OpenMP performance for benchmark 2D shallow water equations using LBM

Khairul Sabri1,*, Hasbi Rabbani2 and Putu Harry Gunawan3
School of Computing, Telkom University, Jalan Telekomunikasi No. 1 Terusan Buah Batu, Bandung 40257, Indonesia.
E-mail: 1irulsabri@student.telkomuniversity.ac.id; 2hasbirabb@student.telkomuniversity.ac.id; 3phgunawan@telkomuniversity.ac.id
*Corresponding author

Abstract. Shallow water equations or commonly referred as Saint-Venant equations are used to model fluid phenomena. These equations can be solved numerically using several methods, like Lattice Boltzmann method (LBM), SIMPLE-like Method, Finite Difference Method, Godunov-type Method, and Finite Volume Method. In this paper, the shallow water equation will be approximated using LBM or known as LABSWE and will be simulated in performance of parallel programming using OpenMP. To evaluate the performance between 2 and 4 threads parallel algorithm, ten various number of grids \( L_x \) and \( L_y \) are elaborated. The results show that using OpenMP platform, the computational time for solving LABSWE can be decreased. For instance using grid sizes 1000 \( \times \) 500, the speedup of 2 and 4 threads is observed 93.54 s and 333.243 s respectively.

1. Introduction
The fluid flow phenomena such as flow in the rivers, channels, and lake can be described by a known mathematical model which is called shallow water equations (SWE)[1]. Consisting of mass (1) and momentum (2 - 3) equations, SWE in 2-dimension can be given as follows

\[
\frac{\partial t}{\partial t} h = - \left( \frac{\partial x}{\partial x} (hu) + \frac{\partial y}{\partial y} (hv) \right),
\]

\[
\frac{\partial t}{\partial t} (hu) = - \left( \frac{\partial x}{\partial x} (hu^2 + \frac{1}{2} gh^2) + \frac{\partial y}{\partial y} (huv) \right),
\]

\[
\frac{\partial t}{\partial t} (hv) = - \left( \frac{\partial x}{\partial x} (huv) + \frac{\partial y}{\partial y} (hu^2 + \frac{1}{2} gh^2) \right),
\]

where \( h \) denotes the water depth/height, \( u \) the lateral velocity in \( x \) direction, \( v \) the lateral velocity in \( y \) direction, and \( g \) a constant gravitational force. The variables \( x, y \) and \( t \) denote the space and time respectively.

The approximation solution of Equations (1 - 2) can be obtain using several methods. The most popular method is grid method known as the finite volume method (FVM). This method is developed into several numerical schemes which are widely applied and are shown robust scheme, for instance see the references [2, 3, 4, 5] and [6]. However, the deficiency of this method is to solve the general water flow with a complex structure. Here, the LBM is the numerical method
based on particle method. LBM is already applied in many applications of fluid flows [7]. The advantage of LBM is the complexity of boundary condition can be omitted [8]. However, LBM can gain higher computational cost than the other methods.

In order to accelerate the computation in time, the parallel computing techniques will be elaborated. In references [9, 10, 11, 12] and [13], parallel computing is shown as a good idea to accelerate the numerical computation. Thus the goal of this paper is to obtain the performance of parallel computing for simulating LABSWE. Additionally, here the shared parallel architecture using OpenMP platform will be used.

2. Approximation Solution of SWE using LBM

Lattice Boltzmann method (LBM) is the lattice gas automata (LGA) evolved method for solving a discrete computational equation [14]. The characteristics of LBM is composed of three important parts: lattice Boltzmann equation (LBE), lattice pattern which formed by arranged particles, and equilibrium distribution function.

2.1. Lattice boltzmann equation

In the LBM, there are two important processes: streaming step and collision step. On the streaming process, the particles move to another nearby lattice pattern particles in any directions using their velocities which is written by

\[
f_{\alpha}(x + e_{\alpha}\Delta t, t + \Delta t) = f'_{\alpha}(x, t) + \frac{\Delta t}{N_{\alpha}e^{2}}e_{\alpha}F_{i}(x, t),
\]

where \(f_{\alpha}\) is a function for distributing particles, \(f'_{\alpha}\) a distribution function before process of streaming, \(F_{i}\) the force function in \(i\)-direction, \(\Delta x\) the uniform distance of particles over the lattice, and \(\Delta t\) the time step. The velocity is given by \(e = \Delta x/\Delta t\), \(e_{\alpha}\) and a constant \(N_{\alpha}\) is the constant can be determined by the pattern of following lattice

\[
N_{\alpha} = \frac{1}{e^{2}} \sum_{\alpha} e_{\alpha}e_{\alpha i}.
\]

Particles move because the influence of the velocity by another particles in collision step. This step can be described by the following equation

\[
f'_{\alpha}(x, t) - f_{\alpha}(x, t) = \Omega_{\alpha}[f(x, t)].
\]

where \(\Omega_{\alpha}\) is called the collision operator which can control the acceleration of the change of \(f_{\alpha}\). This operator is determined from the microscopic dynamics in generally. Higuera and Jiménez [15] introduce the linearization of collision operator by the equilibrium condition. Therefore, the collision operator can be simplified becomes

\[
\Omega_{\alpha}(f) = -\frac{1}{\tau}(f_{\alpha} - f_{\alpha}^{eq})
\]

where \(\tau\) is defined as relaxation time and \(f_{\alpha}^{eq}\) is known as distribution function of the local equilibrium.

Finally the combination of two steps (streaming and collision) which consists of Equations (4), (6) and (7) becomes following popular formula known as Lattice Boltzmann Equation (LBE),

\[
f_{\alpha}(x + e_{\alpha}\Delta t, t + \Delta t) = f_{\alpha}(x, t) - \frac{1}{\tau}(f_{\alpha}(x, t) - f_{\alpha}^{eq}(x, t)) + \frac{\Delta t}{N_{\alpha}e^{2}}e_{\alpha}F_{i}(x, t).
\]
2.2. Lattice pattern

Lattice pattern is used for representing grid points and determining particle’s motions in LBM. From the lattice pattern, we could easily know that the LBM doing a 1D, 2D, or 3D model. In addition, the constant $N_\alpha$ from Equation (8) is determined by the lattice pattern.

In 1D model, the lattice pattern is depend on the amount of particle speed in lattice points. Generally it is known as the line lattice with two or three speed model. Meanwhile, there are commonly two kinds of lattice patterns in 2D models: the hexagonal and the square lattices. Hexagonal lattice can have 6 or 7-speed models, while square lattice can have 4, 5, 8, or 9-speed models.

According to recent study, the most accurate lattice pattern of all 2D models is square lattice with 9-speed (D2Q9) models [16]. Hence this paper will use D2Q9 to simulate 2D SWE which the model of lattice and its system are shown in Figure 1 below.

*Figure 1.* (a) System of interaction over lattices. (b) The D2Q9 lattice.

In D2Q9 lattice (Figure 1 (b)), each particle moves to its neighbor with certain velocity and direction along the 1-8 links which 0 is indicated as the link with zero speed in the particle. This vector of particle velocity is defined by

$$ e_\alpha = \begin{cases} (0, 0), & \alpha = 0, \\ e\cos(0.25\pi(\alpha - 1)), e\sin(0.25\pi(\alpha - 1)), & \alpha = 1, 3, 5, 7, \\ \sqrt{2}\cos(0.25\pi(\alpha - 1)), \sqrt{2}\sin(0.25\pi(\alpha - 1)), & \alpha = 2, 4, 6, 8. \end{cases} $$

(9)

and the D2Q9 lattice has basic features as follows [14]

$$ \sum_{\alpha} e_{\alpha i} = \sum_{\alpha} e_{\alpha i} e_{\alpha j} e_{\alpha k} = 0, $$

(10)

$$ \sum_{\alpha} e_{\alpha i} e_{\alpha j} - 6e^2\delta_{ij} = 0, $$

(11)

$$ \sum_{\alpha} e_{\alpha i} e_{\alpha j} e_{\alpha k} e_{\alpha l} - 4e^4(\delta_{ij}\delta_{kl} + \delta_{ik}\delta_{jl} + \delta_{il}\delta_{jk}) + 6e^4\Delta_{ijkl} = 0, $$

(12)

where $\Delta_{ijkl} = 1$ in terms of $i = j = k = l$, and $\Delta_{ijkl} = 0$ besides.

Using Equation (9), the value of $N_{\alpha}$ can be found by evaluating Equation (5), yields

$$ N_{\alpha} = \frac{1}{e^2} \sum_{\alpha} e_{\alpha i} e_{\alpha i} = \frac{1}{e^2} \sum_{\alpha} e_{\alpha y} e_{\alpha y} = 6. $$

(13)
Substituting Equation (13) to the Equation (8), leads to
\[ f_\alpha(x + c_\alpha \Delta t, t + \Delta t) = f_\alpha(x, t) - \frac{1}{\tau} (f_\alpha - f_\alpha^{eq}) + \frac{\Delta t}{6c_\alpha^2} c_\alpha F_i(x, t). \]  (14)

Finally, Equation (14) is in the form of LBM which is used for simulating fluid flows.

2.3. Local equilibrium distribution function

Based to the LGA, an equilibrium function is originally obtained from Maxwell-Boltzmann equilibrium distribution function [17, 18]. Here the equilibrium function is expressed as,
\[ f^{eq}_\alpha = P_\alpha + Q_\alpha e_\alpha u_i + R_\alpha e_\alpha e_\alpha j u_i u_j + S_\alpha u_i u_j, \]  (15)

where \( u_i \) and \( u_j \) are the components of flow velocity, or well known as \( u_i = u \) and \( u_j = v \).

Equation (15) becomes a popular equation since this equation is successfully applied for approximating the solution of many flow models [19, 20]. Since the lattice on Figure 2.2 (b) has the same shape as the equilibrium function, thus
\[ P_1 = P_3 = P_5 = P_7 = \bar{P}, \]  (16)
\[ P_2 = P_4 = P_6 = P_8 = \tilde{P}, \]  (17)

likewise the expressions for \( Q_\alpha \), \( R_\alpha \) and \( S_\alpha \). According to Equation (12), the local equilibrium function D2Q9 lattice (Figure 2.2 (b)) can be written as
\[ f^{eq}_\alpha = \begin{cases} P_0 + S_0 u_i u_i, & \alpha = 0, \\ \bar{P} + \tilde{Q} e_\alpha u_i + \tilde{R} e_\alpha e_\alpha j u_i u_j + \tilde{S} u_i u_j, & \alpha = 1, 3, 5, 7, \\ \tilde{P} + \tilde{Q} e_\alpha u_i + \tilde{R} e_\alpha e_\alpha j u_i u_j + \tilde{S} u_i u_j, & \alpha = 2, 4, 6, 8. \end{cases} \]  (18)

Basically, the coefficients \( P_0, \bar{P} \) and \( \tilde{P} \) are defined by a constraint of equilibrium distribution function and are based on the law of conservation such as mass and momentum conservation. Moreover, to get the local equilibrium distribution function in SWE, three conditions should be satisfied,
\[ \sum_\alpha f^{eq}_\alpha(x, t) - h(x, t) = 0, \]  (19)
\[ \sum_\alpha e_\alpha f^{eq}_\alpha(x, t) - h(x, t) u_i(x, t) = 0, \]  (20)
\[ \sum_\alpha e_\alpha e_\alpha f^{eq}_\alpha(x, t) = \frac{1}{2} g h^2(x, t) \delta_{ij} + h(x, t) u_i(x, t) u_j(x, t). \]  (21)

The detail of previous equations can be found in book of Guo Zhou [20] and known as Lattice Boltzmann Method for The Shallow Water Equations (LABSWE).

3. Parallel Architecture

For accelerating the computational time of 2D SWE using LBM, the process will be elaborated by parallel algorithm. Since the existence of patterned particles in the system of LBM, the computation of functions from each grid can be separated and computed in different processor. In this paper, the parallel algorithm will be implemented using an application programming interface (API) OpenMP.

OpenMP is an API that provide multithreading and shared memory [21, 22, 23, 24, 25, 26]. Multithreading is the capability of CPU for running some processes in the same time. This
capability may reduce the runtime of process and increase the efficiency of an algorithm. Moreover, shared memory is the architecture of parallel which allows each running process to gain memory source proportionally. With this feature, access time to the storage can be minimum and accelerate the runtime of process can be increased.

In LABSWE algorithm, the parallelization can be implemented on streaming and collision step for each grids. More detail about the parallel algorithm in LABSWE is shown in this flowchart Figure 2.

![Flowchart](image)

**Figure 2.** The configuration of parallel algorithm for approximating the solution of LABSWE in accelerating the computational time.

In serial process, all of the parameters are declared and stated the initial condition of the water flows. Meanwhile in parallel process, the procedures to compute and update the collision and streaming step of all lattice particles are elaborated.

4. Numerical Results and Parallel Performances

Here, the initial conditions of 2D benchmark LABSWE are given as follows

\[ h(x, y, 0) = 0.05 \]

\[ u(x, y, 0) = v(x, y, 0) = 0 \]

The wall boundary condition is given in this simulation. The wave is generated at the boundary \( x = 0 \) which is

\[ h(x, y, t) = \sum_{\alpha} f_{\alpha}(x, y, t). \]

The results of water height \( h(x, y, t) \) in two final time observations can be seen in Figure 3. It can be seen clearly that the water wave are generated on the boundary \( x = 0 \) and produces the highest wave amplitude 0.053 m. The waves propagate to the right side of domain. Figure 3 at top show the three dimensions view of water surface, meanwhile the slice in two dimensions view is shown in Figure 3 at bottom.
Moreover, the velocity profiles in $x$ and $y$-direction at final time of simulation $t = 30$ and $t = 105$ seconds are given in Figure 4. From Figure 4, there are discrepancy between the velocities $v$ and $u$ in two final time of simulation. The velocity $u$ is shown more fluctuated than the velocity $v$ since the water waves propagate along the $x$-direction.

This paper is focus on the parallel performances of previous simulation. The results of the parallel performance can be found in Table 1. Here, two experiments are given, using 2 and 4 threads. The ten various number of $L_x$ and $L_y$ are given in order to evaluate the parallel performance.

From the Table 1, the OpenMP platform is shown able to accelerate the computational time for various number of grid sizes. In using the large number $L_x = 1000$ and $L_y = 500$, the CPU time of serial code is obtained 463.624 s. Meanwhile the CPU time in parallel code using 2 and 4 threads are observed 393.54 s and 333.243 s respectively. This result produces speedup 1.178 and 1.391 times using 2 and 4 threads respectively. Moreover, the efficiency using 2 threads is obtained 58.9% and using 4 threads is calculated 34.8%.

In this paper, by the experiment using 2 and 4 threads, the speedup for all simulations using 4 threads is observed get the best speedup than using 2 threads. However, in contrast with the speedup, the efficiency using 2 threads is obtained better than using 4 threads.

5. Conclusion
According to the results, the parallel algorithm for solving LABSWE can decrease the execution time for serial algorithm. It means parallel algorithm is successfully applied in this problem. Furthermore, parallel code with 2 threads has lower average speed up than using 4 threads.

Figure 3. The results of water waves which are generated on the boundary $x = 0$ when $t = 30$ (left) and $t = 105$ (right).
Figure 4. Visualization of $u(x, y, t)$ (top) and $v(x, y, t)$ (bottom).

Table 1. The speedup and efficiency of parallel LABSWE algorithm performances using OpenMP.

| Lx  | Ly  | Execution Time (s) | Speed Up | Efficiency (%) |
|-----|-----|---------------------|----------|----------------|
|     |     | Serial 2 Threads 4 Threads 2 Threads 4 Threads 2 Threads 4 Threads |          |                |
| 100 | 50  | 2.782 2.218 0.976 1.254 48.8 31.4 |          |                |
| 200 | 100 | 11.581 7.099 1.199 1.631 59.9 40.8 |          |                |
| 300 | 150 | 28.677 18.705 1.221 1.533 61.1 38.3 |          |                |
| 400 | 200 | 53.363 33.882 1.255 1.575 62.8 39.4 |          |                |
| 500 | 250 | 84.749 53.064 1.258 1.597 62.9 39.9 |          |                |
| 600 | 300 | 123.766 77.579 1.256 1.595 62.8 39.9 |          |                |
| 700 | 350 | 170.399 107.787 1.253 1.581 62.7 39.5 |          |                |
| 800 | 400 | 225.605 142.236 1.252 1.586 62.6 39.7 |          |                |
| 900 | 450 | 291.857 183.981 1.245 1.586 62.3 39.7 |          |                |
| 1000| 500 | 463.624 333.243 1.178 1.391 58.9 34.8 |          |                |

However, the average efficiency of parallel code using 2 threads is higher than using 4 threads. For instance using grid sizes 1000 $\times$ 500, the speedup of 2 and 4 threads is obtained 393.54 s and 333.243 s respectively. Moreover the efficiency of 2 and 4 threads is conducted 58.9% and 34.8 % respectively.
References

[1] Doyen D and Gunawan P H 2014 Finite Volumes for Complex Applications VII-Methods and Theoretical Aspects (Springer) pp 227–235

[2] Delestre O, Cordier S, James F and Darboux F 2008 12th International Conference on Hyperbolic Problems vol 67 (American Mathematical Society) pp 537–546

[3] Delestre O and Marche F 2011 Journal of Scientific Computing 48 41–51

[4] Audusse E, Bouchut F, Bristeau M O, Klein R and Perthame B 2004 SIAM Journal on Scientific Computing 25 2050–2065

[5] Gunawan P H and Lhébrard X 2015 Computers & Fluids 121 44–50

[6] Bouchut F 2004 Nonlinear stability of finite Volume Methods for hyperbolic conservation laws: And Well-Balanced schemes for sources (Frontiers in Mathematics. Birkhäuser Verlag, Basel)

[7] Mohamad A A 2011 Lattice Boltzmann method: fundamentals and engineering applications with computer codes (Springer Science & Business Media)

[8] Mei R, Luo L S and Shyy W 1999 Journal of computational physics 155 307–330

[9] Gunawan P H 2016 Information and Communication Technology (ICoICT), 2016 4th International Conference on (IEEE) pp 1–5

[10] de la Asunci´on M, Castro M, Mantas J and Ortega S 2016 Advances in Engineering Software 99 59–72

[11] De La Asunci´on M, Mantas J M and Castro M J 2011 The Journal of Supercomputing 58 206–214

[12] Brodtkorb A R, Setra M L and Altinakar M 2012 Computers & Fluids 55 1–12

[13] Castillo D, Ferreiro A, García-Rodríguez J A and Vázquez C 2013 Applied Mathematics and Computation 219 11233–11257

[14] Zhou J G 2004 Lattice Boltzmann methods for shallow water flows vol 4 (Springer)

[15] Higuera F and Jimenez J 1989 EPL (Europhysics Letters) 9 663

[16] Skordos P 1993 Physical Review E 48 4823

[17] Chen S and Doolen G D 1998 Annual review of fluid mechanics 30 329–364

[18] He X and Luo L S 1997 Physical Review E 55 R6333

[19] Chen H, Chen S and Matthaeus W H 1992 Physical Review A 45 R5339

[20] Guangwu Y 2000 Journal of Computational Physics 161 61–69

[21] Dagum L and Menon R 1998 IEEE computational science and engineering 5 46–55

[22] Mulyani, Putri N D and Gunawan P H 2017 Information and Communication Technology (ICoICT), 2017 5th International Conference on (IEEE) pp 1–5

[23] Pahlevi M R and Gunawan P H 2017 Information and Communication Technology (ICoICT), 2017 5th International Conference on (IEEE) pp 1–4

[24] Juliati S and Gunawan P H 2017 Information and Communication Technology (ICoICT), 2017 5th International Conference on (IEEE) pp 1–5

[25] Alamsyah M N A, Simanjuntak C A, Bagustara B A R H, Pradana W A and Gunawan P H 2017 Information and Communication Technology (ICoICT), 2017 5th International Conference on (IEEE) pp 1–6

[26] Iryanto and Gunawan P H 2017 Information and Communication Technology (ICoICT), 2017 5th International Conference on (IEEE) pp 1–5