Parallel realization of the computational algorithm based on the implicit lattice Boltzmann equations

Elizaveta A Prokhorova, Gerasim V Krivovichev
Faculty of Applied Mathematics and Control Processes Saint Petersburg State University, 7/9 Universitetskaya nab., Saint Petersburg, 199034, Russian Federation
E-mail: proxliza@mail.ru, g.krivovichev@spbu.ru

Abstract. The approach to parallel realization of the computational algorithm based on the implicit lattice Boltzmann equations is proposed and discussed. Program realization is performed in C++ language using the OpenMP technology. The program is tested on the lid-driven cavity flow problem. The obtained speedup for the cases of the computations with various processes numbers demonstrate the perspectives for realization of the algorithm on systems based on graphics processing units and systems with hybrid architecture.

1. Introduction
Lattice Boltzmann method (LBM) in the last decades is considered as an efficient tool for computational modeling of fluid and gas flows defined by a wide range of the Knudsen number [1, 2]. Despite the traditional approach applied in computational fluid dynamics, LBM is based on the solution of the problems for kinetic equations. Widely used difference schemes are obtained by discretization along the projections of its characteristics on the physical space. These schemes are called lattice Boltzmann equations (LBE’s). The great advantage of the algorithms, based on LBE’s, is based on its high parallelism and possibilities to realization on the modern multiprocessor systems on various architectures.

In the last years, algorithms of LBM are successfully parallelized and its program realizations are implemented in various programming languages and multiprocessor systems. In [3] LBM code is written in C language and is parallelized using OpenMP libraries. Domain decomposition method of data parallelism is adopted. In [4] five different parallel libraries, such as matlabpool, pMatlab, GPU-Matlab, OpenMP and OpenMP+OpenMPI are used to parallel realization of the LBM code. The results of the computations demonstrate that parallel simulations using Matlab subroutines perform almost equally. Parallel simulations using C language with OpenMP libraries were ten times faster. Most of the performed realizations deal with the computations using Graphics Processing Units (GPU’s). In [5] GPU compatible version of the so-called bundle data layout is proposed to achieve the performance of the LBM algorithm. Szoke et al in [6] provides a detailed comparison between the Unified Parallel C (UPC) library and the Computer Unified Device Architecture (CUDA) development kit through the parallelization of the algorithm of two-dimensional LBM. In [7] algorithm of the entropic LBM is realized on the hybrid system, which consists of multicore Central Processing Units (CPU’s) with GPU’s. The resulting computational performance is compared with the results obtained after computation
on a single CPU core and a single GPU device. Wang et al in [8] proposed an approach to the acceleration of the volumetric LBM for the problems of computational haemodynamics.

In the papers, considered above, the algorithms are based on the explicit LBE’s. This type of LBE’s is mathematically formulated as an explicit difference equations, which are conditionally stable. It restricts the values of the relaxation parameter of the gas, so the range of the considered physical processes is restricted. This problem may be solved by an application of the implicit LBE’s.

Presented paper is devoted to the parallel realization of the computational algorithm based on implicit LBE’s proposed by Krivovichev in [9]. These equations may be considered as an extension of the LBE’s proposed in [10].

The structure of the paper is follows. In Section 2 implicit LBE’s are presented and its stability condition is discussed. In Section 3 the parallel algorithm is presented. Program realization and performance of the algorithm are discussed in Section 4. Some concluding remarks are made in Section 5.

2. Implicit lattice Boltzmann equations

Implicit LBE’s are obtained as discrete analogues of the following system of kinetic Bhatnagar — Gross — Krook equations for the modeling of the dynamics of the ensemble of particles with velocities \( V_i \):

\[
\frac{\partial f_i}{\partial t} + V_i \nabla f_i = -\frac{1}{\lambda} (f_i - f_i^{eq}),
\]

where \( f_i = f_i(t, r) \) are the distribution functions of particles with velocities \( V_i = V eq \), where \( V = l/\delta t \), \( l \) is a mean free path, \( \delta t \) is a mean free time, \( e_i \) are the dimensionless vectors, \( t \) is a time, \( r \) is a vector of space variables, \( \lambda \) is a relaxation time, \( f_i^{eq} \) are the equilibrium distribution functions. In this paper two-dimensional case is considered and the following set of vectors \( e_i \) (named D2Q9 lattice) is used: \( e_1 = (0,0), e_2 = (1,0), e_3 = (0,1), e_4 = (-1,0), e_5 = (0,-1), e_6 = (1,1), e_7 = (-1,1), e_8 = (-1,-1), e_9 = (1,-1) \).

System (1) may be rewritten in integral form on the time interval \([t, t + \delta t]\) [11]:

\[
f_i(t + \delta t, r + V_i \delta t) = f_i(t, r) - \frac{\delta t}{\lambda} \int_0^{\delta t} (f_i(t + \xi, r + V_i \xi) - f_i^{eq}(t + \xi, r + V_i \xi)) d\xi.
\]

In [9] system (2) is rewritten in equivalent form with parametric coefficients:

\[
f_i(t + \delta t, r + V_i \delta t) = f_i(t, r) - \frac{\delta t}{\lambda} \int_0^{\delta t} (f_i(t + \xi, r + V_i \xi) - f_i^{eq}(t + \xi, r + V_i \xi)) d\xi - \frac{\sigma}{\lambda} \int_0^{\delta t} (f_i(t + \xi, r + V_i \xi) - f_i^{eq}(t + \xi, r + V_i \xi)) d\xi,
\]

where \( \sigma \in [0,1] \) is a dimensionless parameter.

After the computation of integral in (3) by simple quadrature rules and discretization of (3) on time and space grids, the following system of implicit difference equations is obtained

\[
f_i(t_j + \delta t, r_{kl} + V_i \delta t) - f_i(t_j, r_{kl}) = A(\sigma, \tau)(f_i(t_j, r_{kl}) - f_i^{eq}(f(t_j, r_{kl}))) + B(\sigma, \tau)(f_i(t_j + \delta t, r_{kl} + V_i \delta t) - f_i^{eq}(f(t_j + \delta t, r_{kl} + V_i \delta t))),
\]

(4)
where coefficients $A$ and $B$ are defined by quadrature rule used for the computation of integrals in (3), $t_j$ is a node of time grid constructed with step $\delta t$, $r_{kl}$ is a node of space grid constructed with step $l$. In [9] six systems of equations (4) are obtained.

The expression for the apparent viscosity of LBE’s (4) may be obtained by application of the Chapman — Enskog method to the first differential approximation of (3) [12] and is written as follows:

$$\nu = \left(1 + \frac{A - B}{2}\right) \frac{\tau l^2}{3 \delta t}. \quad (5)$$

The stability conditions of LBE’s (4) may be obtained from the condition of non-negativity of $\nu$: $\nu \geq 0$. For example, for the LBE’s defined by the coefficients $A$ and $B$, this condition is written as following inequality: $\sigma \geq 1/2 - \tau$. In the presented paper, we consider this system of implicit LBE’s for the performing of all calculations.

3. Parallelization of the algorithm

The main computational problem solved during the calculations based on LBE’s (4) is a problem of the solution of the systems of nonlinear algebraic equations for the computation of $f_i(t_j + \delta t, r_{kl} + V_i \delta t)$. These systems are formed by (4) and are written in every node of two-dimensional grids. For the convenience, let us rewrite (4) in following form:

$$f_i(t_j + \delta t, r_{kl}) - f_i(t_j, r_{kl} - V_i \delta t) = A(\sigma, \tau)(f_i(t_j, r_{kl} + V_i \delta t) - f_i^{eq}(f(t_j, r_{kl} + V_i \delta t))) +$$

$$+ B(\sigma, \tau)(f_i(t_j + \delta t, r_{kl}) - f_i^{eq}(f(t_j + \delta t, r_{kl}))), \quad (6)$$

So at one time step $[t_j, t_j + \delta t]$ system (6) must be solved in every internal node $r_{kl}$ of the space grid for the determination of the values $f_i(t_j + \delta t, r_{kl})$. If the case of D2Q9 lattice is considered ($i = 1, 9$) and space grid of $N_x \times N_y$ nodes is used, at every time step $(N_x - 2) \times (N_y - 2)$ systems of nine nonlinear algebraic equations must be solved. It must be noted, that at moment $t = t_j + \delta t$ these systems are independent on each other, so the algorithm is highly parallelizable on data.

The following approach to the parallelization of the algorithm may be proposed: at every time step the marches on nodes of fixed space coordinate ($x$ or $y$) may be considered independently and every march may be realized by a specific computational process. In the presented paper march on coordinate $x$ is realized (fig. 1). During the computations these processes may be performed independently in cycle on $k$ from 2 to $N_x - 1$ and results must be synchronized during the march on $x$ at every time step by means of the technology of the programming realization.

In every grid node system (6) is solved by the Newton — Raphson method. Let us rewrite (6) in node $r_{kl}$ in compact form. Let us introduce new variables $z_i = f_i(t_j + \delta t, r_{kl})$, $i = 1, n$, so (6) may be rewritten as:

$$F(z) = 0,$$

where $F_i = (1 - B)z_i + B f_i^{eq}(z) - \varphi_i$, $\varphi_i = (1 + A)f_i(t_j, r_{kl} - V_i \delta t) + A f_i^{eq}(f(t_j, r_{kl} - V_i \delta t))$.

The Newton — Raphson method is based on the following iterative procedure:

$$z^{s+1} = z^s - [\nabla F(z^s)]^{-1} F(z^s), \quad (7)$$

where $\nabla F$ is a Jacobi matrix defined by following components:

$$\frac{\partial F_i}{\partial z_j}(z^s) = \begin{cases} 1 - B + B \frac{\partial f_i^{eq}}{\partial z_j}(z^s), & i = j, \\ B \frac{\partial f_i^{eq}}{\partial z_j}(z^s), & i \neq j. \end{cases}$$
For the avoiding of the procedure of the inversion of the Jacobi matrix, the following equivalent of the procedure (7) is used:

\[ z^{s+1} = z^s + \delta z^s, \]

where \( \delta z^s = z^{s+1} - z^s \) may be obtained after the solution of the following system of linear algebraic equations:

\[ [\nabla F(z^s)] \delta z^s = -F(z^s). \]  

In our work system (8) is solved by Gaussian method.

Initial member of the iterative sequence \( z^0 \) is obtained by application of the explicit LBE’s [11]:

\[ f_i(t_j + \delta t, r_{kl}) - f_i(t_j, r_{kl} - V_i \delta t) = -\frac{1}{\tau} (f_i(t_j, r_{kl} - V_i \delta t) - f_i^{(eq)}(f(t_j, r_{kl} - V_i \delta t))), \]

so the procedure ”predictor-corrector” is realized during the computations.

4. Program realization

Due to the proposed algorithm, the parallelization is realized by processes, which may be realized by threads. So the program realization may be effectively performed by using the procedures of OpenMP technology. Using the OpenMP directives the parallel performance may be realized by introduction of parallel sections in the code. It may be realized at the following way: the section of code that is meant to run in parallel regimes is marked accordingly, with a compiler directive that will cause the threads to form before the section is executed.
The realization is performed using C++ language. The march procedure on the coordinate $x$ is realized by directive `pragma omp for` for the cycle on $k$.

The program realization is tested on the numerical solution of the problem of lid-driven cavity flow, stated in [14]. The performance of the parallel realization of the algorithm is estimated by computations on space grids with $50 \times 50$, $100 \times 100$, $150 \times 150$, $200 \times 200$ and $250 \times 250$ nodes. After the computations on every grid the speedup is used for the estimation of the algorithm:

$$R(n) = \frac{T_1}{T_n},$$

where $n$ is a number of threads, $T_1$ is a time of computation for the case of one thread.

All computations are realized on a computational cluster of the Faculty of Applied Mathematics and Processes of Control of Saint-Petersburg State University. At fig. 2 the plots of speedup $R(n)$ for the case of every space grid are presented. As it can be seen, the speedup dependency on $n$ is close to linear, so the proposed algorithm is effective for the parallel realization and perspective for realization on system with multicore GPU’s. For this purpose authors wish to realize the proposed parallel algorithm for computations using CUDA technology. After this step and after testing of the obtained software, the authors try to make their code free to download according to GNU license agreement.

5. Conclusion
In the presented paper the approach to parallel realization of the computational algorithm based on implicit LBE’s is proposed and discussed. Program realization is performed in C++ language using the OpenMP technology. The program is tested on the lid-driven cavity flow problem. The obtained speedup for the cases on the computations with various processes numbers demonstrate the perspectives for realizations on GPUs and systems with hybrid architectures.
Acknowledgements
The reported study was funded by Russian Foundation of Basic Research according to research project No. 16-31-00021 mol_a.

References
[1] Levesley J, Gorban A N and Packwood 2011 *Approximation algorithms for complex systems* ed E H Georgoulis, A Iske and J Levesley (Berlin: Springer-Verlag) p 127
[2] Bo Y, Wang P, Guo Z and Wang L-P 2017 *Comp. and Fluids* **155** 9
[3] Basha M, Sidik M A C and Beriache M 2017 *Int. J. of Heat and Mass Transf.* **115** 158
[4] Basha M and Sidik M A C 2018 *Int. J. of Heat and Mass Transf.* **116** 715
[5] Robertsen F, Westerholm J and Mattila K 2017 *Int. J. of High Perf. Comput. Appl.* **31** 246
[6] Szoke M, Jozsa T I, Koleszar A, Moulitsas I and Konoszy L 2017 *ACM Trans. on Math. Soft.* **44** 8
[7] Ye Y, Li K, Wang Y and Deng T 2015 *Comp. and Fluids* **110** 114
[8] Wang Z, Zhao Y, Sawchuk A P, Dalsing M C and Yu H 2015 *Comp. and Fluids* **115** 192
[9] Krivovichev G V 2014 *Appl. Math. Sci.* **8** 5003
[10] Sankaranarayanan B, Shan X, Kevrekidis I G and Sundaresan S 2002 *J. Fluid Mech.* **452** 61
[11] He X and Luo L-S 1997 *Phys. Rev. E* **55**(6) R6333
[12] Sofonea V 2003 *J. Comp. Phys.* **184**, 422
[13] Wolf-Gladrow D A 2000 *Lattice-Gas Cellular Automata and Lattice Boltzmann Models* (Berlin: Springer)
[14] Ghia U, Ghia K N and Shin C T 1982 *J. Comp. Phys.* **48** 387