Walailak J Sci & Tech 2004; 1(1):53-70.

Two Dimensional Lattice Boltzmann Method for Cavity Flow Simulation

Panjit MUSIK[1,2] and Krisanadej JAROENSUTASINEE[2]

Faculty of Science and Technology[1], Rajabhat Institute Nakhon Si Thammarat and School of Science[2], Walailak University, Thasala, Nakhon Si Thammarat 80160, Thailand.

ABSTRACT

This paper presents a simulation of incompressible viscous flow within a two-dimensional square cavity. The objective is to develop a method originated from Lattice Gas (cellular) Automata (LGA), which utilises discrete lattice as well as discrete time and can be parallelised easily. Lattice Boltzmann Method (LBM), known as discrete Lattice kinetics which provide an alternative for solving the Navier–Stokes equations and are generally used for fluid simulation, is chosen for the study. A specific two-dimensional nine-velocity square Lattice model (D2Q9 Model) is used in the simulation with the velocity at the top of the cavity kept fixed. LBM is an efficient method for reproducing the dynamics of cavity flow and the results which are comparable to those of previous work.

Key words: Cavity flow - Lattice Boltzmann Method - Lattice Gas (cellular) Automata - Incompressible Viscous flow - D2Q9 Model

INTRODUCTION

The objective of our study is to simulate water flow using Lattice Boltzmann Method (LBM) (1,2,3,4). In general, in order to understand the dynamics of water flow, we have to solve the Navier-Stokes equation (5) since it is the most fundamental equation that governs fluid flow. Initial variables to study are, for example, velocity ($\vec{u}$), density (r) and pressure (P). The Navier-Stokes equation is a partial differential equation that is difficult to solve for complicated boundary and initial conditions. Consequently, numerical methods have been applied to find their solutions. Those conventional numerical methods are Computational Fluid Dynamics (CFD), Finite Difference Method (FDM), Finite Element Method (FEM), Finite Volume Method (FVM) etc. A new numerical method with more advantages has recently made its appearance among those conventional ones. This is the LBM that was developed from Lattice Gas Automaton of fluid. LBM enables parallelisation; moreover, it has the simplicity of programming and the ability to incorporate microscopic interaction.

Historically, Frisch and co-workers introduced Lattice Boltzmann Equation (LBE) to calculate the viscosity of the Lattice Gas Cellular Automata (LGCA) in 1987 (6,7,8). In 1988, McNamara and Zanetti introduced Lattice Boltzmann models as an independent numerical method for hydrodynamic simulation. LGCA was modified to get rid of the noise. The Boolean fields were replaced by continuous distribution over Frisch Hasslacher Pomeau (FHP) and Face-Centered–Hyper-Cubic (FCHC) Lattice with Fermi-Dirac distribution as equilibrium function. In 1989, Higuera and Jiménez simplified the LBM by linearising collision operator. From 1990 to 1991 Chen and co-workers, and Qian and co-workers replaced Fermi-Dirac distribution by Maxwell-Boltzman distribution that is close to the local equilibrium state. Around 1991 and 1992, Koelman, Qian et al., and others recovered the Navier-Stokes macroscopic equation by choosing the local equilibrium distribution in the Lattice BGK model, and collision operator was replaced by single time relaxation.

This work utilises a version of LBM in *Mathematica* computing environment (9,10,11). *Mathematica* is known to be a high-level programming language as well as a computing software system that provides a wide variety of numerical, algebraic, symbolic and graphical computations. From our experience of undertaking this work, *Mathematica* is found to have expected simplicity.

Lattice Boltzmann Method for Two-dimensions: Nine-velocity Square Lattice Model

A square Lattice with unit spacing is used on each node of which there are eight neighbours connected by eight links. Particles can only reside on a node and move to their nearest neighbours along these links in the unit time. There are 3 types of particles on each node with nine different velocities. $\dot{e}_{\sigma i}$ is the velocity (see **Figure 1**). The symbol s signifies the type of particle (s = 0, 1, 2). The symbol *i* represents the velocity direction. That is, for a particle at rest, we use $\vec{e}_{0i}$ with speed $\left|\vec{e}_{0i}\right| = 0$. For a particle moving along axes, we use $\vec{e}_{1i}$ with speed $\left|\vec{e}_{1i}\right| = 1$ and for a diagonally moving particle, $\vec{e}_{2i}$ with speed $\left|\vec{e}_{2i}\right| = \sqrt{2}$.

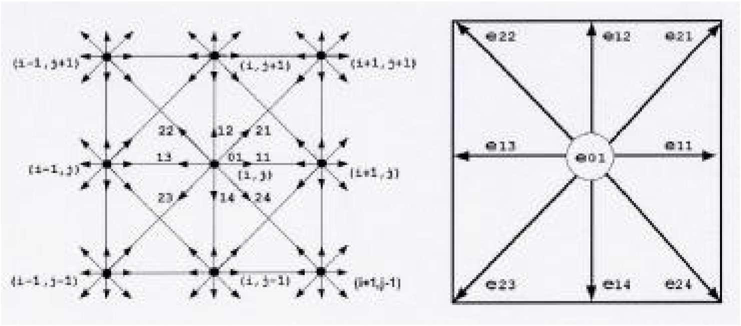


Figure 1. Nine-velocity square Lattice model (D2Q9 Model)

The occupation of the three types of particles is represented by the single-particle distribution function, $f_{\sigma i}(\dot{x}, t)$ which is the probability of finding a particle at node $\vec{x}$ and time t with velocity $\dot{e}_{\sigma i}$. When $\sigma = 0$, there is only f_{01}. The occupation of particles moving along axes is defined as f_{11}, f_{12}, f_{13} and f_{14} while the occupation of diagonally moving particles is defined as f_{21}, f_{22}, f_{23} and f_{24}. In general, the particle distribution function satisfies the Lattice Boltzmann equation.

$$f_{\sigma i}(\vec{x}+\vec{e}_{\sigma i}, t+1) - f_{\sigma i}(\vec{x}, t) = \vec{\Omega}_{\sigma i}, \tag{1}$$

where $\Omega_{\sigma i}$ is the collision operator, representing the rate of the particle distribution change due to collisions.

The collision operator is then simplified by the single time relaxation approximation (12) and the Lattice Boltzmann equation becomes the Lattice Boltzmann GK (LBGK) equation (in Lattice units):

$$\begin{aligned} f_{\sigma i}(\vec{x}+\vec{e}_{\sigma i}, t+1) - f_{\sigma i}(\vec{x}, t) \\ = -\frac{1}{\tau}\left[f_{\sigma i}(\vec{x}, t) - f_{\sigma i}^{(0)}(\vec{x}, t)\right], \end{aligned} \tag{2}$$

where $f_{\sigma i}^{(0)}(\vec{x}, t)$ is the equilibrium distribution at $\vec{x}$ and t and τ is the single time relaxation, which controls the rate of approaching equilibrium. The density per node, ρ, and the macroscopic velocity, $\vec{u}$ are defined in terms of the particle distribution function by

$$\rho = \sum_{\sigma}\sum_{i} f_{\sigma i}, \quad \rho\vec{u} = \sum_{\sigma}\sum_{i} f_{\sigma i}\vec{e}_{\sigma i} \tag{3}$$

A suitable equilibrium distribution can be chosen in the following form for each type of particles.

$$\begin{aligned} f_{0i}^{(0)} &= \frac{4}{9}\rho\left(1 - \frac{3}{2}|\vec{u}|^2\right) \\ f_{1i}^{(0)} &= \frac{1}{9}\rho\left(1 + 3(\vec{e}_{1i}\cdot\vec{u}) + \frac{9}{2}(\vec{e}_{1i}\cdot\vec{u})^2 - \frac{3}{2}|\vec{u}|^2\right) \\ f_{2i}^{(0)} &= \frac{1}{36}\rho\left(1 + 3(\vec{e}_{2i}\cdot\vec{u}) + \frac{9}{2}(\vec{e}_{2i}\cdot\vec{u})^2 - \frac{3}{2}|\vec{u}|^2\right) \end{aligned} \tag{4}$$

The relaxation time is related to the viscosity by

$$\tau = \frac{6\nu + 1}{2}, \tag{5}$$

where n is the kinetic viscosity.

Having chosen the appropriate Lattice size and the characteristic velocity, v can be calculated for a given Reynolds number (Re) and then the relaxation time can be determined by using Eq. (5). Starting from an initial density and velocity fields, the equilibrium distribution function can be obtained using Eq. (4), and $f_{\sigma i}(\dot{x}, t)$ can be initialised as $f_{\sigma i}^{(0)}(\vec{x}, t)$. For each time step, the updating of the particle distribution can be split into two sub-steps: collision and streaming. It is irrelevant which one is the first for a long time run. The collision process at position $\vec{x}$ occurs according to the right hand side of the Lattice Boltzmann equation given as Eq. (2). The resulting particle distribution at $\vec{x}$, which is the sum of the original distribution and the collision term, is then streamed to the nearest neighbour of $\vec{x}+\vec{e}_{\sigma i}$, for particle velocity $\vec{e}_{\sigma i}$. Then r , $\vec{u}$ can be computed from the updated $f_{\sigma i}(\vec{x},t)$ using Eq. (3). The updating procedure can be terminated for steady state problems when a certain criterion is satisfied. The method can also be used for transient problems.

Cavity Flow Simulation

Cavity flow simulation uses Cartesian coordinates with the origin located at the lower left hand corner as shown in **Figure 2.** The cavity has an *n* node on each side. The velocity components *u* and *v* are in *x* and *y* directions. Re is the Reynolds number defined as

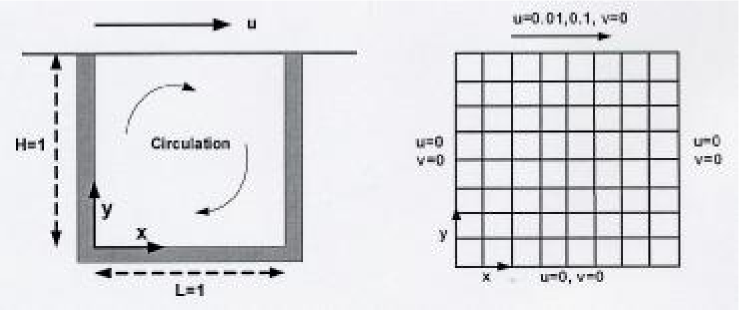


Figure 2. Cavity flow configuration, coordinate, boundary conditions and initial conditions of the velocity

$$\mathrm{Re}=\frac{|\vec{u}|H}{\nu}, \tag{6}$$

and ν is the kinematic viscosity as given in Eq. (5), the height of the cavity is *H*, and velocity of the top plate is $\vec{u}$.

The top boundary moves from left to right with constant velocity. Initially the velocities at all nodes, except at the top node, are set to zero. The *x*-velocity of the top is *u*=0.01 for Re<100, *u*=0.1 for Re≥100 and the *y*-velocity, *v*=0. Uniform fluid density $\rho=2.7$ is imposed initially. The equilibrium distribution function, $f_{\sigma i}^{(0)}$ is then calculated using (4) and $f_{\sigma i}$ is set to equal to $f_{\sigma i}^{(0)}$ for all node at *t*=0. The evolution of $f_{\sigma i}$ can then be found by succession of streaming and relaxation processes. After streaming, the velocity of the top boundary is reset to its uniform initial velocity. At the end of each streaming and collision process cycle, $f_{\sigma i}$ at the top is set to the equilibrium state. Bounce-back boundary conditions are used on the three stationary walls.

There are three types of nodes on a Lattice, namely, fluid nodes (white node), bounce-back (no-slip) boundary conditions nodes or wall nodes (black nodes), and pressure-velocity boundary conditions nodes or top nodes (gray nodes). (see **Figure 3**)

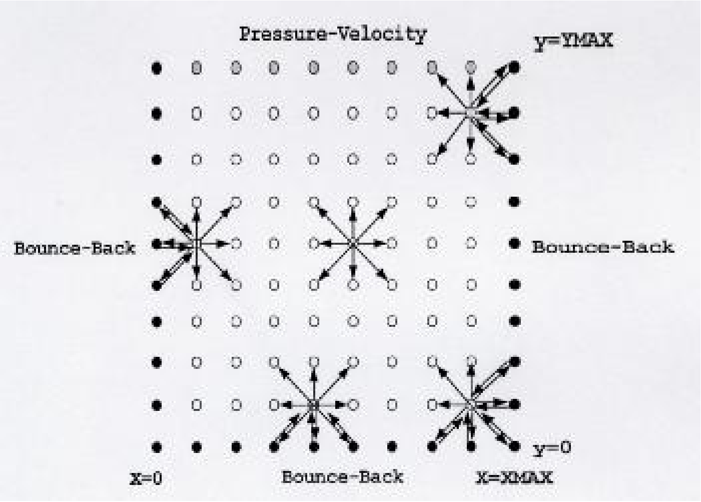


Figure 3. Boundary conditions of cavity flow. Particles propagate to their next neighbours

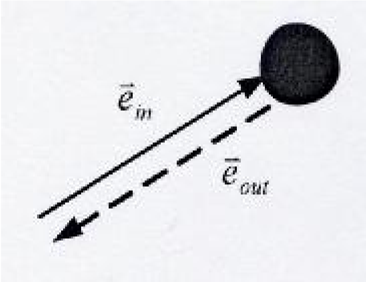


Figure 4. Bounce-back boundary conditions, $\overset{\mathbf{v}}{e}_{\text{out}} = -\overset{\mathbf{v}}{e}_{in}$

We use two types of boundary conditions (8,13) in this simulation.

1. Bounce back boundary conditions

The on-grid situation is easy; just reverse all populations sitting on a boundary node. When fluid nodes move into wall nodes, the bounce-back occurs; fluid nodes move in the opposite directions with the same speed or $f_{out} = f_{in}$, as shown in **Figure 5**.

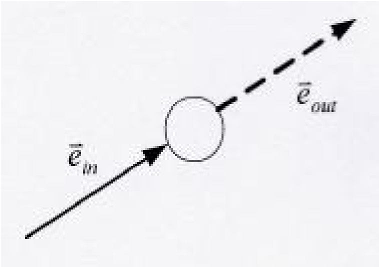


Figure 5. Propagation boundary conditions, $\dot{e}_{out} = -\dot{e}_{in}$

2. Propagation boundary conditions (see **Figure 5**)

When fluid nodes move into fluid nodes, they continue moving in the same direction with the same speed, as shown in **Figure 6**.

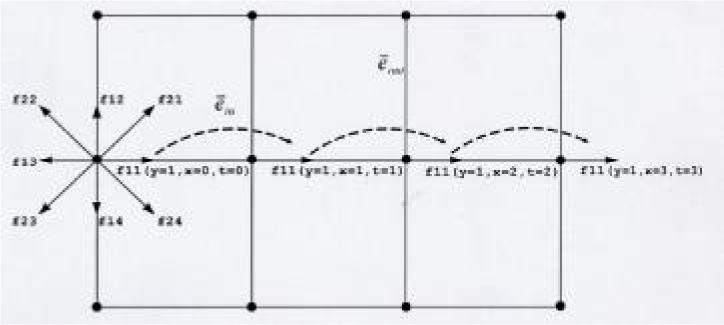


Figure 6. Propagations of f_{11}

At t=0, the first time step, f_{11} is at the node x=0, y=1. At t=1, f_{11} moves to node x=1, y=1, and so on. The propagation of $f_{12}, f_{13}, f_{14}, ..., f_{24}$ occurs in the similar way as f_{11} in their own directions except f_{01} which is the propagation for rest particles. These rest particles are of course still at the node.

The complexity of the problem depends on the speed on the top, viscosity, and size and geometry of the cavity.

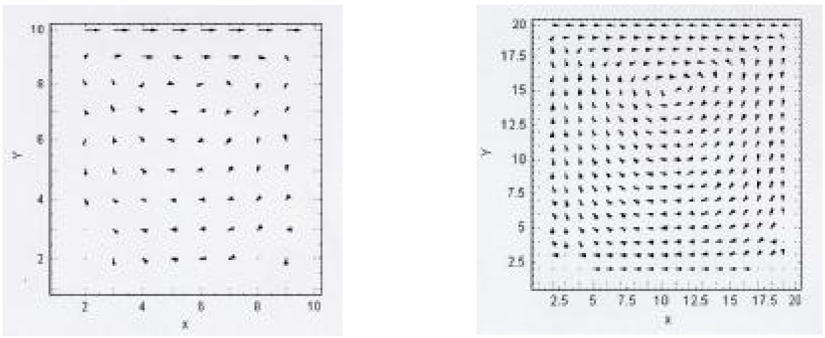
Our Algorithm for Programme Construction.

1. Data input are Lattice size, $t_{\max}$, the initial density, $\rho = 2.7$, Re, u=0.01 or u= 0.1
2. Compute the viscosity, n and the relaxation time (t), (Eq. 5,6)
3. Compute the initial $f_{\sigma i}^{(0)}$ and initial $f_{\sigma i} = f_{\sigma i}^{(0)}$ (Eq. 4)
4. Implementation of LBM on the Lattice (loop computation with model time)
   - 4.1 Compute streaming of particles to new locations (shifting the particle distribution function to the neighbour node in direction of the velocity)
   - 4.2 Compute the effect of boundary conditions (bounce-back rule if walls)
   - 4.3 Compute particle function redistribution due to collision (Eq. 2)
   - 4.4 Compute the macroscopic variables (density, velocity) (Eq. 3)
   - 4.5 Compute the new equilibrium particle distribution function (Eq. 4)
   - 4.6 Check for reaching the steady flow, $t=t_{\max}$ to stop the calculations.
5. Results output (velocity vector plot, streamlines, velocity profile and velocity components).

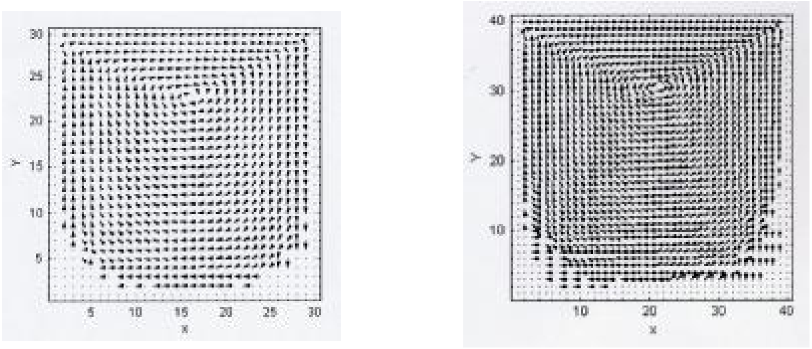
All of the results are normalised to allow comparisons between the present work and other previous studies based on a unit square cavity with unit velocity of the top boundary.

RESULTS

Figure 7 shows that better resolutions can be obtained when the number of Lattice nodes increases. The observed circulation in the cavity is clockwise. The centre of the primary vortex is located at (0.53, 0.75) for the present work, and (0.52, 0.72) in (14). In addition, **Figure 8** shows fluid streamlines when reaching steady state with Re=10 for different Lattice sizes. The better resolution can be observed as the number of Lattice nodes increases.

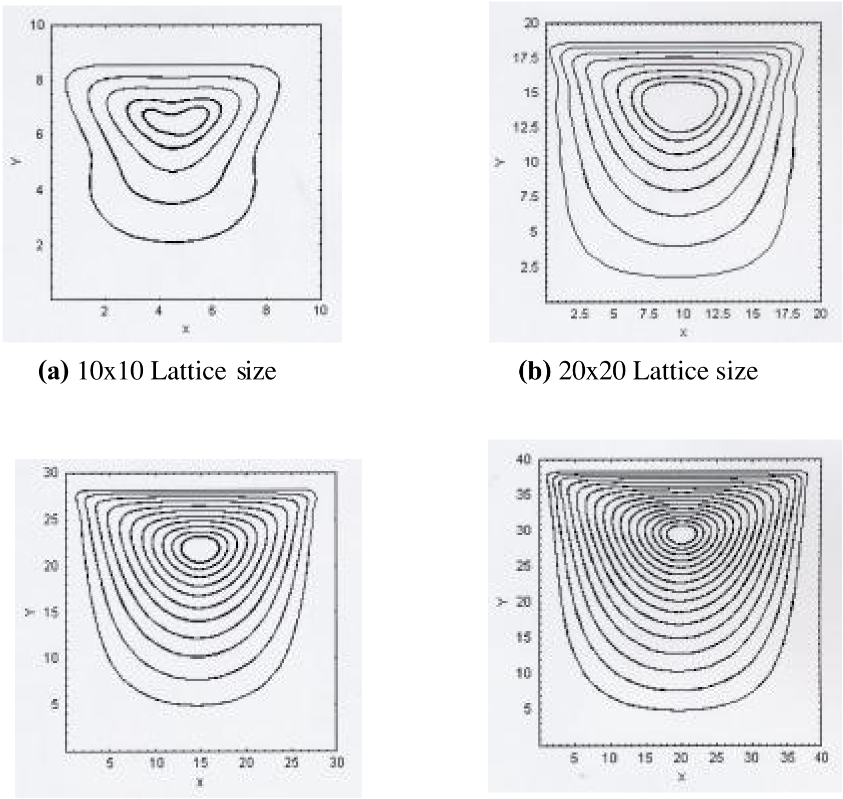


(a) 10x10 Lattice size **(b)** 20x20 Lattice size



(c) 30x30 Lattice size **(d)** 40x40 Lattice size

Figure 7. Velocity vector plots when the system reaches steady state with Re=10 for different Lattice sizes

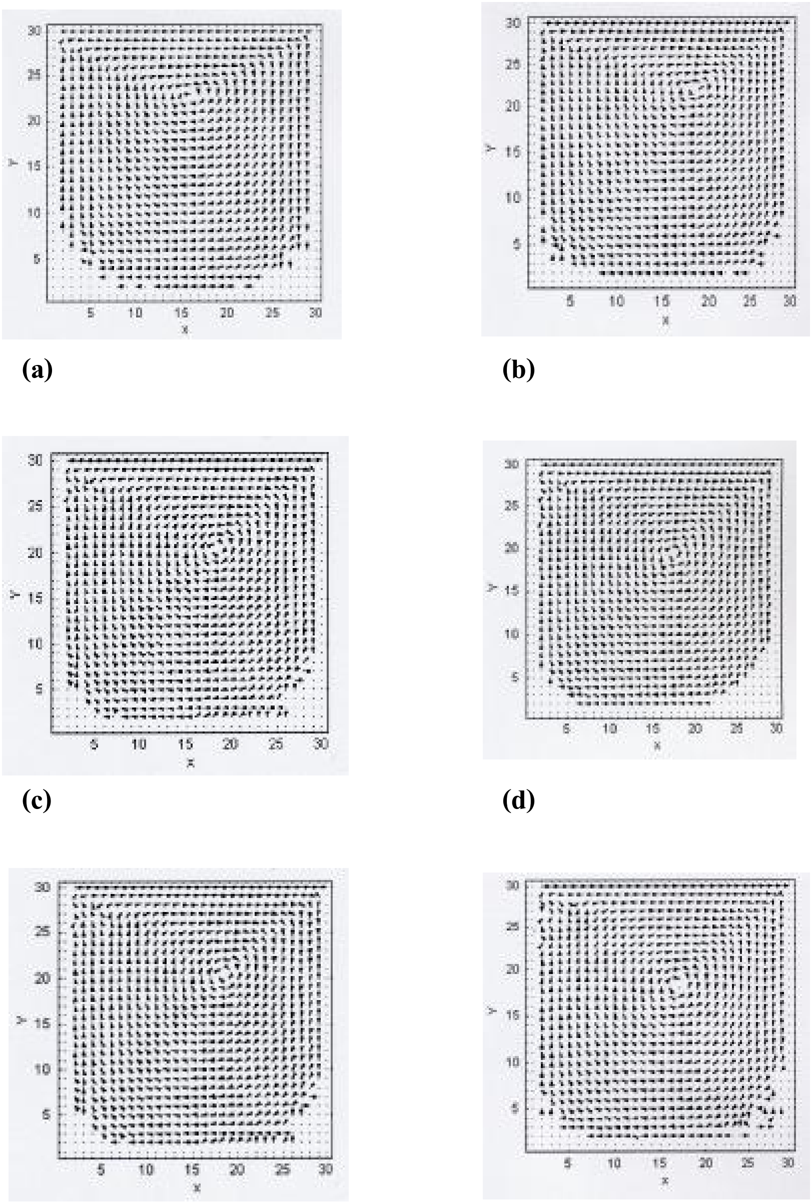


(a) 10x10 Lattice size **(b)** 20x20 Lattice size

(c) 30x30 Lattice size **(d)** 40x40 Lattice size

Figure 8. Fluid streamlines when reaching steady state with Re=10 for different Lattice sizes

Figure 9 shows velocity vector field plots when reaching steady state with 30×30 Lattice size for different Re. The primary vortex moves to the centre when Re grows higher e.g. for Re=100 the vortex is at (0.60,0.73) for the present work, and at (0.62,0.74) in (15). For Re=400, the vortex is at (0.57,0.63) for the present work, and at (0.56,0.60) in (14).



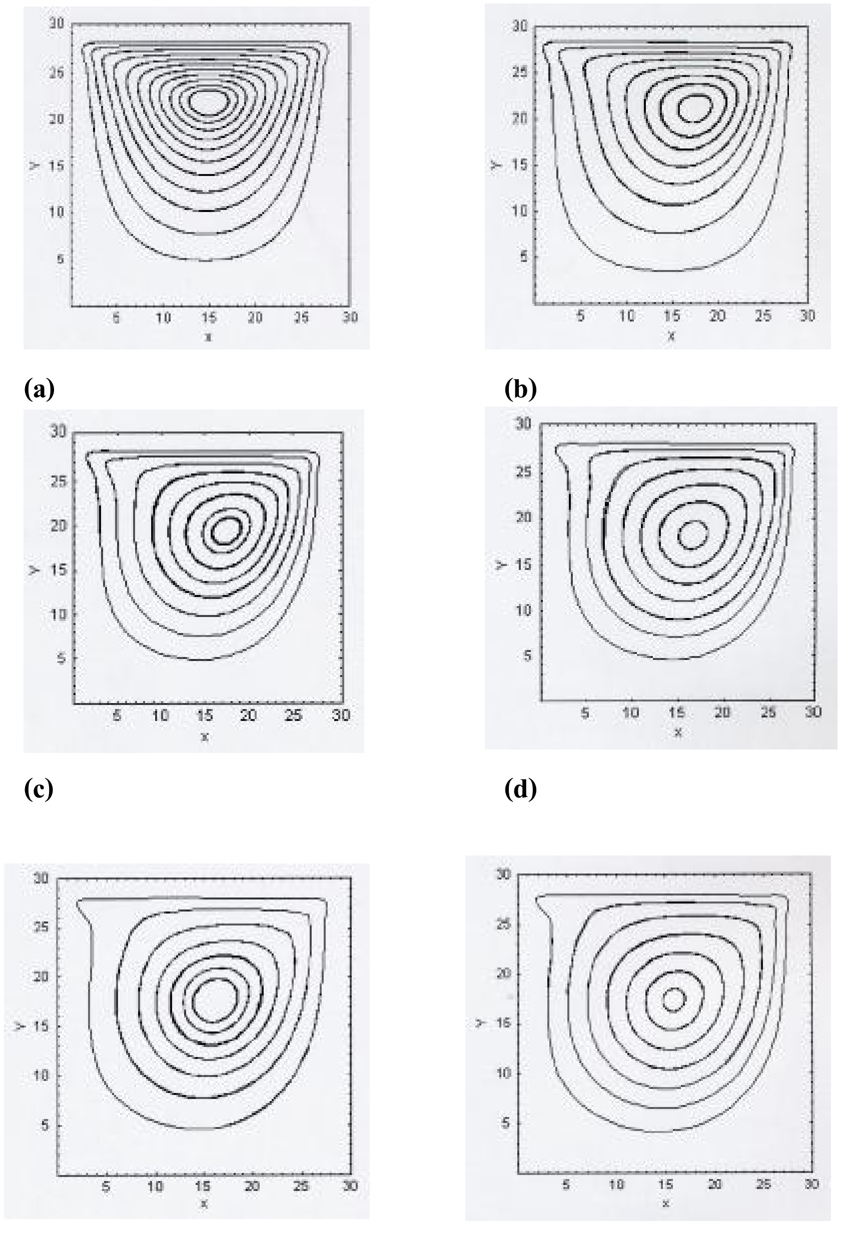
(a) **(b)**

(c) **(d)**

(e) **(f)**

Figure 9. Velocity vector fields. Re numbers were set to 10, 100, 200, 300, 400, and 500 for a, b, c, d, e, and f respectively

Fluid streamlines are shown in **Figure 10** when the system reaches steady state with 30×30 Lattice size for different Re. We observe the primary vortex moving to the centre when Re grows higher. From the plots, circular streamlines for high Re are noted.



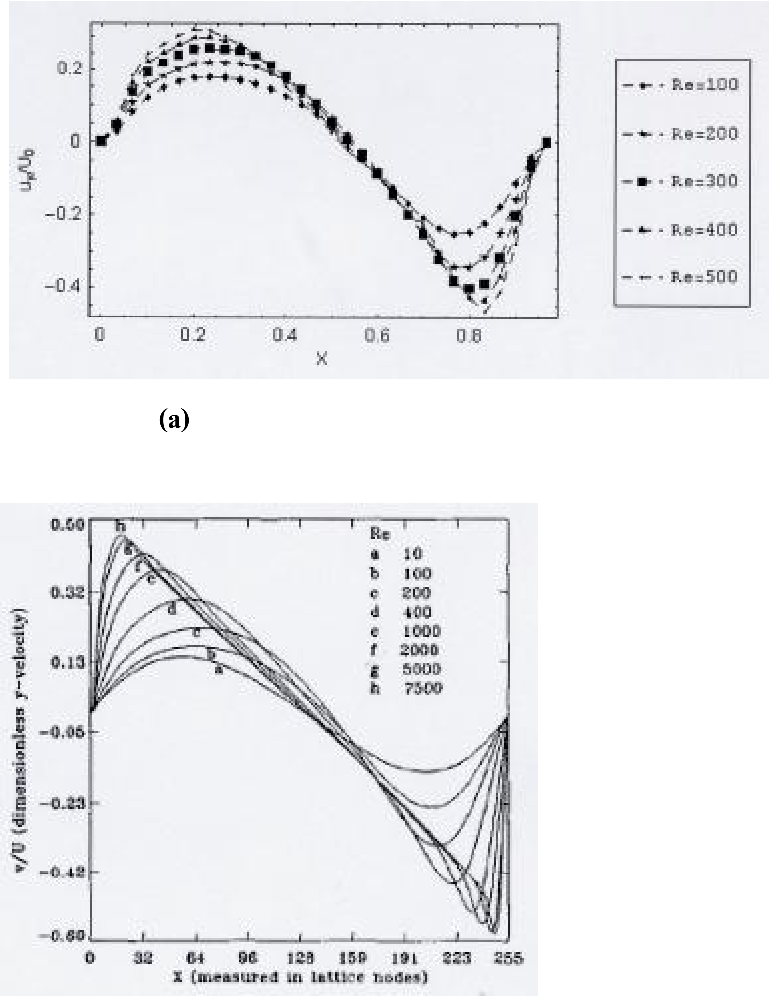
(a) **(b)**

(c) **(d)**

(e) **(f)**

Figure 10. Streamlines of fluid flow. Re numbers were set to 10, 100, 200, 300, 400, and 500 for a, b, c, d, e and f respectively

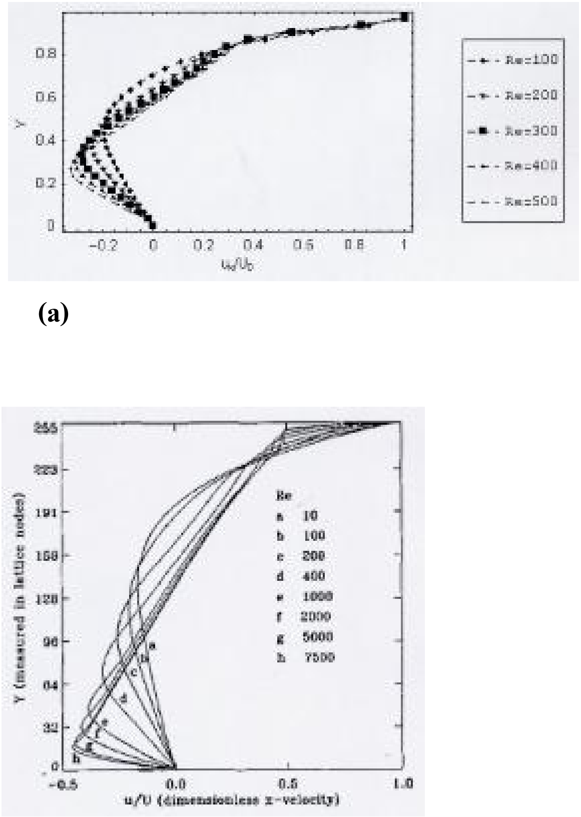
Figure 11 shows comparison of velocity profiles for u_y at y=0.5 when reaching steady state between present results and (14). The velocity profiles observed from the two works are in good agreement. In addition, **Figure 12** shows comparison of velocity profiles for u_x at x=0.5 when reaching steady state between present results and (14). The velocity profiles observed from the two works are also in good agreement.



(a)

(b)

Figure 11. The velocity profiles for v (u_y) change from curve at lower values of Re to straight line for higher Re numbers. (a) the result from present work and (b) the result from (12)

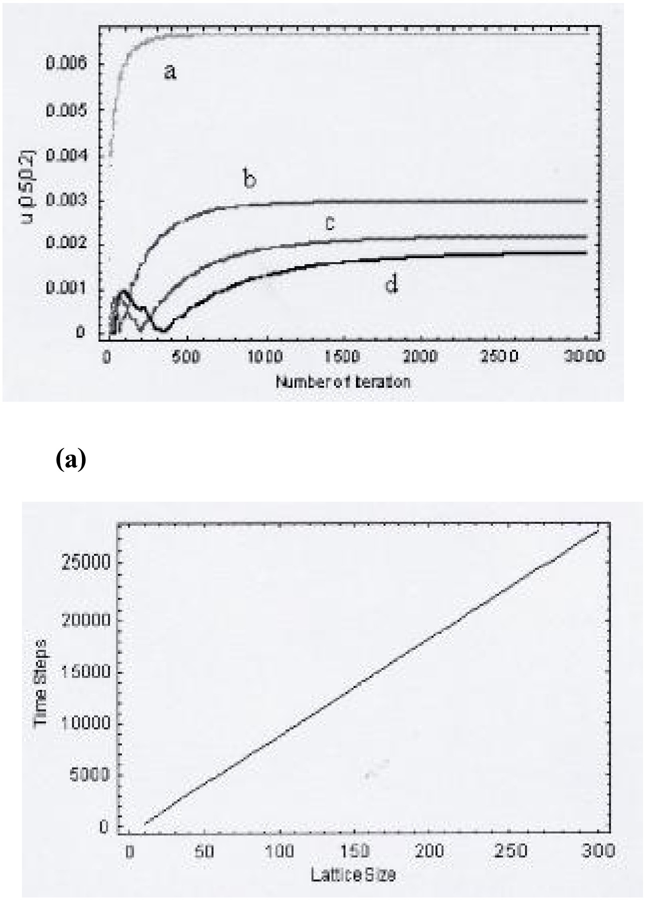


(a)

(b)

Figure 12. The velocity profiles for u (u_x) change from curve at lower values of Re to straight line for higher Re numbers. (a) the result from present work and (b) the result from (12)

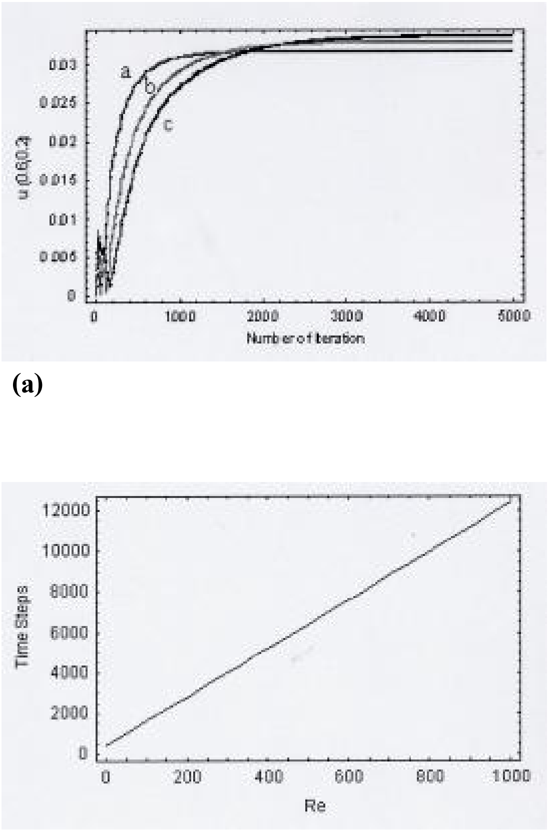
Steady state behaviour in different Lattice sizes for Re =10 is shown in **Figure 13**. The fine Lattice requires more computing time to reach steady state. A linear, not power-law, increase of time step to reach steady state for different Lattice sizes is observed. Steady state behaviour for 20×20 Lattice size with different Re numbers is also shown in **Figure 14**. At low Re numbers the time step to reach steady state is shorter and the time step to reach steady state increases linearly with Re numbers. Finally, **Figure 15** shows the relationship between computing time for a single step and Lattice size. Exponential increases are noted.



(a)

(b)

Figure 13. Steady state behaviour with different Lattice sizes for Re =10. (a) Velocity component, u (0.5,0.2), Labels a, b, c and d are for 10×10, 20×20, 30×30 and 40×40 Lattice sizes respectively. (b) Time step to reach steady state for different Lattice sizes



(a)

(b)

Figure 14. Steady state behaviours for 20×20 Lattice size with different Re numbers. (a) Velocity component, u (0.5,0.2) at each time step. Labels a, b and c are for Re = 100, 200 and 300 respectively. (b) The time step to reach steady state increases linearly when Re increases

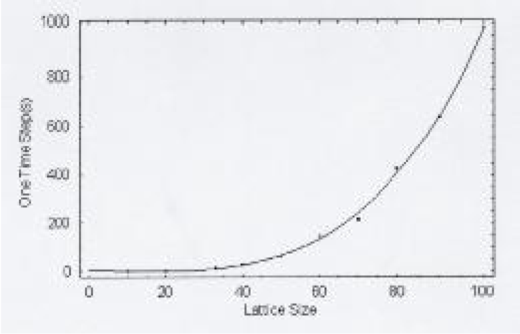


Figure 15. Relationship between computing time for a single step and Lattice sizes. Dots represent computing time for one step from actual simulation and the line is the result of an exponential fitted function

Long computing time on the personal computer (Pentium III 1 GHz, 128 MB RAM, 100 MHz Front size bus, Microsoft Windows XP Professional and *Mathematica* 4.1) alerts us to observe our programme behaviour. Estimated computing time and numbers of time step to reach steady state are shown in **Table 1**.

Table 1. Numbers of time step and computing time for various Lattice sizes with Re=10

Lattice size	Number of Time Step	Estimated Computing Time (days)
65×65	5553	13.6
129×129	11537	205.8
257×257	23505	2111.7
513×513	47441	18920.8

When we perform a simulation with any Re's in a small Lattice, we can also predict the time step and the whole time we need to reach steady state in a larger Lattice. Results are shown in **Table 2.**

Table 2. Numbers of time step and computing time to reach steady state for various Re number for 20×20 Lattice size

Re	Number of Time Step	Predicted Computing Time (hr)
1000	12014	6.3
2000	24014	12.7
5000	60014	31.7
7500	90014	47.5
10000	120014	63.3

Similarly, when we perform a simulation with any Lattice sizes and a small Re we can also predict the time step and the whole time we need to reach steady state for higher Re simulation. The results are shown in **Figure 15**.

DISCUSSION

Using LBM with D2Q9 model, it is observed that Re number affects the primary vortex location and time to reach the steady state while the Lattice size affects only the time to reach the steady state. Re number also has an effect on the time to reach steady state. The simulation of a small scale cavity flow in 2D by LBM in this work strongly calls for a further development of parallel programming in order to improve the computing time and to increase grid sizes because the simulation results indicate a non-linear increase of computing time to reach steady state over Lattice sizes and Re numbers. A PC cluster and perhaps a more efficient parallel programme for computation are required to develop the simulation to finer Lattice with higher Re. It should be noted that even when the velocity profile of our simulation performed on a single personal computer is compared to that of the work (14) performed with 256× 256 Lattice on parallel computers, the normalised velocity profile and required time steps needed to reach a steady state observed from the two works are in good agreement. Future work will include different geometry settings to study other flow patterns.

ACKNOWLEDGEMENTS

This paper is partial fulfillment of the requirements for the PhD at Walailak University. We thank T na Nagara and two anonymous reviewers for useful comments on the manuscript. We would like to thank Walailak University for financial supports to this work under predoctoral fellowship and the Complex System Key University Research Unit of Excellence (CXKURUE) and under recommendation of National Research Council of Thailand.

REFERENCES

1) Succi S. The Lattice Boltzmann Equation for Fluid Dynamics and Beyond. Oxford University Press, 2001.
2) Guo Z Shi B Wang N. Lattice BGK model for incompressible Navier-Stokes equation. *J Com Phys 2000; 165: 288-306.*
3) Qian Y d'Humieres D Lallemand P. Lattice BGK models for Navier-Stokes equation. *Euro Phys Let 1992; 17: 479-84.*
4) Satofuka N Nishioka T. Parallelisation of Lattice Boltzmann method for incompressible flow computation. *Com Mech 1999; 23: 164-71.*
5) Mei R Luo LS Shyy W. An accurate curved boundary treatment in the Lattice Boltzmann Method**.** *Com Phys 1999; 155: 307-30.*
6) Chen H Chen S Matthaeus WH. Recovery of the Navier-Stokes equation using a Lattice-Gas Boltzmann method. *Phys Rev A 1992; 45: 5339-42.*
7) Chen S Doolen GD. Lattice Boltzmann method for fluid flows. *Ann Rev Fluid Mec 1998; 30: 329-64.*

8) Wolf-Gladrow DA. Lattice-Gas Cellular Automata and Lattice Boltzmann Models: An Introduction, Lecture Notes in Mathematics, Springer-Verlag, 2000.

9) Jaroensutasinee K Jaroensutasinee M. Parallel construction of poincare surface of section method on WAC16P4 cluster. Proceedings of the 6th National Computer Science and Engineering Conference 2002, 217-22.

10) Maeder RE. Computer Science with Mathematica. CUP (USA), 2000.

11) Wolfram S. The Mathematica Book. CUP and Wolfram Media, 1999.

12) Bhatnagar PL Gross EP Krook M. A model for collision processes in gases. 1. Small amplitude processes in harged and neutral one-component systems**.** *Phys Rev 1954; 94: 511-25.*

13) Succi S. The Lattice Boltzmann Equation for Fluid Dynamics and Beyond (Numerical Mathematics and Scientific Computation). Oxford University Press, 2001.

14) Hou SL Zou Q Chen SY Doolen G Cogley AC. Simulation of cavity flow by the Lattice Boltzmann method. *J Com Phys 1995; 118: 329-47*.

บทคัดย่อ

ปานจิต มุสิก[1,2] กฤษณะเดช เจริญสุธาสินี[2]

การจำลองการไหลแบบไม่อัดตัวในโดเมนสี่เหลี่ยมสองมิติโดยวิธีโครงผลึกโบลทซ์มาน

การศึกษาครั้งนี้เป็นการจำลองการไหลแบบไม่อัดตัวในโดเมนสี่เหลี่ยมสองมิติ วัตถุประสงค์ เพื่อพัฒนาวิธีการที่พัฒนามาจากออโตมาตาแบบ Lattice Gas Automata (LGA) ที่ใช้โครงร่างและเวลาที่ไม่ต่อเนื่อง โดยสามารถทำให้คำนวณแบบขนานได้ง่าย โดยเลือกวิธี Lattice Boltzmann Method (LBM) ซึ่งเป็นที่รู้จักว่าเป็นโครงร่างแบบจลนศาสตร์แบบไคเนติกไม่ต่อเนื่อง เพื่อแก้สมการนาเวียร์สโตกส์ (Navier–Stokes equation) ซึ่งมักใช้กับการจำลองของไหลโดยทั่วไป นอกจากนี้ เรากำหนดให้แบบจำลองสองมิติมีโครงร่างแบบสี่เหลี่ยม 9 ทิศทาง (D2Q9 Model) ใช้ในการจำลองความเร็ว โดยให้ความเร็วที่ด้านบนของโดเมนคงที่เป็นเงื่อนไขขอบเขต พบว่า วิธีนี้เป็นวิธีที่มีประสิทธิภาพที่จะสร้างพลวัตของการไหล และผลที่ได้จากการจำลองก็ใกล้เคียงกับงานวิจัยอื่น

[1] สถาบันราชภัฏนครศรีธรรมราช อำเภอเมือง จังหวัดนครศรีธรรมราช 80000

[2] สำนักวิชาวิทยาศาสตร์ มหาวิทยาลัยวลัยลักษณ์ อำเภอท่าศาลา จังหวัดนครศรีธรรมราช 80160