SW to its 3D version

We conducted a set of comparative experiments for the algorithm. The experimental content is the performance comparison test of a 1D parallel algorithm and 3D acceleration algorithm in CAS-ESM with 1.4° resolution. In order to demonstrate the efficient parallelism of the algorithm, we selected *ncol* value of 1024 and carried out experimental tests for one model day. The experimental results are shown in Table 2.

Table 2 experimental results show that the decoupled algorithm is 2.86× faster than the 1D acceleration algorithm. This further demon-

Table 3
The performance of 3D HIP-RRTMG_SW at 1.4° resolution.

Subroutines	Fortran time (s) (32 CPU cores)	*ncol* = 512 (s)	*ncol* = 1024 (s)	*ncol* = 2048 (s)	*ncol* = 4096 (s)	*ncol* = 8192 (s)
inatm_sw	3.1152	0.4517	0.3701	0.3215	0.3202	0.2861
cldprmc_sw	0.5710	0.4424	0.2487	0.1608	0.1279	0.0971
setcoef_sw	0.7035	0.2143	0.1417	0.1132	0.1245	0.1158
spcvmc_sw	15.6696	19.7445	10.3508	5.4890	3.2280	2.2612
HtoD	—-	0.0667	0.1264	0.2603	0.5036	1.0685
DtoH	—-	0.0004	0.0006	0.0008	0.0014	0.0026
rrtmg_sw	20.0593	20.9200	11.2383	6.3456	4.3056	3.8313
Speedup	—-	0.96	1.78	3.16	4.66	5.24

Table 4
The performance of 3D HIP-RRTMG_SW at 0.25° resolution.

Subroutines	Fortran time (s) (128 CPU cores)	*ncol* = 864 (s)	*ncol* = 1728 (s)	*ncol* = 3456 (s)	*ncol* = 6912 (s)	*ncol* = 13824 (s)
inatm_sw	9.7468	2.5806	2.0740	1.9201	1.8386	2.0863
cldprmc_sw	2.7597	1.9787	1.1716	0.8412	0.6425	0.5399
setcoef_sw	3.6824	1.1228	0.7741	0.7346	0.7108	0.7068
spcvmc_sw	106.2512	80.9633	42.4940	24.8559	16.2445	12.3063
HtoD	—-	0.1105	0.2175	0.4286	0.8672	1.7243
DtoH	—-	0.0006	0.0008	0.0014	0.0025	0.0047
rrtmg_sw	122.4401	86.7565	46.7320	28.7818	20.3061	17.3683
Speedup	—-	1.41	2.62	4.25	6.03	7.05

strates the necessity of multi-dimensional parallelism and the effectiveness of algorithm optimization.

5.2.3. With varying values of ncol

HIP-RRTMG_SW, a 3D parallel algorithm for shortwave radiation, is experimentally tested in CAS-ESM with a resolution of 1.4°. The integration step for the radiation transfer model is set to one hour, and the number per hour is 128×256, so the total number of horizontal grid cells in a model day experiment is 128×256×24, and the amount of horizontal grid data calculated for each parallel calculation is *ncol*. A node with four CPU cores and four AMD GPUs is used in the run script file, with each thread block is set to 256. The effects of different horizontal column numbers *ncol* on parallel acceleration are observed, as well as the comparison with RRTMG_SW algorithm. As there are 32 CPU cores and four AMD GPUs in a node, the comparison between the RRTMG_SW Fortran algorithm using 32 CPU cores on a node and HIP-RRTMG_SW using four CPU cores and four AMD GPUs.

We can conclude from the experimental results in Table 3 that: (1) When *ncol* = 8192, the acceleration effect is most obvious. Using four AMD GPUs and four CPU cores, HIP-RRTMG_SW accelerated 5.24× faster than RRTMG_SW running on 32 CPU cores. (2) As *ncol* increases exponentially, the running time of *inatm_sw* function does not show exponential acceleration, mainly because of the large memory access proportion of *inatm_sw*. *cldprmc_sw* and *setcoef_sw* show no significant regular changes. Similarly, the memory access time of these two kernel functions is larger than the computation time, but the memory access time is less than *inatm_sw*. (3) The function *spcvmc_sw* is mainly calculated for the two-stream radiation equation, so it shows a significant regular acceleration effect with different *ncol*.

5.3. Scalability

5.3.1. With varying values of ncol

In order to verify that the algorithm can adapt to different system configurations, we also adopt the configuration of 0.25° resolution to test the algorithm, which reflects the scalability of the algorithm. The amount of data at this resolution is so large that a minimum of 16 processes must be used to drive the entire system pattern. The experimental operation is configured with four nodes, and each node applies for four CPU cores and four AMD GPU acceleration cards. Similarly, we test how the acceleration performance varied with *ncol*. The block size is set to 256, and one model day of experimental tests is carried out.

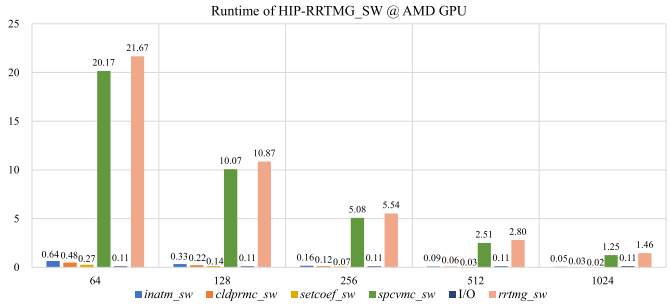


Fig. 7. The runtime of the kernels on multiple AMD GPUs.

From the experimental results in Table 4, we observe the following: (1) When *ncol* = 13824, compared to Fortran code using 128 CPU cores, it is 7.05× faster. (2) When *ncol* = 6912, the acceleration effect is relatively good. (3) When *ncol* exceeds 3456, the acceleration effect of *spcvmc_sw* increases but not exponentially (the *spcvmc_sw* kernel function is mainly used for calculation, less access time, and high stability, so it is an example of high precision). The main reason for this phenomenon is that when the number of cores exceeds 4096, the extra cores need to be executed sequentially.

5.3.2. With varying amounts of AMD GPUs

In order to demonstrate that the algorithm can not only adapt to various resolutions, but also be used for large-scale calculations with the multi-node and multi-GPU algorithm, we carry out test experiments on different AMD GPUs. We fix *ncol* = 864, the block size is set to 256, and we conduct performance tests on HIP-RRTMG_SW for one model day using different number of AMD GPUs.

According to the experimental results in Fig. 7, the algorithm has high scalability. In addition, we compare the multi-GPU acceleration using the algorithm. Fig. 8 shows that with 64 AMD GPU accelerators as a control group, the various functions and HIP-RRTMG_SW increase exponentially with the number of AMD GPUs, and their accelerations increase accordingly. Fig. 9 shows the acceleration ratio of HIP-RRTMG_SW running on AMD GPUs versus the Fortran version of RRTMG_SW running on a single CPU core and 128 CPU cores. At *ncol* = 864, HIP-RRTMG_SW using 64 AMD GPUs is 723.20× faster than RRTMG_SW with a single CPU core and 5.65× faster than the one with 128 CPU cores. Moreover, HIP-RRTMG_SW using 1024 AMD GPUs is

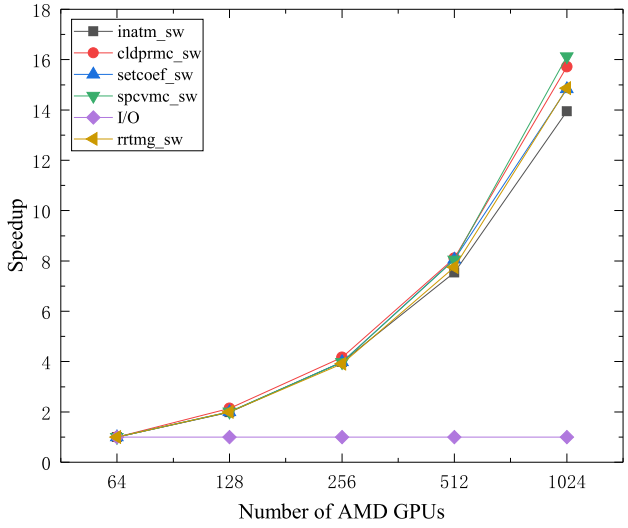


Fig. 8. The acceleration performance of HIP-RRTMG_SW on multiple AMD GPUs compared to 64 AMD GPUs.

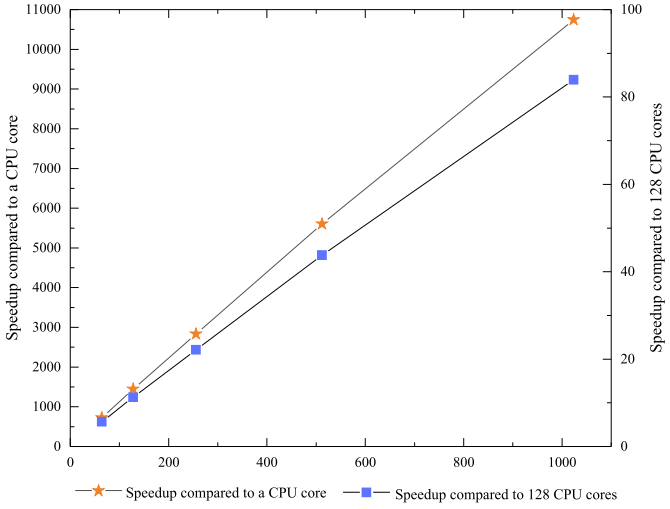
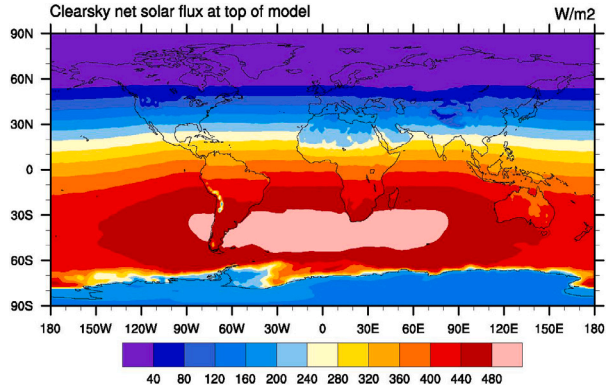


Fig. 9. The acceleration performance of HIP-RRTMG_SW on multiple AMD GPUs compared to one CPU core and 128 CPU cores.

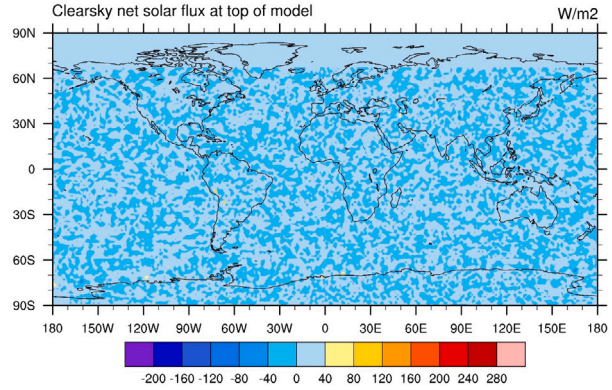
83.94× faster than the one using 128 CPU cores. The above results show that HIP-RRTMG_SW has strong scalability.

5.4. Error analysis

The HIP-RRTMG_SW heterogeneous algorithm is used in ideal simulation experiments to calculate solar radiation flux in CAS-ESM. Fig. 10 (a) shows the shortwave radiation flux simulation chart of CAS-ESM running on AMD GPU. Fig. 10 (b) shows the radiation flux simulation error distribution between RRTMG_SW and HIP-RRTMG_SW. The global average difference is only -0.000456 W/m^2, which is within the acceptable range. The reasons for the deviation include two aspects. (1) The specific implementation of mathematical functions may vary in different programming environments. Some calculations are sensitive to floating-point precision, so quite small differences in numerical representation can lead to slight discrepancies in the calculation results. However, these differences are a natural feature of floating point operations on different computing architectures. (2) There are differences in precision and rounding behavior between CPUs and AMD GPUs, which can contribute to slight numerical discrepancies in results.



(a) CAS-ESM RRTMG_SW radiative flux on AMD GPUs.



(b) Error map of radiative flux simulation between HIP-RRTMG_SW and RRTMG_SW.

Fig. 10. Comparison of RRTMG_SW radiative flux and error analysis on AMD GPUs.

5.5. Arithmetic intensity analysis

We analyzed the arithmetic intensity of kernel functions in 3D HIP-RRTMG_SW to assess whether their performance is limited by bandwidth or compute capacity. The test was conducted at 1.4° resolution, and *ncol* was set to 4096. Arithmetic intensity is defined as the number of floating-point operations (FLOPs) performed per byte of data transferred. The results show:

(1) *inatm_sw* has an arithmetic intensity of 0.0020 FLOPs/Byte, which is very low, indicating that its performance is entirely bandwidth-limited.
(2) *cldprmc_sw* has an arithmetic intensity of 0.5943 FLOPs/Byte, which, while higher than *inatm_sw*, is still significantly below 1, suggesting that its performance is mainly limited by memory bandwidth.
(3) *setcoef_sw* has an arithmetic intensity of 0.9643 FLOPs/Byte, close to 1, indicating that its performance is influenced both by memory bandwidth and compute capacity, with both factors playing a similar role.
(4) *spcvmc_sw* has an arithmetic intensity of 0.4546 FLOPs/Byte, which again points to memory bandwidth as the main limiting factor.

Overall, the kernel functions in 3D HIP-RRTMG_SW have low arithmetic intensities, with most functions being primarily bandwidth-limited. Future optimization work should focus on reducing memory access, increasing data reuse, and better utilizing the computational capacity of the GPU to improve overall performance.

5.6. Discussion

Wang et al. propose a high resolution LASG/IAP climate system ocean model HIP C parallel acceleration scheme that can adapt to CAS-ESM [18]. After 384 AMD GPUs are used for acceleration, the acceleration effect reaches 42× compared with Fortran code with the same number of CPU cores. Compared with their algorithm, the algorithm of the paper is also suitable for high-resolution models and can be used in conjunction with high-resolution ocean models, which undoubtedly enables the various models in CAS-ESM to simulate earth system conditions better and faster.

Theoretically, HIP-RRTMG_SW can run on more nodes and AMD GPUs. Due to the shortage of hardware resources, we can not use a large number of nodes for experimental testing, which is a limitation of our study. If there are no resource constraints and the resources in a warp are fully utilized, we take $ncol = 64$, then we can use up to $128 \times 256/64 = 512$ AMD GPUs in low resolution model and up to $1152 \times 768/64 = 13824$ AMD GPUs in a high resolution model.

6. Conclusion and future work

Large-scale numerical simulation of physical processes needs to be combined with HPC to meet its development trend. The continuous development of NVIDIA CUDA and the rapid rise of AMD ROCm provide us with abundant computing resources. The main differences from previous research work are: (1) The detailed modifications are taken according to the characteristics of the HIP framework, which is described in Section 3.1. (2) Unlike Li's work [30], HIP-RRTMG_SW is coupled into CAS-ESM with different resolutions. (3) A large number of AMD GPUs are used for experimental verification, further demonstrating that the proposed algorithm can achieve large-scale heterogeneous computing. Therefore, this paper proposes a large-scale shortwave radiation heterogeneous algorithm based on AMD GPUs, combined with the "MPI+HIP" hybrid programming model to achieve maximum optimization. Two different resolution CAS-ESMs are selected in this paper to experimentally verify HIP-RRTMG_SW, which ultimately run successfully in model systems. The experimental results show that the multi-AMD GPU acceleration algorithm is effective and scalable. In a model day of high resolution climate simulation, a HIP-RRTMG_SW algorithm using 16 AMD GPUs achieved a speedup of 7.05× with AMD GPUs compared to RRTMG_SW using 128 CPU cores.

The future work in this field mainly includes four aspects: (1) Further optimizing memory access modes can improve memory access efficiency when reading data from global memory. Making full use of registers and other types of memory can improve the computing performance. (2) In the large-scale parallel computation of shortwave radiation physical processes based on "MPI + HIP", the cooperative computation of CPU and GPU should be considered and the computational load should be reasonably divided. On the one hand, the performance of a heterogeneous computing system can be fully exploited. On the other hand, the computational efficiency of parallel algorithms can be further improved. (3) In order to give full play to the advantages of semi-precision computing in GPUs, it is necessary to study mixed-precision computing methods for radiation physics processes. (4) Further porting more components of CAS-ESM to the GPU to reduce frequent data transfers between the GPU and CPU. This will help avoid the overhead of host/device copies, maximizing the benefits of GPU acceleration and improving overall performance.

CRediT authorship contribution statement

Zhenzhen Wang: Writing – original draft, Software, Methodology. **Yuzhu Wang:** Writing – review & editing, Supervision, Project administration, Methodology, Conceptualization. **Fei Li:** Writing – review & editing. **Jinrong Jiang:** Writing – review & editing. **Xiaocong Wang:** Writing – review & editing, Validation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported in part by National Science and Technology Major Project of China under Grant 2024ZD1001900, in part by National Key Research and Development Program of China under Grant 2016YFB0200800, in part by National Key Scientific and Technological Infrastructure project "Earth System Science Numerical Simulator Facility" (EarthLab), and in part by the GHFUND D of China under Grant ghfund202407045842. The numerical calculations in this study were carried out on the ORISE supercomputer.

Data availability

Data will be made available on request.

References

[1] Z. Jin, Z. Lu, H. Li, X. Chi, J. Sun, Origin of high performance computing-current status and developments of scientific computing applications, Bull. Chin. Acad. Sci. 34 (6) (2019) 625–639.
[2] D. Chen, X. Li, L. Wang, S.U. Khan, J. Wang, K. Zeng, C. Cai, Fast and scalable multi-way analysis of massive neural data, IEEE Trans. Comput. 64 (3) (2014) 707–719.
[3] CUDA C Programming Guide v10.0, 2019, [Online]. Available: https://docs.nvidia.com/pdf/CUDA_C_Programming_Guide.pdf.
[4] HIP Programming Guide, 2021, [Online]. Available: https://rocmdocs.amd.com/en/latest/Programming_Guides/HIP-GUIDE.html.
[5] W. Xue, C. Yang, H. Fu, X. Wang, Y. Xu, J. Liao, Y. Lu, R. Ranjan, L. Wang, Ultra-scalable CPU-MIC acceleration of mesoscale atmospheric modeling on Tianhe-2, IEEE Trans. Comput. 64 (8) (2015) 2382–2393.
[6] C. Yang, et al., 10M-core scalable fully-implicit solver for nonhydrostatic atmospheric dynamics, in: International Conference for High Performance Computing, Networking, Storage and Analysis, SC'16, 2016.
[7] H. Fu, J. Liao, N. Ding, X. Duan, L. Gan, Y. Liang, X. Wang, J. Yang, Y. Zhang, W. Liu, L. Wang, G. Yang, Redesigning CAM-SE for peta-scale climate modeling performance and ultra-high resolution on Sunway TaihuLight, in: International Conference for High Performance Computing, Networking, Storage and Analysis, SC'17, 2017.
[8] J. Xiao, S. Li, B. Wu, H. Zhang, K. Li, E. Yao, Y. Zhang, G. Tan, Communication-avoiding for dynamical core of atmospheric general circulation model, in: 47th International Conference on Parallel Processing, 2018.
[9] H. Cao, L. Yuan, H. Zhang, B. Wu, S. Li, P. Lu, Y. Zhang, Y. Xu, M. Zhang, A highly efficient dynamical core of atmospheric general circulation model based on leap-format, in: 2020 IEEE International Parallel and Distributed Processing Symposium (IPDPS), 2020.
[10] H. Zhang, M. Zhang, Q. Zeng, Sensitivity of simulated climate to two atmospheric models: interpretation of differences between dry models and moist models, Mon. Weather Rev. 141 (5) (2013) 1558–1576.
[11] M.J. Iacono, Enhancing cloud radiative processes and radiation efficiency in the advanced research weather research and forecasting (WRF) model, in: Atmospheric and Environmental Research, 2015.
[12] M.J. Iacono, J.S. Delamere, E.J. Mlawer, M.W. Shephard, S.A. Clough, W.D. Collins, Radiative forcing by long-lived greenhouse gases: calculations with the AER radiative transfer models, J. Geophys. Res., Atmos. (2008).
[13] E. Price, J. Mielikainen, M. Huang, B. Huang, H.L.A. Huang, T. Lee, GPU-accelerated longwave radiation scheme of the Rapid Radiative Transfer Model for General Circulation Models (RRTMG), IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 7 (8) (2014) 3660–3667.
[14] J. Michalakes, M.J. Iacono, E.R. Jessup, Optimizing weather model radiative transfer physics for Intel's Many Integrated Core (MIC) architecture, Parallel Process. Lett. 26 (4) (2016).
[15] N. Kondratyuk, V. Nikolskiy, D. Pavlov, V. Stegailov, GPU-accelerated molecular dynamics: state-of-art software performance and porting from Nvidia CUDA to AMD HIP, Int. J. High Perform. Comput. Appl. 35 (4) (2021) 312–324.
[16] E. Kuznetsov, V. Stegailov, Porting CUDA-based molecular dynamics algorithms to AMD ROCm platform using hip framework: performance analysis, in: Russian Supercomputing Day, Springer, Cham, 2019.
[17] ROCm documentation, 2021, [Online]. Available: https://rocmdocs.amd.com/en/latest/.
[18] P. Wang, et al., The GPU version of LASG/IAP Climate System Ocean Model version 3 (LICOM3) under the heterogeneous-compute interface for portability (HIP) framework and its large-scale application, Geosci. Model Dev. 14 (5) (2021) 2781–2799.

[19] A. Borisov, Performance evaluation of Kuznyechik and Magma symmetric ciphers on AMD and NVIDIA GPUs, in: 2020 Ural Symposium on Biomedical Engineering, Radioelectronics and Information Technology (USBEREIT), 2020.
[20] A. Borisov, E. Myasnikov, Dimensionality reduction using GPU-accelerated gradient descent, in: CEUR Workshop Proceedings, 2020.
[21] B. Jang, D. Schaa, P. Mistry, D. Kaeli, Exploiting memory access patterns to improve memory performance in data-parallel architectures, IEEE Trans. Parallel Distrib. Syst. 22 (1) (2010) 105–118.
[22] T.T. Dao, J. Kim, S. Seo, B. Egger, J. Lee, A performance model for GPUs with caches, IEEE Trans. Parallel Distrib. Syst. 26 (7) (2014) 1800–1813.
[23] C. Blakeney, X. Li, Y. Yan, Z. Zong, Parallel blockwise knowledge distillation for deep neural network compression, IEEE Trans. Parallel Distrib. Syst. 32 (7) (2020) 1765–1776.
[24] G.Y. Shi, Atmospheric Radiology, Science Press, 2007.
[25] H. Yin, Fundamentals of atmospheric radiology, China Meteorological Press, 1993.
[26] J. Mielikainen, B. Huang, H.L.A. Huang, M.D. Goldberg, GPU acceleration of the updated Goddard shortwave radiation scheme in the weather research and forecasting (WRF) model, IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 5 (2) (2012) 555–562.
[27] G.Y. Shi, On the k-distribution and correlated k-distribution models in the atmospheric radiation calculations, Chin. J. Atmos. Sci. 22 (4) (1998) 659–676.
[28] Z. Wang, Y. Wang, X. Wang, F. Li, C. Zhou, H. Hu, J. Jiang, GPU-RRTMG_SW: accelerating a shortwave radiative transfer scheme on GPU, IEEE Access 9 (2021) 84231–84240.
[29] HIPify Programming Guide, 2021, [Online]. Available: https://rocmdocs.amd.com/en/latest/Current_Release_Notes/Current-Release-Notes.html?highlight=hipify#hipify-documentation.
[30] F. Li, Y. Wang, Z. Wang, X. Ji, J. Jiang, X. Tang, H. Zhang, CC-RRTMG_SW++: further optimizing a shortwave radiative transfer scheme on GPU, J. Supercomput. 78 (15) (2022) 17378–17402.



Zhenzhen Wang received the M.Eng. degree in computer technology from School of Information Engineering, China University of Geosciences, Beijing, in 2022. She is currently a software engineer at Dawning Information Industry Co., LTD. Her research interests include high performance computing and parallel algorithm.



Yuzhu Wang received the Ph.D. degree in engineering from University of Chinese Academy of Sciences, in 2015. He is an associate professor at School of Information Engineering, China University of Geosciences, Beijing, China. His research interests include high performance computing and parallel algorithm.



Fei Li received the M.Eng. degree in computer science from School of Information Engineering, China University of Geosciences, Beijing, in 2024. He is currently pursuing the Ph.D. degree in computer science from School of Computer Science and Technology, Xi'an Jiaotong University, Xi'an, China. His research interests include high performance computing and parallel algorithm.



Jinrong Jiang received the Ph.D. degree in engineering from Chinese Academy of Sciences, in 2007. He is a full professor at Computer Network Information Center, Chinese Academy of Sciences, Beijing, China. His research interests include parallel algorithm and high performance computing.



Xiaocong Wang received the Ph.D. degree in meteorology from Chinese Academy of Sciences, in 2012. He is an associate professor at Institute of Atmospheric Physics, Chinese Academy of Sciences, Beijing, China. His research interests include parameterizations of cumulus convection and cloud radiation in climate models.