*ICIC Express Letters*
*Part B: Applications*
Volume **9**, Number **7**, July **2018**

ICIC International ©2018 ISSN 2185-2766
pp. **665–672**

# SIMULATION OF PLATELET AGGREGATION ON THE WALL OF ORIFICE FLOW BY DISSIPATIVE PARTICLE DYNAMICS ACCELERATED BY OPENMP

Yingming Yi and Masaaki Tamagawa

Department of Biological Functions Engineering
Graduate School of Life Science and Systems Engineering
Kyushu Institute of Technology
Kitakyushu, Fukuoka 808-0196, Japan
yi-yingming@edu.life.kyutech.ac.jp

Received December 2017; accepted March 2018

Abstract. *Platelet transport and aggregation are two important steps in the formation of white thrombus. Platelet transport has been investigated by finite difference method (FDM) in our previous research. However, FDM cannot simulate platelet aggregation. Other numerical methods, such as dissipative particle dynamics (DPD), should be used to simulate platelet aggregation. And DPD can be combined with FDM to have a complete analysis of the formation of white thrombus. However, vast computational time in DPD system is needed if particle number is big. To decrease this vast computational time and have a balance with FDM, parallel computation is an effective method. This paper describes the parallelization of DPD by OpenMP. For parallelization performance, it is found that the computational time is efficiently shortened by OpenMP. After the confirmation of the high performance by OpenMP, platelet aggregations with different initial platelet numbers are simulated in orifice flow. It is concluded that recirculation area and reattachment point have higher probability of platelet aggregation.*
**Keywords:** DPD, OpenMP, Parallel performance, Platelet aggregation

1. **Introduction.** It is usual that white thrombus happens in artificial blood pumps. The main composition of white thrombus is platelet. Platelet transport and aggregation are the two main steps in thrombus formation. Platelet transport has been investigated by finite difference method (FDM) in our previous research [1]. However, FDM cannot simulate platelet aggregation. Other numerical methods, such as dissipative particle dynamics (DPD), should be used to simulate platelet aggregation. And DPD can be combined with FDM to have a complete analysis of the formation of white thrombus. Compared with FDM, much larger computational time is needed by DPD. In order to have a balance of the computational time with FDM, Open Multi-Processing (OpenMP) is used to parallelize the DPD for acceleration.

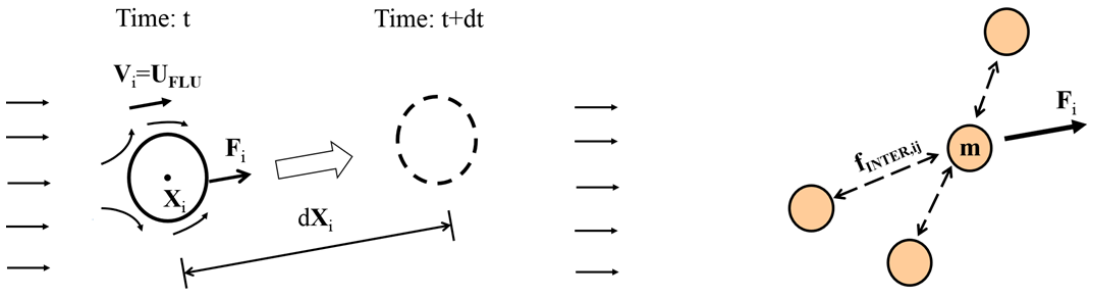
A. Tosenberger et al. [2] used dissipative particle dynamics method to model different stages of platelet adhesion process. And their simulation results are in a good agreement with the experimental results. N. Filipovic et al. [3] suggested that dissipative particle dynamics offers a promising approach to the modelling of platelet-mediated thrombosis. Pivkin and Richardson [4] used DPD to investigate the effect of the presence of red blood cells on platelet aggregation. However, time consumption is one disadvantage of DPD. N. Zhang et al. [5] completed the simulation of 1-*ms* multi-scale phenomena of flowing platelets in blood vesssels within approximate 37 days. Eckstein and Belgacem [6] used hundreds of thousands of hours to simulate platelet aggregation even in low number density of platelets.

DOI: 10.24507/icicelb.09.07.665

Previous researchers have developed several methods to accelerate their computation of DPD. In hardware, H. Wu et al. [7] used graphics processing unit (GPU) for the acceleration. Their conclusion was that the implementation running on single GPU can be more than 20x faster than conventional implementation running on single CPU core. In software, N. Goga et al. [8] presented the message passing interface (MPI) parallelization of the DPD and discussed its main characteristics and performances.

Usually, MPI requires an explicit reconstruction of the code in order to expose parallelization and also needs to insert communication routines at the appropriate points. So MPI requires the programmers to understand its details. However, one of the other parallelization methods, OpenMP provides programmers a simple and flexible interface for developing parallel applications. In OpenMP, the programmers only need to simply add directives to the sequential version. The lower requirement makes OpenMP more convinent than MPI. First released in 1997, OpenMP is an implementation of multithreading, with the runtime environment allocating threads to different processors [9]. A. Amritkar et al. [10] compared OpenMP with MPI based parallelization during the simulation of dense particulate systems. They found that OpenMP is between 50-90% faster than MPI. F. Broquedis et al. [11] proposed a new, hierarchical approach to the execution of OpenMP threads onto multicore machines and achieved excellent speedups on 16-core machines.

In this paper, dissipative particle dynamics is used to simulate platelet aggregation in orifice flow. To overcome the demerit of time consumption, OpenMP is used to parallelize the DPD. The parallelization performance will be checked. After the confirmation, platelet aggregations with different initial platelet numbers are simulated in orifice flow. And the aggregation on the wall is evaluated.




(a) Particle motion in the fluid (b) Force exerted on particle

FIGURE 1. Particle motion and its force

2. **Computational Method.** Dissipative particle dynamics is a kind of particle method. So variables like velocity, force and position should be computed to trace particles in DPD system. One particle in the fluid and the relevant variables are shown in Figure 1. And computations of velocity, force and position are three steps in solving the DPD problem. In DPD [12], velocity vector of particles ($\boldsymbol{V}_i$) is obtained from a local, volume-averaged fluid velocity. And $\boldsymbol{V}_i$ is computed by the following STEP 1 of the Verlet algorithm for the integration of the DPD:

$$\text{STEP 1:} \qquad \boldsymbol{V}_i = \boldsymbol{U}_{\text{FLU}}, \tag{1}$$

where $\boldsymbol{V}_i$ is the velocity vector of the $i$th particle, and $\boldsymbol{U}_{\text{FLU}}$ is the fluid velocity. The force exerted on the $i$th particle ($\boldsymbol{F}_i$) is the sum of the interaction force with other particles. And $\boldsymbol{F}_i$ is calculated by the following STEP 2:

$$\text{STEP 2:} \qquad \boldsymbol{F}_i = \sum_{j=1}^{N_{\text{T}}} \boldsymbol{f}_{\text{INTER},ij}, \tag{2}$$

where $N_{\mathrm{T}}$ is the total number of particles in the computational region. If position vectors of particles are $\boldsymbol{X}_i$ and $\boldsymbol{X}_j$, and relative position vector between the two particles is $\boldsymbol{r}_{ij}$, the relationship between $\boldsymbol{X}_i$, $\boldsymbol{X}_j$ and $\boldsymbol{r}_{ij}$ is

$$\boldsymbol{X}_j = \boldsymbol{X}_i + \boldsymbol{r}_{ij}. \tag{3}$$

The interaction force term ($|\boldsymbol{f}_{\mathrm{INTER},ij}|$) is a piecewise function:

$$|\boldsymbol{f}_{\mathrm{INTER},ij}| = \begin{cases} -2\alpha\left(1 - \dfrac{r_{ij}/a}{R}\right) & \text{if } r_{ij}/a \le R, \\ 0 & \text{if } R \le r_{ij}/a \le L, \\ \alpha\left(\dfrac{r_{ij}/a}{L} - 1\right) & \text{if } L \le r_{ij}/a \le M, \\ \alpha\left(\dfrac{M}{L} - 1\right)\dfrac{B - r_{ij}/a}{B - M} & \text{if } M \le r_{ij}/a \le B, \end{cases} \tag{4}$$

where $r_{ij} = |\boldsymbol{r}_{ij}|$, $\alpha$ is the force coefficient, given to be $4.0\times10^{-9}$N, and $a$ is platelet radius. In Equation (4), $R$, $L$, $M$ and $B$ are dimensionless parameters which controls the range of the interaction force. In Equations (2) and (4), $\boldsymbol{f}_{\mathrm{INTER},ij}$ is directed along the normal to the wall or the vector connecting the centers of particles. The motion of each particle is described by Newton's law equation. And the computation of the displacement ($\mathrm{d}\boldsymbol{X}_i$) is STEP 3:

$$\text{STEP 3:} \qquad \mathrm{d}\boldsymbol{X}_i = \boldsymbol{V}_i \mathrm{d}t + \frac{1}{2}\frac{\boldsymbol{F}_i}{m}\mathrm{d}t^2. \tag{5}$$

In Equation (5), $m$ is the mass of single particle, and $\mathrm{d}t$ is the time step, set to be $1.0\times10^{-6}$s.

Orifice flow is used for analysis in this computation. For the computation of flow field, boundary conditions, turbulent model and computational region are described in our previous work [1]. Figure 2 shows the streamline in orifice flow. For the computation of particle transport, 5,000 particles are uniformly set in the flow field and cyclic boundary condition is used. Figure 3 shows the position of particles in orifice flow at 0.0s and 0.02s.

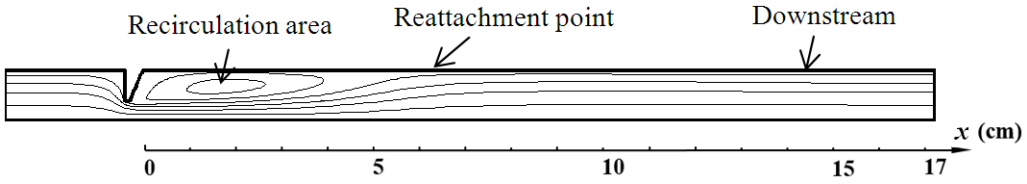
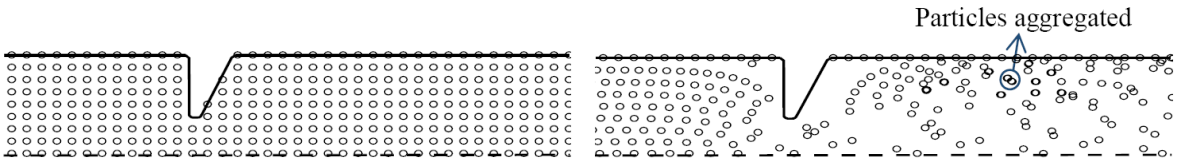



FIGURE 2. Streamline in orifice flow




(a) 0.0s (b) 0.02s

FIGURE 3. Position of particles in orifice flow at 0.0s and 0.02s

It is found that particles transport in the flow, and some particles aggregate and become a cluster during the flow.

Figure 3 shows the transport of particles in orifice flow. However, the 0.02s' simulation took about 0.8 hours on a computer with Intel core (TM) i7, 3.40 GHz CPU, 64G RAM, and Ubuntu 14.04 64bit operating system. And the computational time needed in each

TABLE 1. Computational time in each step

| | Computational time (s) | Occupation ratio (%) |
|---|---|---|
| STEP 1 | 0.020 | 6.1 |
| STEP 2 | 0.304 | 92.7 |
| STEP 3 | 0.004 | 1.2 |

```
PROGRAM DPD
… …
DO I=0, NT
  DO J=0, NT
    F(I,J)=F_INTER
  CONTINUE
 CONTINUE
… …
END
```

(a) Serial code

```
PROGRAM DPD
include 'omp_lib.h'      //library routine
… …
call cpu_time(t1)        //test of CPU time before time loop
DO T=T_START, T_END      //time iteration
c$omp parallel           //parallel statement before do loop
c$omp do                 //parallelization of do loop
 DO I=0, NT              // I iteration
  DO J=0, NT             // J iteration
    F(I,J)=F_INTER       // computation of interaction force between
  CONTINUE                  i^th and j^th platelet
 CONTINUE
c$omp end do
c$omp end parallel
CONTINUE
call cpu_time(t2)        // test of CPU time after time loop
… …
END
```

(b) Parallel code

FIGURE 4. Parallelization of DPD

step is tested and shown in Table 1. From Table 1, it is found that the code spends most of the time (92.7%) on STEP 2, that is the computation of the force. In our code, the force exerted on the $i$th particle is computed by the sum of the interaction force between the $i$th particle and all the other particles. And the force exerted on all particles should be calculated. However, the velocity and displacement of single particle are computed by using its own variables, without variables of other particles. If the number of particles is $N_{\mathrm{T}}$ in the system, the computation of force for all particles should be done for $N_{\mathrm{T}}^2$ times. And the computation of velocity and displacement should be done for $N_{\mathrm{T}}$ times. Then the algorithm of STEP 1 and STEP 3 is linear time, and the algorithm of STEP 2 is subquadratic time. It is thought to be the reason of time consumption in STEP 2. In our simulation, OpenMP is used to parallelize STEP 2 to decrease the computational time.

OpenMP in our code is used as shown in Figure 4. Figure 4(a) is the serial code, and Figure 4(b) is the parallel code. Library routine, parallel statement and parallelization of do loop are used to realize the parallelization. The library routine includes external procedures which can be used for directed parallel decomposition, and parallel statement is used to declare the part that will be parallelized. And *call cpu_time* is used to test the CPU time.

The parallel part is used for the computation of the interaction force between the $i$th and $j$th platelet. It consists of two DO loops. For the computation of the force of the $i$th particle, variables of other particles are not needed. It is thought that one DO loop

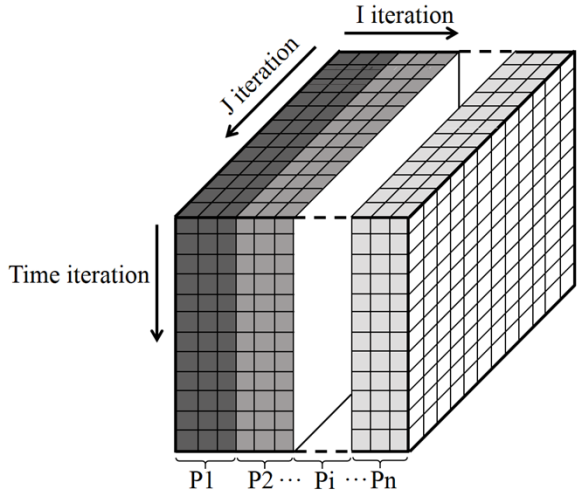



FIGURE 5. Parallel processing ($P_i$ means the $i$th processor)

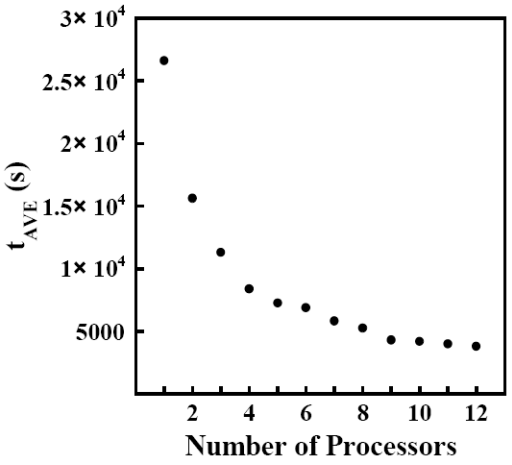



FIGURE 6. Averaged CPU time vs. number of processors

is independent. So OpenMP can divide this DO loop into several DO loops as shown in Figure 5. The computation in each DO loop can be done in each CPU processor.

## 3. Results and Discussion.

3.1. **Test case and precomputation.** In this paper, OpenMP is used to parallel the DPD to shorten the computational time. In order to check the performance of OpenMP, call cpu_time ($t_1$) and call cpu_time ($t_2$) are used to test the CPU time at the beginning and at the end of the computation as shown in Figure 4(b). Then $t_2 - t_1$ is the total CPU time for computation by all CPU processors. And the averaged CPU time ($t_{\text{AVE}}$) is calculated by

$$t_{\text{AVE}} = \frac{t_2 - t_1}{N_{\text{PROC}}}, \tag{6}$$

where $N_{\text{PROC}}$ is the number of processors used in the parallelization. Figure 6 shows its performance. In the graph, the $x$-axis represents the number of processors used in the parallelization, and the $y$-axis represents the averaged CPU time. It is found that the averaged CPU time decreases when the number of processors increases. Table 2 shows the averaged CPU time under different number of processors and the ratio to the time in serial code. It is found that the code parallelized by 12 processors need only 14.3% of the time in serial computation. Speedup is also computed, and it is defined to be

$$S = \frac{t_{\text{SER}}}{t_{\text{PARA}}}. \tag{7}$$

In Equation (7), $S$ is speedup, $t_{\text{SER}}$ is the CPU time in serial computation, and $t_{\text{PARA}}$ is the averaged CPU time when the code is parallelized. Figure 7 shows the relationship between number of processors and speedup. It is found that the speedup increases when the numbers of processors increase. And speedup reaches 7.1 when 12 processors are used.

TABLE 2. Relationship between number of processors and averaged CPU time

| | Parallel comp. | | | | Serial comp. |
|---|---|---|---|---|---|
| Number of processors | 1 | 2 | 6 | 12 | 1 |
| $t_{\text{AVE}}$(s) | 26605 | 15623 | 6890 | 3857 | 26972 |
| Ratio to the serial comp. | 98.6% | 57.9% | 25.5% | 14.3% | 100.0% |

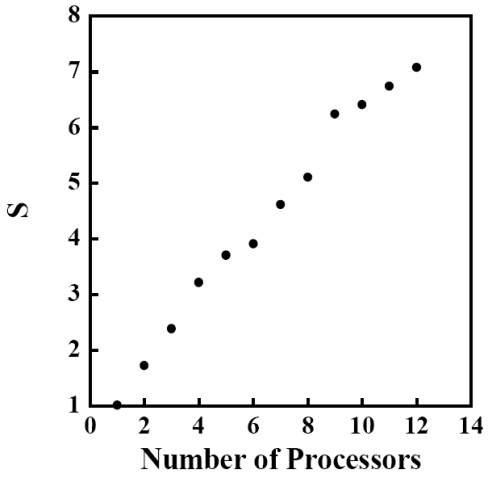



FIGURE 7. Speedup vs. number of processors

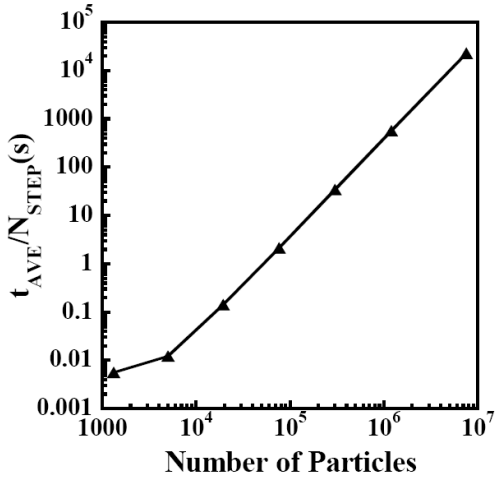



FIGURE 8. Averaged CPU time vs. number of particles

3.2. **Main results.** Previously, the performance of OpenMP has been discussed. The first topic of this section is the CPU time needed for computation. The second topic is aggregation process. This aggregation process is composed of averaged force and aggregation number (PDF). Platelet aggregates on the wall when the platelet interacts with the already aggregated platelets. So the aggregation number means the number of platelets in the aggregations.

In the test case, the number of particles is about 5,000. And the number of particles under real concentration ($2.0\times10^8$/ml) in the scale of orifice flow is $7.5\times10^6$. And the relationship between the number of particles and $t_{\rm AVE}/N_{\rm STEP}$ is shown in Figure 8. $t_{\rm AVE}$ is the averaged CPU time, and $N_{\rm STEP}$ is the number of time iterations. It is found that the code needs about 144 years to complete 0.2s' simulation in the case of real platelet concentration. In order to avoid this vast time, the number of particles was decreased to control the computational time within one week. And it is interesting to discuss the effect of the decrease of the number of particles. Fourteen cases with different particle numbers in the fluid are used in the simulation. The numbers of particles are from 100 to 36,000.

As for aggregation process, averaged force and aggregation number (PDF) are computed. Averaged force is defined to be

$$F_{\rm AVE} = \frac{\sum_{i=1}^{N_{\rm T}} F_i}{N_{\rm T}}, \tag{8}$$

where $F_{\rm AVE}$ is the averaged force in DPD system, $F_i$ is the force exerted on the $i$th particle, and $N_{\rm T}$ is the total number of particles in the system. Figure 9 shows the time history of $F_{\rm AVE}$ when the numbers are 110, 1500, 24000 and 33000. It is found that the force increases at first, and then decreases to the stable value before about 0.18s. And the stable averaged force is calculated. Relationship between number of particles and the stable averaged force is shown in Figure 10. It is found that the averaged force increases when number of particles increases. However, the increase gradient becomes lower and lower. And $F_{\rm AVE}$ tends to be constant when the number of particles is above $2.0\times10^4$. It is thought to be convergent.

Next is platelet aggregation. Figure 11 shows the distribution of probability distribution function (*PDF*) on the wall when numbers of platelets are 24,000 and 33,000. The aggregation number at a point of the wall describes the probability of platelet aggregation

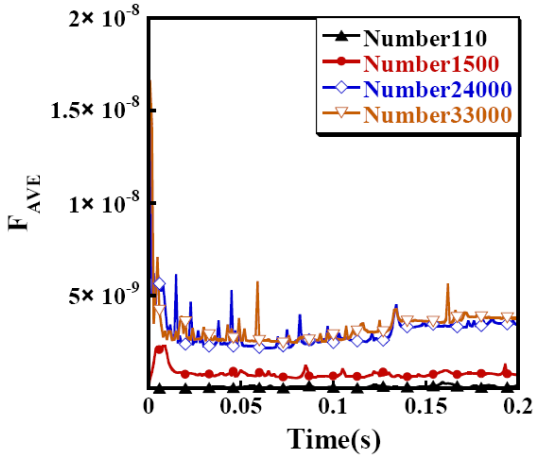



FIGURE 9. Time history of averaged force

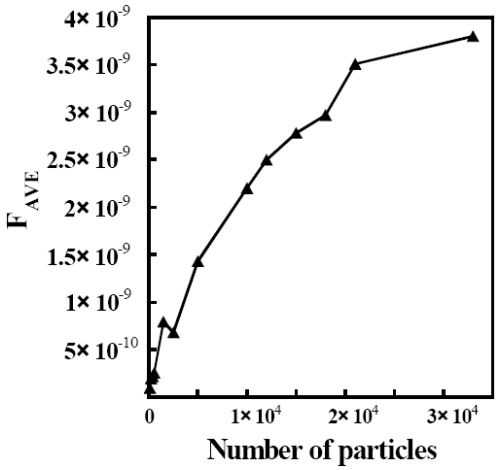
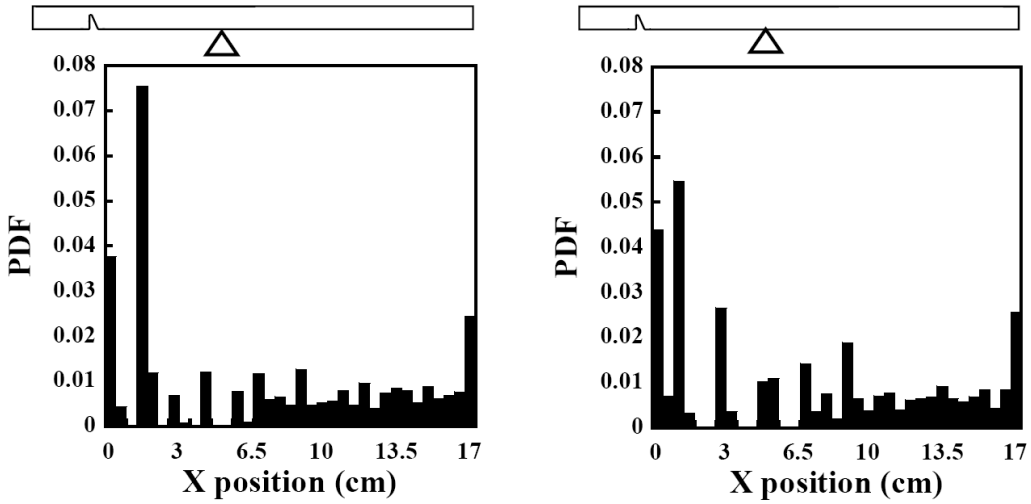



FIGURE 10. Number of particles vs. averaged force




(a) Particle number: 24,000 (b) Particle number: 33,000

FIGURE 11. Distribution of probability distribution function on the wall

at the point. *PDF* is defined to be

$$PDF = \frac{Num(x)}{N_{\mathrm{T}}}, \tag{9}$$

where $Num(x)$ is aggregation number on the wall at position $x$. It is found that aggregation probability at the recirculation area and the reattachment point is higher than the probability downstream. It is also found that for both cases, less than 8% of all particles will aggregate at a position on the wall.

4. **Conclusions.** This paper parallelizes dissipative particle dynamics by OpenMP. It is found that computational time is efficiently shortened. It is also found that recirculation area and reattachment point have higher probability of platelet aggregation.

As the CPU time for DPD can be shortened, we will try to establish the hybrid approach combining DPD and FDM to obtain a prediction method of thrombosis with high accuracy and high performance in the orifice flow.

**Acknowledgment.** We acknowledge the OpenMP Architecture Review Board for their providing this free software. A part of this work was supported by Grant-in-Aid for Scientific Research on Innovative Areas 15H01601 and Challenging Research (Exploratory) 17K18844.

## REFERENCES

[1] Y. Yi, M. Tamagawa and W. Shi, Prediction of thrombus formation on the wall by high shear rate on Couette and orifice blood flows, *Journal of Medical Imaging and Health Informatics*, vol.7, no.1, pp.79-84, 2017.

[2] A. Tosenberger, F. Ataullakhanov, N. Bessonov, M. Panteleev, A. Tokarev and V. Volpert, Modelling of thrombus growth and stopping in flow by the method of dissipative particle dynamics, *Russian Journal of Numerical Analysis and Mathematical Modelling*, vol.27, no.5, pp.507-522, 2012.

[3] N. Filipovic, M. Kojic and A. Tsuda, Modelling thrombosis using dissipative particle dynamics method, *Philosophical Transactions of the Royal Society*, vol.366, pp.3265-3279, 2008.

[4] I. V. Pivkin and P. D. Richardson, Effect of red blood cells on platelet aggregation, *Engineering in Medicine and Biology Magazine*, vol.28, no.2, pp.32-37, 2009.

[5] N. Zhang, P. Zhang, L. Zhang, X. Zhu, L. Huang and Y. Deng, Performance examinations of multiple time-stepping algorithms on stampede supercomputer, *Proc. of the 2015 XSEDE Conference: Scientific Advancements Enabled by Enhanced Cyberinfrastructure*, St. Louis, MD, USA, 2015.

[6] E. C. Eckstein and F. Belgacem, Model of platelet transport in flowing blood with drift and diffusion terms, *Biophysical Journal*, vol.60, no.1, pp.53-69, 1991.

[7] H. Wu, J. Xu, S. Zhang and H. Wen, GPU accelerated dissipative particle dynamics with parallel cell-list updating, *IEIT Journal of Adaptive & Dynamic Computing*, vol.1, no.2, pp.33-42, 2011.

[8] N. Goga, H. Berendsen, S. Baoukina, S. A. Moga, G. Dragoi, A. Hadar and B. Paviloiu, MPI parallelization of innovative DPD thermostats, *IEEE International Conference on Healthcare Informatics*, Philadelphia, PA, USA, p.476, 2013.

[9] *OpenMP*, http://www.openmp.org, 2015.

[10] A. Amritkar, S. Deb and D. Tafti, Efficient parallel CFD-DEM simulations using OpenMP, *Journal of Computational Physics*, vol.256, no.1, pp.501-519, 2014.

[11] F. Broquedis, O. Aumage and B. Goglin, Structuring the execution of OpenMP applications for multicore architectures, *Proc. of the 25th IEEE International Parallel & Distributed Processing Symposium*, Atlanta, GA, USA, 2010.

[12] I. V. Pivkin, P. D. Richardson and G. Karniadakis, Blood flow velocity effects and role of activation delay time on growth and form of platelet thrombi, *Proc. of the National Academy of Sciences of the United States of America*, vol.103, no.46, pp.17164-17169, 2006.