PROCEEDINGS OF SCIENCE

Parallelism of MRT Lattice Boltzmann Method based on Multi-GPUs

Liankai Yao[1]
School of Information Engineering, China University of Geosciences (Beijing)
Beijing, 100083, China
E-mail: `Yaolk1119@icloud.com`

Ailan Wang
School of Information Engineering, China University of Geosciences (Beijing)
Beijing, 100083, China
E-mail: `alwang@cugb.edu.cn`

Xiaohui Ji[2 3]
School of Information Engineering, China University of Geosciences (Beijing)
Beijing, 100083, China
E-mail: `xhji@cugb.edu.cn`

In order to accelerate the Multi-Relaxation-Time Lattice Boltzmann Method (MRT-LBM), it was parallelized on multi-GPUs located in one computer. Lattices of the MRT-LBM were distributed to the different GPUs and they were operated on the GPU threads concurrently. In the step streaming, in order to get the neighbour lattice information, all lattices were combined and operated on the master GPU. Three-dimensional (3D) groundwater flow was simulated by using the parallelized MRT-LBM and the experimental results showed that almost 95 times acceleration was attained on a six GPUs computer. The parallelized MRT-LBM based on the multi-GPUs located in one computer is efficient in terms of both time and energy.

CENet2017
22-23 July 2017
Shanghai, China

[1] Speaker
[2] Corresponding Author
[3] This research was supported by the National Natural Science Foundation of China (41102158) and the Fundamental Research Funds for the Central Universities of China.

1.Introduction

As an intermediate mesoscopic method connecting the macroscopic method with the microcosmic method, Lattice Boltzmann Method (LBM) features the advantages of simple algorithm steps, easy lattice partitions, good parallelism and tractable boundary conditions[1-2]. In LBM, Multi-Relaxation-Time (MRT) model is popular because it highlights high precision, good convergence, high stability, short computation time and many manageable parameters[3-7]. So MRT-LBM has been applied to many 3D flow problems[3-7].

The more lattices the MRT-LBM has, the higher precision it can provide and needs more calculation time. To accelerate the computation, the parallel computing has been used to simulate various flows based on MRT-LBM[8-14]. Most of the current researches either focus on the single GPU[8-10] or on the GPU clusters[11-14]. In this paper, we parallelize the MRT-LBM on multi-GPUs located on one computer, which can make MRT-LBM efficient in terms of both time and energy. A 3D groundwater flow was simulated in the experiments to test the correctness and the efficiency.

The structure of this paper is as follows: in Section 2, we describe the MRT-LBM model and the two parallel architectures. In Section 3, we parallelize the MRT-LBM model on GPUs. In Section 4, a 3D groundwater flow is simulated to test our parallelized MRT-LBM model.

2.Background

The MRT-LBM model and the two parallel architectures adopted herein are introduced in this section.

2.1MRT Lattice Boltzmann Method

The process of solving partial differential equation (PDE) by using MRT-LBM is divided into seven steps as follows. In this paper, D3Q19 model was used[15].

1. Discretization. The 3D area is discretized into lattices. Each lattice owns 19 distribution functions in different directions.

2. Initialization. The initial distribution functions are usually set to its equilibrium distribution function (EDF) which is described as

$$f_i^{(eq)}=\omega_i\rho[1+(c_i\cdot u)/(c_s^2)+(c_i\cdot u)^2/(2c_s^4)-u^2/(2c_s^2)] \tag{2.1}$$

where c_s is regarded as the sound velocity, ω_i is the weighting factor, c is lattice velocity and ρ is the macroscopic value.

3. Collision. Collision is executed based on Lattice Boltzmann Equation(LBE) which is formulated as that in Equation (2).

$$f_i(x+e_i\delta_t,t+\delta_t)-f_i(x,t)=-M_{ij}^{-1}S_{ij}(m_j-m_j^{(eq)}) \tag{2.2}$$

In Equation (2), S is a diagonal matrix that can be written as $S=diag(0,s_e,s_\varepsilon,0,s_q,0,s_q,0,s_q,s_v,s_\pi,s_v,s_\pi,s_v,s_v,s_v,s_t,s_t,s_t)$, where $s_e=1.19$, $s_\varepsilon=s_\pi=1.4$, $s_t=1.98$ and $s_v=1/\tau$, here τ is determined by the initial conditions of the simulated problem; m and $m^{(eq)}$ are the moment spaces of the distribution function and its EDF, which can be formulated as $m=M\cdot f$ and $m^{(eq)}=M\cdot f^{(eq)}$; M is the the transformation matrix and can be written as

4. Streaming. Lattice streaming can be formulated as

$$f_i(x+c_i\delta_t, t+\delta_t)=f_i^{'}(x,t) \tag{2.3}$$

5. The computing of macroscopic values. The macroscopic values are computed based on

$$\rho=\sum_{(i=0)}^{18} f_i \tag{2.4}$$

6. Boundary condition. The boundary condition directly influences the results of MRT-LBM and it deals with the distribution functions that are not achieved after collision and streaming.

7. Convergence judgment. If the errors between the current macroscopic values and the previous ones are small enough, we judge the system achieves a stable condition.

2.2Parallel Architecture

Many parallel architectures like CUDA[16], MPI[17] and OpenMP[18] have been applied to accelerate the calculations. In this paper, CUDA and OpenMP made use of the powerful calculating capabilities of the GPUs.

2.2.1CUDA Architecture

CUDA published by NVIDIA can implement calculations on computational General Purpose GPUs. After copying data from CPU (host) to GPU (device), the kernel functions implement the primary algorithms run on devices.

A kernel function maps to a grid (__global__ function) on a GPU. A grid can access the global memory of a GPU. When a kernel function is called, CUDA allocates a lot of blocks and each block has many threads. A thread owns a register and the local memory, and the threads in the same block have a shared memory.

2.2.2OpenMP Architecture

OpenMP supports the shared memory parallel programming in a computer. It can be used to start the CPUs working concurrently first and then each CPU core starts with each GPU. The multi-GPUs in a computer can work concurrently.

3.GPU Parallelism of MRT-LBM

To accelerate the MRT Lattice Boltzmann Method, we parallelize it on GPUs by using CUDA and OpenMP.

3.1 Parallelism of MRT-LBM on a GPU

Our method for parallelizing MRT-LBM on a GPU is shown in Figure1. Seven steps of each MRT-LBM lattice shown in 2.1 are operated on each GPU thread. So all the lattices can be computed concurrently. According to the number of lattices, the number of blocks and threads in a GPU were set as follows,

```
dim3 threads(num_threads, 1, 1)
dim3 blocks(Nx*Nz/num_threads, Ny)
```

as that of Jonas Tolke[19]. Here num_threads means the number of threads in one block, 16 was set in the paper, and Nx, Ny and Nz mean the number of lattices in x, y and z directions.

On the GPU, the lattice data are all stored in the global memory to make all threads access the data. When lattices start to stream in Step 4, the current lattices need to read all neighbour lattices, even the neighbour lattices may be in different blocks. Other data like the constant data are stored in the shared memory to accelerate the read speed for threads in the same blocks because the threads in the same blocks can access the data in the shared memory in a faster speed.

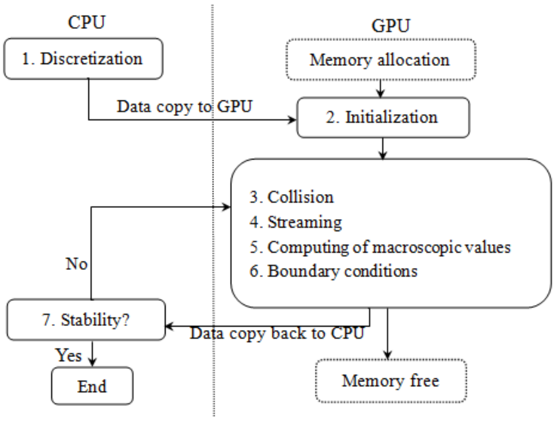


Figure 1: Pallelism of MRT-LBM on a GPU

Some basic linear algebraic operations like the matrix vector product were implemented by using the CUBLAS library provided by CUDA.

3.2Parallelism of MRT-LBM on Multi-GPUs

With the growth of the number of lattices, the number of threads in a GPU is not enough to compute all the lattices concurrently; so we compute all the lattices on multi-GPUs. In this paper, OpenMP is used to implement collaboration and communication on multi-GPUs. First, the CPUs starte to work concurrently by using OpenMP and then each CPU core starts each GPU. The parallelism of MRT-LBM on multi-GPUs are shown in Figure 2.

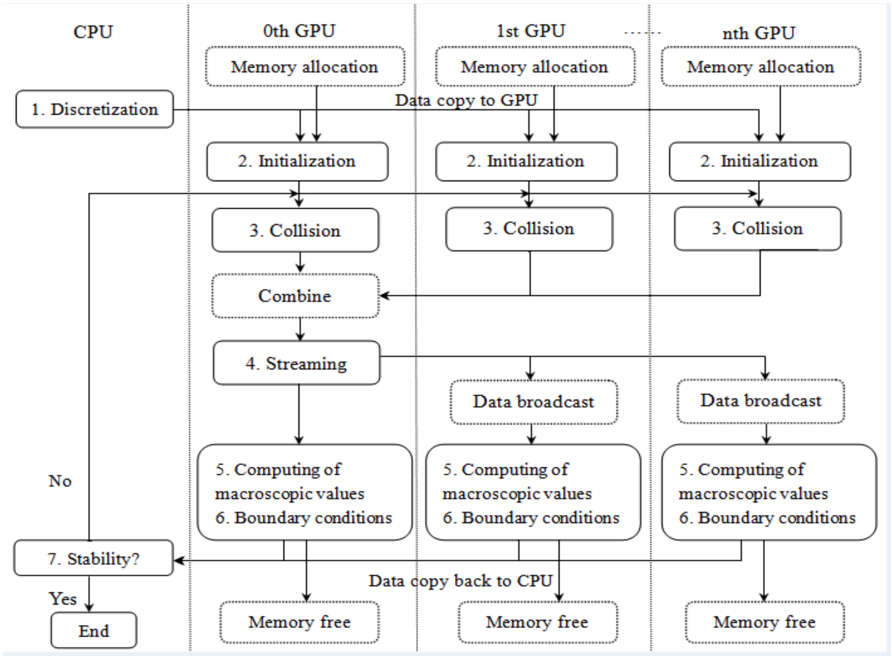


Figure 2: Parallelism of MRT-LBM on Multi-GPUs

The lattices are equally divided into the GPUs. For the number of lattices $Nx \times Ny \times Nz$, the number of GPUs *num_GPU,* each GPU will compute $Nx \times Ny \times Nz / num\_GPU$ lattices, and the number of threads in each GPU will decrease to num_threads/num_GPU from num_threads compared with that by using one GPU only.

As described in 3.1, in Step 4 streaming, the current lattices need to read all neighbour lattices, even the neighbour lattices may be in different GPUs. So in step 4, lattices in different GPUs are combined together and the streaming is implemented in the master GPU as shown in Figure 2. After streaming, the updated distribution functions are broadcasted to other GPUs.

4. Experiments on 3D Groundwater Flow Simulation

To test the correctness and the efficiency of our methods, a 3D groundwater flow was simulated. The experiments are conducted on the server with 2 x Intel Xeon CPU E5-2695 v2@2.40GHz and 6 x Nvidia Tesla GPU K40m.

Under the condition of not considering the variation in water density, the 3D groundwater flow in porous media can be described as

$$\partial / \partial x \left(K_{xx} \partial h / \partial x \right) + \partial / \partial y \left(K_{yy} \partial h / \partial y \right) + \partial / \partial z \left(K_{zz} \partial h / \partial z \right) - W = S_s \partial h / \partial t \tag{4.1}$$

where K_{xx} , K_{yy} and K_{zz} are the weights of permeation coefficient in *x*, *y* and *z* directions. *h* means the waterhead, W means the flux of unit volume and S_s means the storativity.

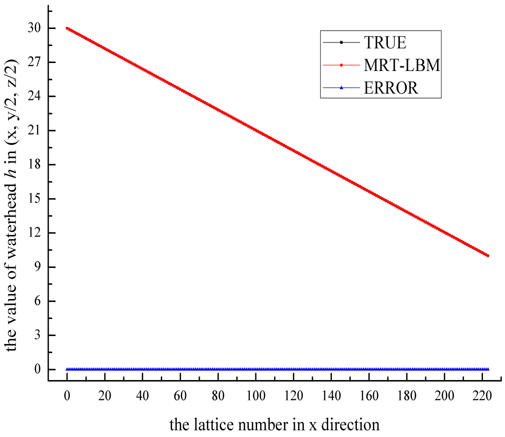
We simulated a 3D groundwater field with the size of 700*600*120 and whose hydraulic values are listed in Table 1. The water heads *h* in the left boundary and the right boundary are constant 30 and 10 initially.

Variable name	Physical Meaning	Value
K	permeation coefficient	0.5
Ss	storativity	0.00001
h	waterhead	10 (right boundary) or 30 (others)

Table 1: Macroscopical Values in Equation (5)

The experimental results are shown in Figure 3 and 4. In Figure 3, values in *(x, y/2, z/2)* were output to verify the correctness. The water heads decline from 30 to 10 linearly under the influence of left boundary and right boundary. The simulation results are consistent with the theoretical values (TRUE in Figure 3).

Figure 4 shows the speedups of the parallelism of MRT-LBM on multi-GPUs. We can see that the more of the number of lattices, the larger speedups can be achieved. Because a MRT-LBM model with more lattices needs more iterations to become stable than the model with less lattices. From Figure 1 and Figure 2, we can see that the parallelism of the MRT-LBM on GPUs save the computation time in each iteration. With the iterations grow, the time saved grows too. The speedup of using the same number of GPUs has an increasing trend with the number of lattices grow. For the same number of lattices, the more the GPUs are used, the larger speedups can be achieved because the more number of GPU threads can be used. The speedup gets a maximum about 95 for the number of lattices 896*768*4 by using 6 GPUs.



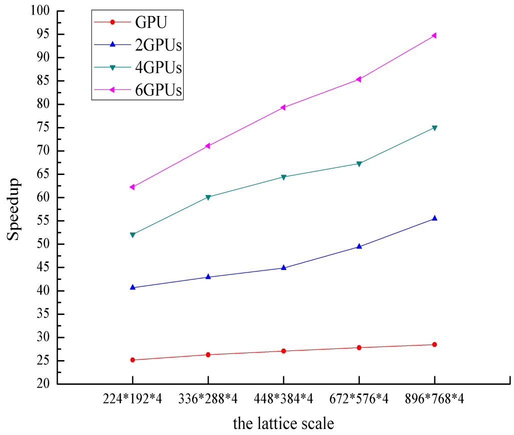


Figure 3: Numerical and Rheoretical Results **Figure 4:** Speedups of Multi-GPUs

5.Conclusion

The parallelism method of MRT-LBM on multi-GPUs located on one computer was given in the paper and the experiments on a 3D groundwater flow simulation were made. The experimental results indicate that the more the GPUs are used and the bigger the lattice number will be and the more speedups can be achieved. The maximum speedup for the number of lattices 896*768*4 on 6 GPUs is 95.

References

[1] P. L. Bhatnagar, E. P. Gross, M. Krook. *A model for collision processes in gases.* Ⅰ:Small amplitude processes in charged and neutral one-dimensional systems. Physical Review, 1954, 94(3): 511-525.

[2] U. Frisch, B. Hasslacher, Y. Pomeau. *Lattice-gas automata for the Navier-Stokes equations[J].* Physical Review Letters. 1986, 56: 1505-1508.

[3] Y. H. Li, Y. G. CHENG. *Three-dimensional simulation of water hammer wave by multiple-relaxation-time lattice Boltzmann method.* Engineering Journal of Wuhan University, 2013, 46(4): 417-422.

[4] Y. Liu, R. M. C. So, Z. X. Cui. *Bluff body flow simulation using lattice Boltzmann equation with multiple relaxation time.* Computers & Fluids, 2006, 35: 951-956.

[5] D. d'Humieres. Generalized lattice Boltzmann equations[C]. *In Rarefied gas dynamics: theory and simulations (ed. B. D. Shizgal & D. P. Weaver)*, Prog. Aeronaut. Astronaut. 1992, 159: 450-458.

[6] Y. Gao. *Using MRT lattice Boltzmann method to simulate gas flow in simplified catalyst layer for different inlet-outlet pressure ratio.* International Journal of Heat and Mass Transfer, 2015, 88: 122-132.

[7] A. Xu, T. S. Zhao, L. An, L. Shi. *A three-dimentional pseudo-potential-based lattice Boltzmann model for multiphase flows with large density ratio and variable surface tension.* International Journal of Heat and Fluid Flow, 2015, 56: 261-271.

[8] L. S. Lin, H. W. Chang, C. A. Lin. *Multi relaxation time lattice Boltzmann simulations of transition in deep 2D lid driven cavity using GPU*. Computers & Fluids, 2013, 80: 381-387.

[9] Q. L. Ren, C. L. Chan. *GPU accelerated numerical study of PCM melting process in an enclosure with internal fins using lattice Boltzmann method*. International Journal of Heat and Mass Transfer, 2016, 100: 522-535.

[10] J. Tölke& M. Krafczyk. *TeraFLOP computing on a desktop PC with GPUs for 3D CFD*. International Journal of Computational Fluid Dynamics, 2008, 22(7):443-456.

[11] P. Y. Hong, L. M. Huang, L. S. Lin, C. A. Lin. *Scalable multi-relaxation-time lattice Boltzmann simulations on multi-GPU cluster.* Computers & Fluids, 2015, 110: 1-8.

[12] H. W. Chang, P. Y. Hong, L. S. Lin, C. A. Lin. *Simulations of flow instability in three dimensional deep cavities with multi relaxation time lattice Boltzmann method on graphic processing units*. Computers & Fluids, 2013, 88: 866-871.

[13] C. Obrecht. F. Kuznik, B. Tourancheau, J.J.Roux.*Muli-GPU implementation of the lattice Boltzmann method.* Original Research Article Computers & Mathematics with Applications. 2013, 65: 252-261.

[14] Chang H W, Hong P Y, Lin L S, et al. *Simulations of Three-dimensional Cavity Flows with Multi Relaxation Time Lattice Boltzmann Method and Graphic Processing Units.* Procedia Engineering, 2013, 61:94-99.

[15] Y. Qian, D. d'Humieres, P. Lallemand. *Lattice BGK models for Navier-Stokes equation[J]*. Europhys. Lett. 1992, 17: 479-484.

[16] S. Cook. (2013). *CUDA Programming: A developer's guide to parallel computing with GPUs.* Waltham: Morgan Kaufmann.

[17] W. Gropp. *MPICH2 User''s Guide*. Mathematics & Computer Science Division Argonne National Laborary, 2004.

[18] B. Chapman, G. Jost, R. W. Pas. (2008). *Using OpenMP: portable shared memory parallel programming*. London: The MIT Press.

[19] J. Tölke. *Implementation of a Lattice Boltzmann kernel using the Compute Unified Device Architecture develop by NVIDIA.* Computing and Visualization in Science, 2010, 13(1): 29-39.