Parallel Algorithms of Multi-relaxation-time Lattice Boltzmann Method

Lei Xu¹, Pan Cheng¹², Zhixiang Liu³ and Wu Zhang¹*

¹ School of Computer Engineering and Science, Shanghai University, Shanghai, 200444, China.
² Shanghai Aircraft Design and Research Institute, Shanghai, 201210, China
³ College of Information, Shanghai Ocean University, Shanghai, 201306, China.
leixushu@t.shu.edu.cn, wzhang@shu.edu.cn

Abstract. The lattice Boltzmann method has become an attractive and promising approach in computational fluid dynamics. In this paper, the D3Q19 multi-relaxation-time lattice Boltzmann method is employed to simulate complex fluid flow and its parallel algorithm is presented including Cartesian grid generation, domain decomposition method, and data exchange strategy on clusters. Considering load balancing on large scale cluster, details of domain decomposition method are presented. The numerical results show that the presented algorithm have considerable scalability on 2048 cores and the efficiency can achieve 92.01%.

1. Introduction

In the past three decades, lattice Boltzmann method (LBM) has become an interesting and alternative method in computational fluid dynamics (CFD). LBM has several advantages especially in dealing with complex boundaries, incorporating microscopic interactions and parallelization compared with other traditional CFD methods based on the macroscopic continuum equations. The most popular collision operator is modeled by single-relaxation-time LBM (SRT-LBM) [1,2]. However, it is known that the SRT-LBM model may suffer from numerical instability when simulating fluids with relatively low viscosities [3]. A suggested method to improve numerical stability is to make use of a multiple-relaxation-time LBM (MRT-LBM) model instead of the SRT-LBM [4,5]. Because the relaxation times can be adjusted independently, the numerical stability of the MRT-LBM model can be enhanced.

As a result of the dimension of the problems treated with the LBM, high computing power and large memory space are required. Due to the local dynamics of MRT-LBM, it is an efficient way to adopt domain decomposition method. Wu and Shao simulated two dimensional incompressible steady lid-driven cavity flows by parallel LBM on 8 nodes, but they did not show the details and performance of presented parallel algorithm [6]. Schepke et al. presented and discussed a block parallel implementation of two and three-dimensional versions of SRT-LBM, the scalability and efficiency are given on 40 CPU cores [7]. Gao et al. studied the inner fluid dynamics and calculated permeability of the porous media microstructures by parallel LBM on 512 CPU cores without performance analysis [8]. Tian et al. deployed Palabos [9] on Petascale Sunway BlueLight MPP Supercomputer and presented some performance results using 1024 CPU cores [10]. In this paper, the D3Q19 MRT-LBM is adopted to simulate complex fluid flow and its parallel algorithm is presented including cartesian grid generation, domain decomposition method, and data exchange strategy on clusters.
The rest of the paper is organized as follows. In Section 2, we give a description of MRT-LBM. MRT-LBM parallel strategy is presented in Section 3. Next, the numerical results are discussed in Section 4. Finally, the major conclusions are summarized in Section 5.

2. MRT-LBM and boundary condition

2.1. MRT-LBM and boundary condition

The time evolution equation of LBM can be written as[1-3,11,12]

\[ f_i(x + e_i \delta_t, t + \delta_t) - f_i(x, t) = \Omega_i, \]

(1)

where \( f_i(x, t) \) is the particle distribution function (PDF) at site \( x \) and time \( t \), \( \Omega_i \) is the collision operator, \( \delta_t \) is the time step, \( e_i \) is the particle velocity in the \( i \) direction, and \( i = 0,1, \ldots ,18 \).

According to Eq. (1), LBM can be divided into two steps: collision and propagation,

\[ f_i(x + e_i \delta_t, t + \delta_t) = f_i(x, t) + \Omega_i, \]

(2)

\[ f_i^+(x, t) = f_i(x, t) + \Omega_i, \]

(3)

where \( f_i^+(x, t) \) denotes the post-collision state of PDF.

In three-dimensional applications of LBM, the discrete velocities \( e_i \) in the three-dimensional nineteen-velocity (D3Q19) model are defined as

\[ e = \begin{pmatrix} 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \times c, \]

(4)

where the lattice speed \( c = \frac{\delta_x}{\delta_t} \) takes 1, \( \delta_x \) is the lattice size.

The equilibrium distribution function \( f_i^{eq}(x, t) \) can be obtained by

\[ f_i^{eq} = \rho \omega_i \left[ 1 + \frac{e_i u_i}{c_s^2} + \frac{(e_i u_i)^2}{2c_s^4} - \frac{u_i^2}{2c_s^2} \right], \]

(5)

where \( c_s \) is the speed of sound \( c_s = \frac{1}{\sqrt{3}} \) and the coefficients of \( \omega_i \) are \( \omega_0 = \frac{1}{3} (i = 0), \omega_1 = \frac{1}{18} (i = 1 \ldots 6) \) and \( \omega_1 = \frac{1}{18} (i = 7 \ldots 18) \).

In the MRT-LBM, the collision operator can be expressed as

\[ \Omega = -M^{-1} \cdot S \cdot \left[ m(x, t) - m^{eq}(x, t) \right], \]

(6)

where \( m \) and \( m^{eq} \) represent the velocity moments of the PDFs \( f \) and their equilibria \( f^{eq} \) respectively. \( m \) is arranged in the following order

\[ m_i = (\rho, e, j_x, j_y, j_z, q_x, q_y, q_z, 3p_{xx}, 3\pi_{xx}, 3\pi_{uw}, \pi_{ww}, \pi_{xy}, p_{xy}, p_{xz}, p_{xz}, m_x, m_y, m_z)^T \]

(7)

where \( \rho \) is the mass density, \( e \) is energy, \( \epsilon \) is energy square, \( j_x \) is the momentum, \( q = (q_x, q_y, q_z) \) is the heat flux, \( p_{xx}, p_{ww}, p_{xy}, p_{yxy} \) and \( p_{zx}, p_{zxy} \) are the second order moment with respect to the stresses, \( \pi_{xx} \) and \( \pi_{ww} \) are the fourth-order moment with respect to the velocities, \( m_x, m_y \) and \( m_z \) are the third moments related to the velocities. The moment space \( m \) is related to the PDFs by a 19 × 19 transformation matrix \( M \) which linearly transforms the PDFs \( f \) to the velocity moments \( m \). \( S \) is a non-negative 19 × 19 diagonal relaxation matrix.

\[ S = diag(0, s_1, s_2, 0, s_4, 0, s_4, 0, s_4, 0, s_4, s_9, s_10, s_9, s_10, s_10, s_13, s_{13}, s_{13}, s_{16}, s_{16}, s_{16})^T \]

(8)

where the relaxation parameter \( s_0 \) and \( s_{13} \) satisfies

\[ v = \frac{1}{3} \left( \frac{1}{s_0} - \frac{1}{2} \right) = \frac{1}{3} \left( \frac{1}{s_{13}} - \frac{1}{2} \right). \]

(9)

\( v \) is the kinematic viscosity. \( s_1 = 1.19, s_2 = s_{10} = 1.4, s_4 = 1.2 \) and \( s_{16} = 1.98 \) are taken [5].

It can be observed from Eq. (2) that the difference between MRT and SRT is MRT needs to transform the particle distribution functions to the velocity moments consisting of two matrix transformations. The collision operator of MRT-LBM is executed locally which does not increase the communication time when executed on clusters.

The macroscopic quantities, such as the density, velocity, and pressure of the flow, are defined as follows
\[ \rho = \sum_{i=0}^{18} f_i, \quad u = \frac{1}{\rho} \sum_{i=0}^{18} f_i e_i, \quad p = \rho c_s^2. \quad (10) \]

### 2.2. Boundary Condition

In this present work, a unified boundary treatment for curved boundary is applied to model non-slip boundary condition [13]. In Figure 1, the solid circle is the solid node and the hollow circle is the fluid node, and a curved wall separates the solid nodes from the fluid nodes. The lattice node on the fluid side of the boundary is denoted as \( x_f \) and denoted as \( x_b \) on the solid side. \( x_{ff} \) is the adjacent fluid node of \( x_f \). The filled small rhombus on the boundary wall \( x_w \) denotes the intersections of the wall with various lattice links. The boundary velocity at \( x_w \) is noted as \( u_w \). Next, this curved boundary scheme is given as follows

\[
f_i(x_f, t + \delta t) = \frac{1}{1 + q} \left[ (1 - q) f_i^+ (x_{ff}, t) + q f_i^+ (x_f, t) + q f_{i\perp}^+ (x_f, t) \right], \quad (11)
\]

where \( \perp \) is the opposite direction of \( i \), and \( q \) is the fraction of an intersected link in the fluid region, that is written by

\[
q = \frac{|x_{f\perp} - x_w|}{|x_f - x_b|}, \quad 0 \leq q \leq 1. \quad (12)
\]

![Figure 1 Illustration of Curved Boundary](image)

As for the non-slip static straight wall, a non-equilibrium extrapolation method is employed [14].

### 3. MRT-LBM Parallel Strategy

In the LBM, collision operation is executed locally, and only neighboring lattice nodes are involved during the propagation operation, therefore, it is highly suitable for parallel computing and it is natural to adopt physical domain decomposition method for the implement of parallel stragety [15]. There are fluid node (‘f’), solid node (‘s’), and boundary node (‘b’) in the computational domain. In order to simulate complex geometries, it is necessary to distinguish different type of lattice firstly based on surface mesh of geometry. The framework of MRT-LBM parallel strategy is shown in Algorithm 1.

**Algorithm 1 The Framework of MRT-LBM Parallel Strategy.**

1. Read surface mesh file;
2. Domain decomposition including domain decomposition and the calculation of sub-domain range;
3. Initialize the flow field information;
4. Generate the cartesian grid including the lattice type judgement and the compute of the distance between curve boundary lattice point and boundary wall;
5. MRT-LBM iterative computation containing collision, communication with neighbor processes, propagation, and boundary processing until satisfying the convergence conditions;
6. Output the result.

#### 3.1. Domain Decomposition

With regard to three dimension, it is a best choice to decompose almost evenly domain along three directions (3D) [7]. Considering the 3D, each MPI process copes with a sub-domain, and the sub-
domain is marked with a triple \((i, j, k)\) based on MPI process rank (noted as \(\text{prank}\)), the triple can be got by

\[
\begin{align*}
i &= \text{mod}(\text{mod}(\text{prank}, nPX, nPY), nPX), \\
j &= \frac{\text{mod}(\text{prank}, nPX \times nPY)}{nPX}, \\
k &= \frac{\text{prank}}{nPX \times nPY}.
\end{align*}
\]

where \(nPX\) and \(nPY\) denote the number of domain decomposition along \(x\), \(y\), and \(z\) direction respectively. Then, sub-domain range of \(x\) direction can be calculated as follows

\[
\text{rag}X\text{Beg} = i \times \frac{nLatX}{nPX} + \minInt(i, \text{mod}(nLatX, nPX)),
\]

\[
\text{rag}X\text{End} = \begin{cases} 
\text{rag}X\text{Beg} + \frac{nLatX}{nPX} - 1, & \text{mod}(nLatX, nPX) \leq i, \\
\text{rag}X\text{Beg} + \frac{nLatX}{nPX}, & \text{mod}(nLatX, nPX) > i.
\end{cases}
\]

where \(nLatX\) is the count of lattice nodes along the \(x\) direction, \(\text{rag}X\text{Beg}\) is the begin of lattice node of sub-domain along \(x\) direction, and \(\text{rag}X\text{End}\) is the end. The sub-domain range of \(y\) and \(z\) direction can be got similarly.

3.2. Data Exchange

Due to domain decomposition method, a few buffers are needed in sub-domain to store received data after collision. When transferring data with neighbor MPI processes, there is no need to transfer all lattice data \(f_i^+\) of 18 directions, only the expected in propagation operation are required associated with the directions.

Figure 2 and Figure 3 show the 3D domain decomposition along three directions. The sub-domain \((i, j, k)\) exchanges 12 surfaces data with sub-domain \((i - 1, j, k)\), \((i, j - 1, k)\), \((i, j + 1, k)\), \((i, j, k - 1)\), \((i + 1, j, k)\), and \((i + 1, j, k)\) (see Figure 2) and transfers 24 edges data with \((i - 1, j, k + 1)\), \((i - 1, j + 1, k)\), \((i, j - 1, k - 1)\), \((i, j + 1, k - 1)\), \((i, j + 1, k + 1)\), \((i + 1, j - 1, k)\), \((i + 1, j + 1, k)\), \((i + 1, j + 1, k - 1)\), and \((i + 1, j + 1, k + 1)\) (see Figure 3).

![Figure 2 3D surfaces data exchange](image-url)
4. Numerical Experiment

In this section, a 3D incompressible flow around a sphere with a constant velocity profile, \( u = U_\infty = 0.115Ma \), is simulated as a numerical test for our 3d domain decomposition methods. The Reynolds number \( Re = 200 \), and the sphere radius \( D = 40 \) are taken. Figure 4 shows the flow geometry, coordinate system, and computational domain. In order to eliminate the effect of boundaries, the length of the computational domain is \( 51.2D \), the width is \( 12.8D \) and the height is \( 12.8D \). The lattice scale is \( 2048 \times 512 \times 512 \).

Figure 4 Flow geometry, coordinate system, and computational domain

The numerical results are obtained on the cluster of Shanghai University -"Ziqiang 4000". The cluster includes 140 compute nodes. Each node has two CPUs (Intel E5-4650, 2.7GHz/8-core), and each CPU has 8 cores. The shared memory is 64GB in each node. Compute node is connected by 56Gb FDR Infiniband network. The code is compiled on GCC, mpich2 and the "-O3" option which support auto-vectorization for SIMD parallelism.

Figure 5 show the streamlines of the simulation result when \( Re = 200 \) which are feasible and correct. It can be seen that the numerical results agree well with Ref. [16].

Figure 5 Streamline of a slice of simulation result (Re=200)

Figure 6 and Figure 7 show the speedup and efficiency. It can be seen that our parallel MRT-LBM algorithm based on 3D domain decomposition method has a considerable scalability. The efficiency can achieve 92.01% on 2048 cores.
5. Conclusion

The present work has studied parallel algorithms of MRT-LBM containing Cartesian grid generation, domain decomposition method, and data exchange strategy. The only communication occurs after collision during executing entire algorithm. The 3D domain decomposition method is employed to decompose the fluid domain. When there are million cores on a large cluster, it is impractical to decompose along a direction and it is necessary and efficient to decompose along three directions. According to the numerical results in the cluster "Ziqiang 4000" of Shanghai University, the presented parallel algorithm is efficient and scalable. It can be predicted the speedup will not decline even on million CPU cores.

Acknowledgments

This research is supported by NSF of China [No. 91630206] and [No. 11601315], and Shanghai Sailing Program [No. 16YF1404000].

References

[1] S. Chen, H. Chen, D. Martinez, W. Matthaeus, Lattice Boltzmann model for simulation of magnetohydrodynamics, Phys. Rev. Lett., 67, 3776-3779 (1991)
[2] Y. H. Qian, D. D’Humieres, P. Lallemand, Lattice BGK models for Navier-Stokes equation, Europhysics Letters, 17, 479-484 (1992)
[3] L.-S. Luo, W. Liao, X. Chen, Y. Peng, W. Zhang, Numerics of the lattice Boltzmann method: Effects of collision models on the lattice Boltzmann simulations, Phys. Rev. E, 83, 056710 (2011)
[4] D. D’Humieres, Generalized lattice Boltzmann equation, In Rarefied Gas Dynamics: Theory and Simulations, Progress in Astronautics and Aeronautics, 159, 450-458 (1992)
[5] P. Lallemand, L.-S. Luo, Theory of the lattice Boltzmann method: Dispersion, dissipation, isotropy, Galilean invariance, and stability, Phys. Rev. E, 61, 6546-6562 (2000)
[6] J.-S. Wu, Y.-L. Shao, Simulation of lid-driven cavity flows by parallel lattice Boltzmann method using multi-relaxation-time scheme, International Journal for Numerical Methods in Fluids, 46, 921-937 (2004)
[7] C. Schepke, N. Maillard, P. O. A. Navaux, Parallel Lattice Boltzmann Method with Blocked Partitioning, International Journal of Parallel Programming, 37, 593-611 (2009)

[8] J. Gao, H. Xing, V. Rudolph, Q. Li, S. D. Golding, Parallel Lattice Boltzmann Computing and Applications in Core Sample Feature Evaluation, Transport in Porous Media, 107, 65-77 (2014)

[9] Palabos, http://www.palabos.org/index.php

[10] M. Tian, W. Gu, J. Pan, and M. Guo, Performance Analysis and Optimization of Palabos on Petascale Sunway BlueLight MPP Supercomputer, Computers and Fluids, ParCFD 2013: Parallel Computational Fluid Dynamics, 311–320 (2014)

[11] X. Guo, C. Zhong, C. Zhuo, J. Cao, Multiple-relaxation-time lattice Boltzmann method for study of two-lid-driven cavity flow solution multiplicity, Theoretical and Computational Fluid Dynamics, 28, 215-231 (2011)

[12] V.-L. Pedro, J. Johan, Heterogeneous CPU+GPU approaches for mesh refinement over Lattice-Boltzmann simulations, Concurrency Computat.: Pract. Exper, 29, e3919 (2017)

[13] D. Yu, R. Mei, W. Shyy, A Unified Boundary Treatment in Lattice Boltzmann Method, 41st Aerospace Sciences Meeting and Exhibit, (2003)

[14] Z. Guo, C. Zheng, B. Shi, Non-equilibrium extrapolation method for velocity and pressure boundary conditions in the lattice Boltzmann method, Chinese Physics, 11, 365-374 (2002)

[15] C. Pan, J. F. Prins, C. T. Miller, A high-performance lattice Boltzmann implementation to model flow in porous media, Computer Physics Communications, 158, 89-105 (2004)

[16] R. Turton, O. Levenspiel, A short note on the drag correlation for spheres, Powder Technol., 47, 83–86 (1986)