Parallel simulation of dam-break flow by OpenMP-based SPH method

Zhao Luo\textsuperscript{1,2,*}, Qihe Wu\textsuperscript{1,2} and Lei Zhang\textsuperscript{1,2}

\textsuperscript{1} CCC Second Harbor Engineering Co., Ltd. National Enterprise Technology Center, Hubei Wuhan, China
\textsuperscript{2} CCC Highway Bridges National Engineering Research Centre Co., Ltd., Beijing, China

*Corresponding author e-mail: 490458835@qq.com

Abstract. Smoothed particle hydrodynamics (SPH), a Lagrangian mesh-free particle numerical method, is suitable for simulating strong impact and large deformation problems. In the method, quantities of particles can ensure high precision. However, with the increase of the particle numbers, the calculation efficiency becomes a challenge for applying the method to the engineering practice. OpenMP, a Portable Shared Memory Parallel Programming, is a great solution to improve the efficiency of SPH algorithm. In the paper, dam-break flow is simulated by SPH method. Vortex centre is discovered, the role of numerical technique is compared. At the same time, two parallel schemes for SPH algorithm is introduced, the speedup ratio with respect to number of particles or threads are revealed.

1. Introduction

Smoothed particle hydrodynamics (SPH) is a Lagrangian mesh-less method originally developed to deal with astrophysics problems [1, 2]. Without requirement for generating mesh and capturing free surface, SPH is successfully extended to a variety of fluid-dynamic systems [3] and multiphase flow system [4, 5]. As an interpolation method, the accuracy of the SPH method often depends on the number of particles in the support domain. In theory, when the number of particles approaches infinity, the computational accuracy will be very close to reality. However, as an adaptive method, SPH has to run a large amount of computation at each time step, which is quite time-consuming and limit the development of the method.

Parallel computing, also known as high performance computing, can coordinate multiple processor to solve one problem together. For example, if your computer has 8 processors, the speedup would be 8X after parallel computing in ideal, but actually it can’t be so high because of data exchange. OpenMP, a standard for implementing the thread model by adding directives to the code, has been applied in a number of fields [6, 7]. Compared with MPI or GPU, OpenMP is more cheap and easy to parallelize an existing serial code [8]. SPH algorithm can be paralleled by means of several OpenMP directives. These directives serve as instructions for a compiler which blocks and loops in the source code are to be executed in parallel by available processors. OpenMP directives can divide the loop into several blocks, which are computed by different processors. In the paper, the processor is Intel (R) Xeon (R) CPU E3-1230 V3, four core and eight thread processor, dominant frequency 3.30GHZ.
computer memory 8G. The software platform for parallelizing program execution is Microsoft Visual Studio 2012, and the compiler system is the Intel Fortran compiler.

In SPH algorithm, the loop statements can be divided into two categories, i.e. particles number loop and particle pairs number loop. The parallel schemes for the loop statements are introduced in the paper, and computational efficiency is studied. At the same time, dam-break flow is simulated by OpenMP-based SPH method, the pressure evolution on the wall is studied.

2. Basic model

2.1. SPH formulation

In the model, the water satisfies balance laws for conservation of mass

\[
\frac{\partial \rho}{\partial t} = -\rho \nabla \cdot \mathbf{u},
\]  

(1)

And momentum

\[
\rho \frac{D\mathbf{u}}{Dt} = \nabla \cdot \mathbf{\sigma}.
\]  

(2)

In the SPH method, the problem domain is dispersed into many particles. Every particle holds physical characteristics such as volume, mass, and field function information. A field function of particle can be approximated by summation over the neighboring particles in the support domain, as

\[
f_a = \sum_{b \in P} V_b f_b w_{ab},
\]  

(3)

so as the derivative of field function

\[
\nabla f_a = \sum_{b \in P} V_b (f_a + f_b) \cdot \nabla w_{ab},
\]  

(4)

Where \( f_a \) and \( f_b \) is the field function of particle \( a \) and \( b \), \( V_b \) volum of particle \( b \), and \( P \) assemblage of all particles. The equation (4) is the classic symmetric form of SPH formulation. \( w_{ab} \) is called smoothed kernel function with symmetry and normalization. With the increasing of distance between two particles, the value of smoothed kernel function decreases, given by

\[
w(r, h) = \begin{cases} 
\alpha_d \left( 1 - \frac{q}{2} \right) (2q + 1), & 0 \leq q \leq 2 \\
0, & q > 2
\end{cases}
\]  

(5)
where \( q = \left| r_{ab} \right| / h; r_{ab} = r_a - r_b \). \( r_a \) and \( r_b \) are position vectors of particles \( a \) and \( b \), and \( h \) is smoothing length. \( \alpha_D \) is a parameter to ensure the normalization of the smoothed kernel function, equal to \( 7 / (4\pi h^2) \) in two-dimension model.

In SPH method, the water is regarded as weak compressible fluid, the water pressure is related to density by equation of state, given by

\[
p = \frac{\rho_0 c_0^2}{\xi} \left[ \left( \frac{\rho}{\rho_0} \right)^\xi - 1 \right],
\]

where \( \rho_0 \) is initial density of water, a constant 1000 kg/m\(^3\); \( c_0 \) is artificial sound speed, depends on the case. The continuity equation and momentum equation can be dispersed. Numerical techniques, such as artificial viscosity, MLSPH and XSPH, are also added in the model to make sure a great precision. Besides, ghost particles boundary treatment method is applied to the boundary. A brief introduction to SPH method is presented in this section and detail can be found in [4].

2.2. **OpenMP parallel scheme**

In the SPH method, the problem domain is dispersed into many particles. Every particle has a special number, like the number as shown in the fig.1. Every particle has a support domain, like the circle as shown in the fig.1. In the support domain, particle pair will be formed by the particle and another one, like the \( p_1, p_2, p_3 \) in the figure. In SPH algorithm, the loop statements can be divided into two categories, i.e. particles numbers loop and particle pair’s numbers loop.

![Fig.1 particle number and particle pair number](image)

2.2.1. **Particles numbers loop.** Link-list search algorithm, a popular nearest neighboring particle searching (NNPS), has high efficiency, but still takes about half computing time. So, parallelization of link-list algorithm is imperative. In the particles numbers loop, we divided the particles according to the number of threads at first. Then variables, which are called and assigned at the same time, are defined as private variables. Those private variables can be calculated separately in different processors, thus avoiding data conflict. The private data will be accumulated before ending parallel. And we will
organize memory of data for the subsequent calculations. Flow chart of parallel code design as shown in fig.2(a).

2.2.2. Particles pair’s numbers loop. Continuity equation and momentum equation are calculated by the particles pair’s numbers loop, which is complicated. Every particles pair has two particles, and two different particles pairs could have same particle. It is no problem in the serial computing. But in parallel calculation, the particle information may be assigned and called simultaneously by multiple processors, which is forbidden. For example, when the first thread is going to take the function value of a particle, the value may have been modified by another thread. To solve the problem, we define the function as a private variable and accumulate the field function from different threads before ending parallel. Flow chart of parallel code design as shown in fig.2 (b).

3. Simulation and results analysis
In this section, the SPH method are validated in the simulation of dam-break and impact problem. The effect of numerical techniques, like artificial viscosity, MLSPH and XSPH, is discussed. The pressure evolution on the wall is studied.

(a) Parallel of Link-list search algorithm (b)parallel of field function calculation

Fig.2 Flow chart of parallel code

Fig.3 Dam-break model
3.1. **Dam-break flow against a vertical wall**

As sketched in Fig.3, in the 3.22m×1.8m tank, 1.2m×0.6m water column exists in the lower left corner. After computing, the water column begins to move under gravity. The point on the right wall is pressure probe.

Fig.4 displays snapshots of the flow evolution up to the plunging wave closure and the subsequent generation of a cavity. At $t=0.63s$, water begins to impact the wall and move upward. Under the force of the wall, the water has left velocity. Then, jet flow is formed as shown in the fig.4(c). Finally, impacting the surface of the water, the jet flow splashes water around and forms holes.

Fig.5 displays the evolution of velocity vector when the dam-break flow impacts the wall. The water particle move anticlockwise as a whole, but the water particle near the bottom right corner moves clockwise and forms vortex. As time goes on, vortex center increase gradually, which we can see from the circle in the figure.

Fig.6 is the simulation profile at 1.4s, which shows the role of numerical techniques. Artificial viscosity, XSPH technology and MLSPH technology, have reduce numerical oscillations from acceleration, velocity and density three respects. The combined application of the three numerical techniques can simulate dam break flow more accurately.

![Fig.4 Snapshots of the dam-break flow against a vertical wall.](image-url)
Fig. 5: Velocity vector and vortex evolution

(a) \( t = 1.0 \text{s} \)  
(b) \( t = 1.2 \text{s} \)  
(c) \( t = 1.4 \text{s} \)  
(d) \( t = 1.6 \text{s} \)

Fig. 6: The role of numerical techniques

(a) Without artificial viscosity  
(b) Without XSPH  
(c) Without MLSPH  
(d) With artificial viscosity, XSPH, MLSPH
Fig. 7 displays the Comparison between the pressure loads measured experimentally and predicted by the numerical model. Simulation result is basically in line with experiment value. However, numerical oscillation problem still occurs in the strong impact, especially in the early stage of contact. The numerical techniques can help to decrease oscillation at the cost of increasing energy dissipation, which the pressure is lower than the experimental value. We think the oscillation problem can be eliminated by optimized boundary treatment.

![Comparison between the pressure loads measured experimentally and predicted by the numerical model](image)

### 3.2. Parallel result analysis

For an OpenMp parallel program, computation time is generally consumed in the following three sections: serial computing, parallel computing and data transfer between different processors. Even in parallel program, there must be many serial statements, such as input and output statements. In addition to the serial part, the parallel efficiency is decided by the time consumed by parallel computation and the data transfer between different processors. In general, the more threads in parallel, the less time consumed by parallel computing, but the more time it takes to transfer data between different threads. This means that more processors may have lower computational efficiency. The parallel efficiency is often determined by the speedup ratio $S_p$ and the efficiency of the individual processor $E_p$, as defined below:

$$S_p = \frac{T_1}{T_p}; \quad E_p = \frac{S_p}{p},$$

Where subscript refers to the number of processors. $T_1$ is the time cost by serial computing? $T_p$ is the time cost by parallel computing under p threads? Changing the number of particles or threads, we can get different speedup radio.
Fig. 8 $S_p$ evolution with respect to the number of threads

Fig. 9 $E_p$ evolution with respect to the number of threads

Fig. 8 displays the speedup ratio evolution with respect to the number of threads. When the number of particles is constant, the speedup ratio is directly proportional to number of threads. When the number of threads is constant, the speedup ratio is directly proportional to number of threads within certain range. When the number of particles is too large, such as 2,880,000, computer may consume too much time in data transferring between different threads, resulting in a low parallel efficiency.

Fig. 9 displays the efficiency of the individual processor evolution with respect to the number of threads. The efficiency of a single processor always decreases as the number of threads increases, because of the time consuming of data transferring.

4. Conclusion
In the paper, the dam-break flow is simulated successfully by SPH method. Jet water and cavity have been captured accurately. By observing velocity vector of particles, vortex center near the wall has been discovered. The role of numerical techniques such as artificial viscosity, MLSPH and XSPH have been studied, and combined application of the three numerical techniques can achieve better results. Compared with the pressure loads measured experimentally, the pressure on the wall predicted by the numerical model is basically consistent.

Two parallel schemes for SPH algorithm is introduced, and parallel efficiency and the efficiency of single thread have been studied. The speedup ratio is basically proportional to number of threads within certain range. However, when the number of particles is too large, the parallel efficiency will be poor.
References

[1] L.B. Lucy, A numerical approach to the testing of the fission hypothesis, J. The astronomical journal. 82 (1977): 1013-1024.

[2] R.A. Gingold, J.J. Monaghan, Smoothed particle hydrodynamics: theory and application to non-spherical stars, J. Monthly notices of the royal astronomical society, 181.3 (1977): 375-389.

[3] Z. Luo, C. Wang, Unified semi-analytical wall boundary treatment in SPH and irregular particle distribution, J. Chinese Journal of Hydrodynamics, 32.2 (2017): 189-197.

[4] C. Wang, Y. Wang, C. Peng, et al, SPH simulation of water-soil mixture flows, J. ASCE Journal of Hydraulic Engineering, 142.10 (2016): 04016032.

[5] C. Wang, Y. Wang, C. Peng, et al, Two-fluid smoothed particle hydrodynamics simulation of submerged granular column collapse, J. Mechanics Research Communications, 79 (2017): 15-23.

[6] D. Cheng, W.H. Wang, Application of OpenMP in SAR Image Processing, J. Computer science. 44 (2017): 161-163, 187.

[7] J. Guan, S. Yan, J.M. Jin, An OpenMP-CUDA Implementation of Multilevel Fast Multipole Algorithm for Electromagnetic Simulation on Multi-GPU Computing Systems, J. IEEE Transactions on Antennas & Propagation, 61.7 (2013): 3607-3616.

[8] S. Yeylaghi, B. Moa, P. Oshkai, et al, ISPH modelling of an oscillating wave surge converter using an OpenMP-based parallel approach, J. Journal of Ocean Engineering & Marine Energy. 2.3 (2016): 1-12.