Parallel performance of an IB-LBM suspension simulation framework

Mountrakis, L.; Lorenz, E.; Malaspinas, O.; Alowayyed, S.; Chopard, B.; Hoekstra, A.G.

DOI
10.1016/j.jocs.2015.04.006
Publication date
2015
Document Version
Final published version
Published in
Journal of Computational Science

Journal of Computational Science 9 (2015) 45–50

Contents lists available at ScienceDirect

Journal of Computational Science

journal homepage: www.elsevier.com/locate/jocs

Parallel performance of an IB-LBM suspension simulation framework

Lampros Mountrakis[a,*], Eric Lorenz[a,b], Orestis Malaspinas[c,d], Saad Alowayyed[a,e], Bastien Chopard[c], Alfons G. Hoekstra[a,f]

[a] Computational Science, University of Amsterdam, Science Park 904, 1098 XH Amsterdam, The Netherlands
[b] Electric Ant Lab B.V., Panamalaan 4K, 1019 AZ, Amsterdam, The Netherlands
[c] Computer Science Department, University of Geneva, 7 route de Drize, 1227 Carouge, Switzerland
[d] Institut d'Alembert, UMR CNRS 7190, Université Pierre et Marie Curie – Paris 6, 4, place Jussieu, case 162, F-75252 Paris cedex 5, France
[e] King Abdulaziz City for Science and Technology (KACST), Riyadh, Saudi Arabia
[f] National Research University ITMO, Saint-Petersburg, Russian Federation

ARTICLE INFO

Article history:
Available online 17 April 2015

Keywords:
LBM
IBM
Parallel
MPI
Suspension
Blood
Palabos

ABSTRACT

We present performance results from `ficsion`, a general purpose parallel suspension solver, employing the Immersed-Boundary lattice-Boltzmann method (IB-LBM). `ficsion` is built on top of the open-source LBM framework `Palabos`, making use of its data structures and their inherent parallelism. We describe in brief the implementation and present weak and strong scaling results for simulations of dense red blood cell suspensions. Despite its complexity the simulations demonstrate a fairly good, close to linear scaling, both in the weak and strong scaling scenarios.

1. Introduction

Blood is a substance where the microstructure plays an important role in understanding the rheology and transport properties of this dense suspension. Approximately 5 million deformable red blood cells (RBCs) per cubic millimeter account for 40–45% of the total blood volume. They bring out many biologically interesting phenomena like the *Fåhræus* and *Fåhræus-Lindqvist* effects [1,2], the margination of platelets [3] and the non-Newtonian nature of blood [4]. Blood flow models can give insights to studies of cell diseases, such as malaria or sickle-cell anemia [5,6] and also in cardiovascular diseases like the formation of atherosclerotic plaques or thrombosis in aneurysms [7–10].

Simulations of dense suspensions, like blood, demand considerable computational resources. Software, in terms of algorithms, data structures and parallelism, is becoming more and more crucial in extracting knowledge from such systems. Implementing basic algorithms is relatively straightforward, yet complexity steeply increases by incorporating parallelism, elaborate boundary conditions, or other advanced elements, such as thermal and multiphase flows, moving objects, and suspended particles. A number of lattice-Boltzmann solvers already have several of the aforementioned capabilities implemented and tested and are released under an open-source license. Some examples of such established frameworks are, e.g. `Palabos` [11], `LB3D` [12], `Sailfish` [13], `HemeLB` [14], `LUDWIG` [15] and `Musubi` [16].

In this work we present the parallel performance of `ficsion`, a general purpose parallel IB-LBM solver with a focus on suspensions of deformable particles, like blood. `ficsion` is build on top of the open-source `C++` framework `Palabos` [11], making use of its data structures, parallelism and well-tested modules. The development of a fully parallelized suspension code implemented on top of a third-party framework is a challenging task, especially when the developer has no direct control over parallelization, and the existing parallel data structures have to be employed in a creative way. In this paper we briefly describe the implementation behind `ficsion` and present weak and strong scaling results for its application to the simulation of fully resolved blood flow.

2. Methods

Our approach is based on the immersed boundary-lattice Boltzmann method (IB-LBM), a combination frequently used in modeling blood suspensions [17–19]. Suspensions of deformable cells are the

* Corresponding author.
E-mail addresses: L.Mountrakis@uva.nl (L. Mountrakis), E.Lorenz@electricant.com (E. Lorenz), Orestis.Malaspinas@unige.ch (O. Malaspinas), S.A.Alowayyed@uva.nl (S. Alowayyed), Bastien.Chopard@unige.ch (B. Chopard), A.G.Hoekstra@uva.nl (A.G. Hoekstra).

http://dx.doi.org/10.1016/j.jocs.2015.04.006
1877-7503/

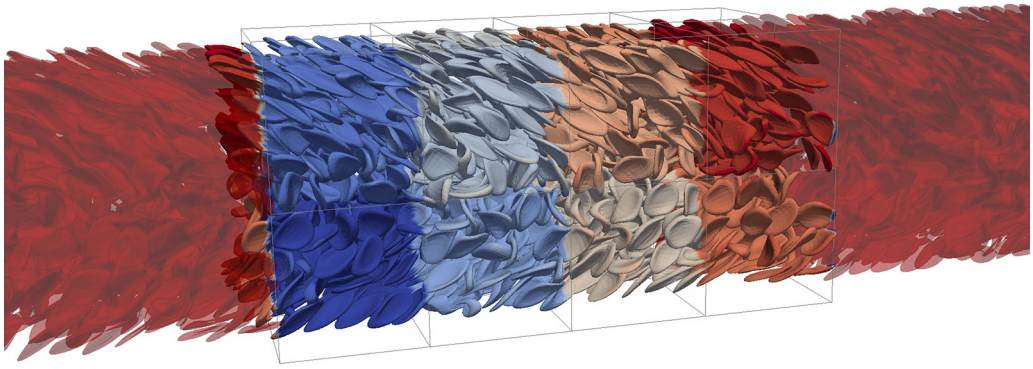

Fig. 1. Snapshot of channel flow between two parallel plates, placed on the top and bottom of the channel. The domain is periodic in the other dimensions. Size of the domain is $128 \times 64 \times 64\,\mu m^2$ with ≈1530 RBCs (hematocrit $H \approx 30\,\%$). The simulation was performed in 16 cores and color denotes the MPI rank. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

focus of `ficsion`, yet methods for hard-objects, like the Noble-Torczynski [20] or Ladd's method [21], could be incorporated and parallelism would be retained.

RBCs are represented as a mesh of Langrangian surface points, interacting via a spectrin link model. LBM is used for the fluid phase and is coupled to the suspended RBCs with the immersed boundary method. Fig. 1 depicts a snapshot from a representative simulation of blood flow between two parallel plates with `ficsion` using $N_p = 16$ processors. Periodic boundary conditions were used for this simulation, with a body force acting as a pressure gradient.

2.1. Lattice Boltzmann method

The lattice Boltzmann methods (LBMs) [22] are a class of mesoscopic particle-based approaches simulating fluid-flow. The local kinetic scheme of LBM allows for an intrinsic parallelization, rendering it a good candidate for parallel computing [23,24].

The main quantity of a lattice Boltzmann model is the population density $f_i(\mathbf{x}, t)$, which corresponds to the discretized probability distribution of finding fluid particles at site $\mathbf{x}$ and time t, moving with a discrete velocity $\mathbf{c}_i$. The general form of a LBM is:

$$f_i(\mathbf{x} + \mathbf{c}_i \Delta t, t + \Delta t) = f_i(\mathbf{x}, t) + \Omega_i(f_i(\mathbf{x}, t)) \tag{1}$$

where $\Omega_i(f_i(\mathbf{x}, t))$ is the collision operator, which re-shuffles densities f_i according to the kinetic theory of gases and Δt is the time step.

A wide range of models for a variety of applications has been developed starting from the general description. A simple and widely used form of $\Omega_i(f_i(\mathbf{x}, t))$ is the linearized single relaxation time collision operator, or LBGK model. In LBGK $\Omega_i(f_i(\mathbf{x}, t)) = -\frac{1}{\tau}(f_i(\mathbf{x}, t) - f_i^{\mathrm{eq}}(\mathbf{x}, t))$, where τ is the relaxation parameter and $f_i^{\mathrm{eq}}(\mathbf{x}, t)$ the equilibrium population corresponding to an expansion of the Maxwell–Boltzmann distribution for small Mach numbers. The zeroth and first moment of the population densities recover the fluid density ρ and velocity $\mathbf{u}$ according to $\rho = \sum_i f_i$ and $\rho\mathbf{u} = \sum_i f_i \mathbf{c}_i$. Kinematic viscosity ν for the LBGK collision operator is given by $\nu = (\tau - \frac{1}{2})c_s^2 \Delta t$ in which $c_s = \frac{1}{\sqrt{3}} \Delta x / \Delta t$ is the lattice speed of sound.

2.2. Immersed boundary method

The immersed boundary method (IBM) [25] is a pure coupling method used in fluid structure interaction. One of the main advantages of IBM, is that the fluid and the immersed structure do not need to conform. This alleviates the need for remeshing and renders complex configurations like dense suspensions easier to handle. With IBM the *cell*[1] follows the Lagrangian description.

The basic concept of IBM is the no-slip condition at the interface of the membrane and the fluid. This is realized as the Langrangian *surface points* (or *surface particles*) which constitute the membrane mesh, exert a force to the fluid, while they are advected by interpolating the fluid velocity.

The surface particle $\mathbf{x}_i(t)$ spreads a force $\mathbf{F}_i(t)$ to the closest Eulerian points $\mathbf{X}$ of the fluid according to

$$\mathbf{f}(\mathbf{X}, t) = \sum_i \mathbf{F}_i(t) \delta(\mathbf{X} - \mathbf{x}_i(t)) \tag{2}$$

where $\delta(\mathbf{X} - \mathbf{x}_i(t))$ is a discrete Dirac delta function. Following this step, the position of the particle is updated according to the Eulerian scheme

$$\mathbf{x}_i(t + \Delta t) = \mathbf{x}_i(t) + \mathbf{u}_i(t + \Delta t) \Delta t \tag{3}$$

where

$$\mathbf{u}_i(t + \Delta t) = \sum_{\mathbf{X}} \mathbf{u}(\mathbf{X}, t + \Delta t) \delta(\mathbf{X} - \mathbf{x}_i(t)), \tag{4}$$

and uses the same discrete Dirac delta function $\delta(\mathbf{X} - \mathbf{x}_i(t))$ as in Eq. (2).

$\delta(\mathbf{r})$ is constructed by multiplying 1D interpolation kernel functions ϕ, as $\delta(\mathbf{r}) = \phi(x)\phi(y)\phi(z)$. It is possible to find many discretized delta functions with varying interpolation ranges, yet a number of restrictions should be fulfilled [25]. In the present work we used the kernel $\phi_2(r)$:

$$\phi_2(r) = \begin{cases} 1 - |r| & |r| \le 1, \\ 0 & |r| \ge 1 \end{cases} \tag{5}$$

for its simplicity and compact support. The subscript of ϕ describes the support width of the kernel. More kernels are available and we have performed some benchmarks in 2D [19].

2.3. Membrane model of a single RBC

Blood consists of a vast number of red blood cells, biconcave disk-shaped membranes with a diameter of 6–8 μm and a thickness of approximately 2–2.5 μm. In the current work we employ the spectrin-link model to represent an RBC [26–29], in which a systematic coarse-graining procedure has been developed [27]

[1] The term *particle* is typically used in the literature of hard objects for what we here define as a *cell*. As *cell* we define the network of *surface particles* (or simply *particles*) that constitute the surface of the object and interact via a constitutive model.

allowing for an RBC to be described with arbitrary number of vertices.

The Helmholtz free-energy of a single RBC is given by

$$F(\{\mathbf{x}_n\}) = F_{\text{in-plane}} + F_{\text{bending}} + F_{\text{volume}} + F_{\text{area}} \tag{6}$$

The in-plane free energy term is written as

$$F_{\text{in-plane}} = \sum_{l \in \text{edges}} \frac{k_B T L_{\max}}{4p} \frac{3x_l^2 - 2x_l^3}{1 - x_l} + \sum_{l \in \text{edges}} \frac{k_{\text{rep}}}{L_l}. \tag{7}$$

The first term of Eq. (7) stands for the worm-like chain potential [26]. L_l, $L_{\max}$ and p, denote the length of edge l, the maximum allowed extension length and the persistence length respectively, k_{rep} is a constant chosen so that the in-plane force is zero for the equilibrium length L_0, x_l is defined as $x_l = L_l/L_{\max}$, k_B is Boltzmann's constant and $T = 300^o K$ the temperature.

The bending energy is defined as

$$F_{\text{bending}} = \sum_{\text{adjacent}\alpha,\beta\text{pair}} k_{\text{bend}}[1 - \cos\{\theta_{\alpha\beta} - \theta_0\}] \tag{8}$$

where k_{bend} is the bending constant, $\theta_{\alpha\beta}$ and θ_0 are the instantaneous and equilibrium angles between two adjacent triangles respectively. Volume and surface constraints read

$$F_{\text{volume}} = k_{\text{volume}} \frac{k_B T}{2L_0^3} \frac{(\Omega - \Omega_0)^2}{\Omega_0} \tag{9}$$

$$F_{\text{area}} = k_{\text{surface}} \frac{k_B T}{2L_0^2} \frac{(S - S_0)^2}{S_0} + \sum_{k \in 1 \ldots N_t} k_{\text{shear}} \frac{k_B T}{2L_0^2} \frac{(A_k - A_0)^2}{A_0} \tag{10}$$

in which k_{volume}, k_{surface} and k_{shear} are the volume, surface and local triangle area constants, while k_{volume} and k_{surface} can be chosen arbitrarily high, to ensure the conservation of volume and surface. Ω, S and A_k are the volume, surface and triangle area of the cell, while Ω_0, S_0 and A_0 are their corresponding equilibrium values. The values of the simulation parameters, including those of the fluid, are shown in Table 1. The force acting on vertex i is derived as $\mathbf{f}_i = -\frac{\partial F(\{\mathbf{x}_i\})}{\partial \mathbf{x}_i}$ and is spread to the fluid with IBM via Eq. (2).

2.4. Parallel implementation

`ficsion` is built on top of `Palabos` [11], one of the parallel open-source CFD solvers based on the lattice Boltzmann method. `Palabos` offers a wide range of modules and data-structures, in which the parallelization schemes are fully hidden in the API.

One of the main `Palabos` modules employed in `ficsion` is the class `ParticleField`: a container of Lagrangian particles in a structure analogous to the so-called *cell-list*[2] in Molecular Dynamics algorithms [31]. Finding neighboring particles for cell–cell forces can be a time consuming task and cell-lists can reduce the potentially $O(N^2)$ complexity to $O(N)$, which is crucial when the number of particles N becomes large [24,31]. The parallelism of a `ParticleField` follows the domain-decomposition of the fluid field and allows to directly identify the particles that belong to the *envelopes* and communicate them to the neighboring subdomains. Envelopes are the boundary nodes communicated in the domain-decomposition approach, also commonly referred to as ghost nodes.

Two instances of `ParticleField` are utilized for the simulation of a cell-type: one where particles represent the vertices of the cell's surface and interact according to the constitutive model – they are coined `SurfaceParticles` – and one for the cell as a whole carrying essential cell information like volume or surface – coined `CellParticles`. Since `ParticleField`s are fully parallelized, the field of `CellParticles` is taking care of all the necessary communication of information between the neighboring subdomains. The position of `CellParticles` is defined as the centroid of the `SurfaceParticles` belonging to the actual subdomain (envelopes excluded) and have only a small memory and communication footprint. The drawback of this approach is that it introduces an additional overhead of frequently instantiating and organizing `SurfaceParticles` into `CellParticles`, due to the motion of cells across the subdomains.

Optimizing communication is an important aspect of the implementation, since the different fields (fluid, `SurfaceParticles`, and `CellParticles`) have different needs for spatial information over the neighboring subdomains. In that respect, the widths of the envelopes differ for each field, to minimize the amount of data transferred. For the fluid this width is determined by the support width of the IBM kernel ϕ, for the `SurfaceParticles` by the maximum distance between two neighboring `SurfaceParticles` of a single cell[3] and for the `CellParticles` by the maximum stretch a cell can have. Typically for blood suspensions, `CellParticles` have larger envelopes than `SurfaceParticles`, which in their turn have larger envelopes than the fluid.

Table 1
Model parameter and constants. Most of the values were taken from modeling studies using a similar model, while others like volume and surface constants resulted from simulations of stretching RBC (data not shown).

Parameter	Value
Plasma density, ρ	1025 kg/m^3
Dynamic viscosity, ν	1.7×10^{-6} m^2/s [30]
Shear modulus, μ_0	5.5 μN/m [28]
Bending constant, k_{bend}	$100 k_B T$ [28]
Volume constant, k_{volume}	6×10^4
Surface constant, k_{surface}	6×10^4
Local area constant, k_{shear}	100 [28]
Maximum length, $L_{\max}$	$2.5 \times L_0$
Viscosity ratio, λ	1.0
Lattice constant, dx	1 μm
Timestep, dt	9.8×10^{-8} s
Relaxation time, τ	1.0
Number of vertices per cell, N_v	258

3. Results

In this section we report on the performance of `ficsion`. We perform an analysis of weak and strong scaling and examine the behavior with respect to hematocrit H, i.e. the cell number density. We are in the process of obtaining more detailed profiling measurements and additional parallel performance results.

The results of this study were obtained on Cartesius, SURFsara in The Netherlands [32], while the code has also been ported on FERMI BG/Q, CINECA in Italy [33]. The part of Cartesius we used consists of 3 thin node islands with 360 thin nodes each, with 2×12-core 2.6 GHz Intel Xeon E5-2690 v3 (Haswell) CPUs/node with hyper-threading technology. Each node has 64 GB/node and a Mellanox ConnectX-3 InfiniBand adapter.

3.1. Weak and strong scaling

A fully periodic unbounded domain was used for the simulations and an initial velocity of (0.02, 0.02, 0.02) lattice units was applied to the fluid. Essentially, this setup is a Galilean invariant case, forcing cells to move only with respect to the reference frame. For the weak-scaling case, cells were initialized randomly at a fraction of

[2] Not to be confused with the definition of a *cell* used here.
[3] Owing to the bending force in which two neighboring triangles must be present for the computation, a safe choice would be two time the maximum distance.

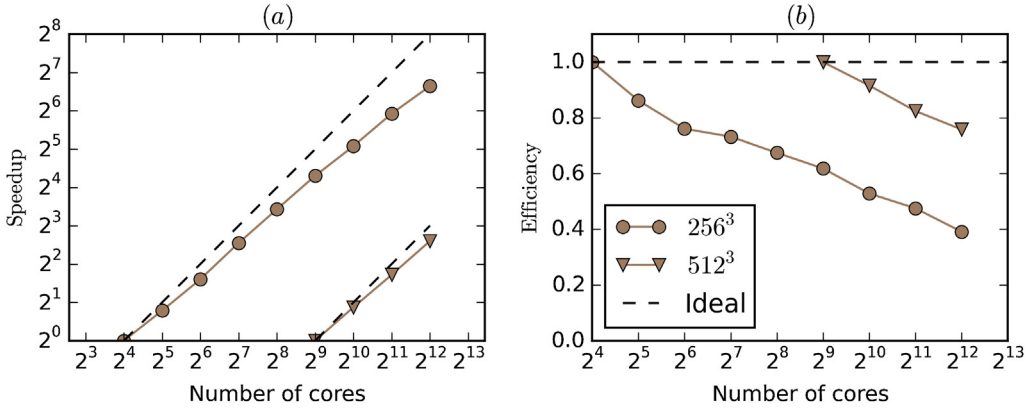

Fig. 2. Strong scaling (a) Speedup and (b) Efficiency for domains 256^3 and $512^3\,\mu m^3$ containing 83,700 and 662,400 RBCs respectively for a hematocrit of 45%. N_0 is 16 and 512 respectively. In the 256^3 case, the subdomain size is $128 \times 128 \times 64$ for $N_p = 2^4$ and $16 \times 16 \times 16$ for $N_p = 2^{12}$. For 512^3, the subdomain size is $64 \times 64 \times 64$ for $N_p = 2^4$ and $32 \times 32 \times 32$ for $N_p = 2^{12}$.

their initial volume, while for the strong-scaling case, cells were initialized with their full volume in ordered columns. Load balancing issues and specifically deviations in the number of cells, were one of the main reasons for the poor scaling performance of Clausen et al. [34] on the IBM Blue Gene/P architecture. Our initialization setup ensures a high degree of load-balance, expecting the same amount of `SurfaceParticles` and `CellParticles` in each subdomain.

The quantity we measure is the wallclock-time per iteration taken from the last 100 iterations of a 200 iteration simulation and one iteration is one LBM timestep. In these 200 iterations all cells moved 4 lattice units from their initial position. Initialization and I/O operations were not taken into account. The parameters of the simulations are shown in Table 1.

For strong scaling the size of the whole domain is fixed and the number of processes varies. Strong scaling Speedup and Efficiency read:

$$\text{Speedup}_{\text{strong}} = \frac{t_{N_0}}{t_N} \tag{11}$$

$$\text{Efficiency}_{\text{strong}} = \frac{t_{N_0}}{t_N}\frac{N_0}{N} \times 100\%. \tag{12}$$

with t_{N_0} being the time spend in N_0 processes and t_N in N processes.

In weak scaling the size of the subdomain per process is fixed, and parallel Speedup and Efficiency are defined as:

$$\text{Speedup}_{\text{weak}} = \frac{t_{N_0}}{t_N}\frac{N}{N_0} \tag{13}$$

$$\text{Efficiency}_{\text{weak}} = \frac{t_{N_0}}{t_N} \times 100\%. \tag{14}$$

Figs. 2a and 3b reveal a fairly good strong and weak scaling. This can be attributed to two points: (a) the computation to communication ratio is in favor of computation due to the low spatial resolution, and (b) the cases we ran are almost perfectly load balanced, with insignificant deviations in the number of cells and without deviations in the number of LBM nodes. With the low resolution employed, when compared for example to Clausen et al. [34], more RBCs can fit per cubic lattice unit increasing the amount of computation per subdomain. The influence of the load imbalance, for example created by the presence of solid walls inducing cell free layers close to the surface, is part of our future work.

In weak scaling, a unit-subdomain of $32 \times 32 \times 32$ is used for each processor. For $N_p = 16$ a grid of $4 \times 2 \times 2$ subdomains is created, while $N_p = 32$ yields a $4 \times 4 \times 2$. The first case where a central subdomain exists, i.e. it has different neighboring subdomains on all sides all residing on different processors, is observed for $N_p = 32$ and in combination with the fact that each thin node in Cartesius has 24 cores so that all subdomains run in a shared-memory space, it could explain the drop from 16 to 32 cores in Fig. 3.

3.2. Preliminary profiling results

In Fig. 4a we observe that the constitutive model (with the identifier `CellModel`) consumes most of the computational time, followed by `IBM` and routines related to book-keeping and data

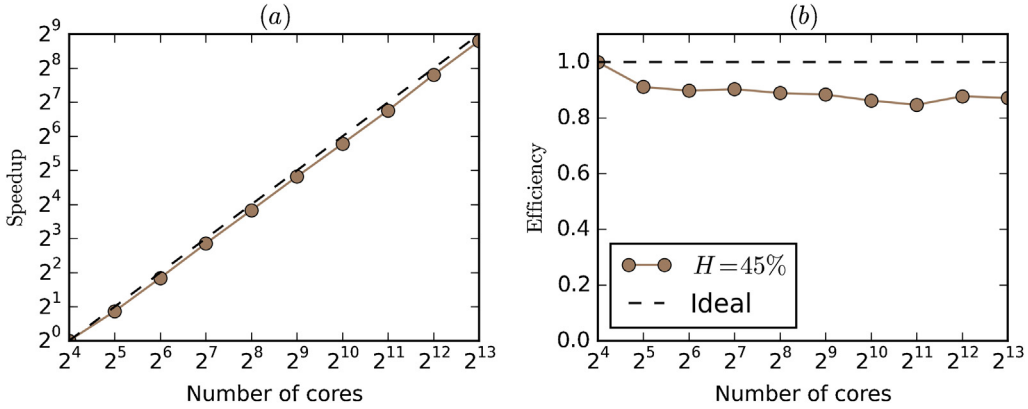

Fig. 3. Weak scaling (a) Speedup and (b) Efficiency for a subdomain of $32^3\,\mu m^3$ with ≈41 RBCs per core and $N_0 = 16$. $N_p = 2^5$ is the first case with a central subdomain, neighboring on all sides and this could explain the drop in performance.

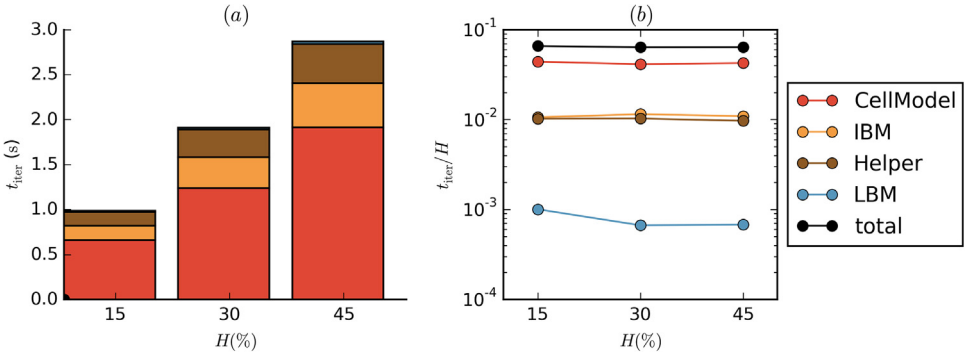

Fig. 4. (a) Time per iteration, per module with respect to hematocrit H. (b) Time per iteration per hematocrit. Hematocrits 15%, 30% and 45% correspond to 3417, 6839 and 10,270 RBCs respectively inside a domain of $128^3\,\mu m^3$. The simulations were performed on 64 cores and timings for modules include communication and synchronization steps.

structures (identifier `Helper`). `LBM` takes up only a small fraction of the computational time. This is a trait of the low resolution ($dx = 1\,\mu m$), resulting in more RBCs per cubic lattice unit, and consequently per subdomain, which as mentioned earlier, leads to a **favourable** computation to communication ratio.

Fig. 4b shows that all modules scale linearly with hematocrit, except `LBM`, which is expected to remain constant and independent of hematocrit, given a small increase due to the increased number of nodes performing the LBM force-collision step.

4. Discussion and conclusions

We have presented the performance of `ficsion`, a suspension framework build on top of the open-source LBM solver `Palabos`. We have shown that `ficsion` scales linearly with the hematocrit and exhibits satisfactory weak and strong scaling results, as a consequence of the computation to communication ratio and the **favourable** load-balance of the simulation setup.

Perfectly load-balanced cases are not representative for suspensions [24]. In blood for example, it is known that RBC distributions are inhomogeneous with a red blood cell-free layer near the walls of the vessel [35,36]. For a static regular decomposition based on the lattice, this would result in load-imbalance, since some subdomains would have less or no RBCs at all. We will study the effect of such vessel wall-induced imbalance in follow-up studies.

Acknowledgements

This research receives funding from the European Union, 7th Framework Programme under grant agreement n° 26996 (http://www.thrombus-vph.eu). We acknowledge PRACE for awarding us access to CINECA, Italy and the Dutch Science Foundation for awarding us access to Cartesius, The Netherlands. Author AGH acknowledges partial financial support by the Russian Scientific Foundation, grant # 14-11-00826. Author SA acknowledges funding by King Abdulaziz City for Science and Technology (KACST), Saudi Arabia.

References

[1] R. Fåhraeus, The suspension stability of the blood, Physiol. Rev. 9 (2) (1929) 241–274.
[2] R. Fåhræus, T. Lindqvist, The viscosity of the blood in narrow capillary tubes, Am. J. Physiol. – Legacy Cont. 96 (3) (1931 March) 562–568.
[3] P.A. Aarts, S.A. van den Broek, G.W. Prins, G.D. Kuiken, J.J. Sixma, R.M. Heethaar, Blood platelets are concentrated near the wall and red blood cells, in the center in flowing blood, Arterioscler. Thromb. Vasc. Biol. 8 (6) (1988 November) 819–824.
[4] Y.C. Fung, Biomechanics: Circulation, Springer, softcover reprint of hardcover 2nd ed. 1997 ed., December 2010.
[5] T. Ye, N. Phan-Thien, B.C. Khoo, C.T. Lim, Stretching and relaxation of malaria-infected red blood cells, Biophys. J. 105 (5) (2013 September) 1103–1109.
[6] X. Li, P.M. Vlahovska, G.E. Karniadakis, Continuum- and particle-based modeling of shapes and dynamics of red blood cells in health and disease, Soft Matter 9 (1) (2013) 28–37.
[7] B. Chopard, R. Ouared, D.A. Ruefenacht, H. Yilmax, Lattice Boltzmann modeling of thrombosis in giant aneurysms, Int. J. Modern Phys. C 18 (4) (2007) 712.
[8] S. Zimny, B. Chopard, O. Malaspinas, E. Lorenz, K. Jain, S. Roller, J. Bernsdorf, A multiscale approach for the coupled simulation of blood flow and thrombus formation in intracranial aneurysms, Procedia Comput. Sci. 18 (2013 January) 1006–1015.
[9] L. Mountrakis, E. Lorenz, A.G. Hoekstra, Where do the platelets go? A simulation study of fully resolved blood flow through aneurysmal vessels, Interface Focus 3 (2) (April 2013).
[10] S. Melchionna, G. Amati, M. Bernaschi, M. Bisson, S. Succi, D. Mitsouras, F.J. Rybicki, Risk assessment of atherosclerotic plaques based on global biomechanics, Med. Eng. Phys. 35 (9) (2013 September) 1290–1297.
[11] http://www.palabos.org
[12] http://mtp.phys.tue.nl/lb3d/
[13] M. Januszewski, M. Kostur, Sailfish: a flexible multi-GPU implementation of the lattice Boltzmann method, Comput. Phys. Commun. 185 (9) (2014 September) 2350–2368.
[14] M.D. Mazzeo, P.V. Coveney, HemeLB: a high performance parallel lattice-Boltzmann code for large scale fluid flow in complex geometries, Comput. Phys. Commun. 178 (12) (2008 June) 894–914.
[15] J.-C. Desplat, I. Pagonabarraga, P. Bladon, LUDWIG: a parallel lattice-Boltzmann code for complex fluids, Comput. Phys. Commun. 134 (3) (2001 March) 273–290.
[16] M. Hasert, K. Masilamani, S Zimny, H Klimach, J Qi, J. Bernsdorf, S. Roller, Complex fluid simulations with the parallel tree-based lattice Boltzmann solver musubi, J. Comput. Sci. 5 (5) (2014 September) 784–794.
[17] T. Kruger, M. Gross, D. Raabe, F. Varnik, Crossover from tumbling to tank-treading-like motion in dense simulated suspensions of red blood cells, Soft Matter 9 (37) (2013) 9008–9015.
[18] J. Zhang, P.C. Johnson, A.S. Popel, An immersed boundary lattice Boltzmann approach to simulate deformable liquid capsules and its application to microscopic blood flows, Phys. Biol. 4 (4) (2007 December) 285.
[19] L. Mountrakis, E. Lorenz, A.G. Hoekstra, Validation of an efficient two-dimensional model for dense suspensions of red blood cells, Int. J. Mod. Phys. C (2014 March) 1441005.
[20] D.R. Noble, J.R. Torczynski, A lattice-Boltzmann method for partially saturated computational cells, Int. J. Mod. Phys. C 9 (8) (1998 December) 1189–1201.
[21] A.J.C. Ladd, Numerical simulations of particulate suspensions via a discretized Boltzmann equation part I. Theoretical foundation, J. Fluid Mech. 271 (1993 July) 285–309.
[22] S. Succi, The Lattice Boltzmann Equation for Fluid Dynamics and Beyond (Numerical Mathematics and Scientific Computation), Oxford University Press, USA, 2001 August.
[23] L. Axner, J. Bernsdorf, T. Zeiser, P. Lammers, J. Linxweiler, A.G. Hoekstra, Performance evaluation of a parallel sparse lattice Boltzmann solver, J. Comput. Phys. 227 (10) (2008 May) 4895–4911.
[24] K. Stratford, I. Pagonabarraga, Parallel simulation of particle suspensions with the lattice Boltzmann method, Comput. Math. Appl. 55 (7) (2008 April) 1585–1593.
[25] C.S. Peskin, The immersed boundary method, Acta Numer. 11 (1) (2002) 479–517.
[26] J. Li, M. Dao, C.T. Lim, S. Suresh, Spectrin-level modeling of the cytoskeleton and optical tweezers stretching of the erythrocyte, Biophys. J. 88 (5) (2005 May) 3707–3719.

[27] I.V. Pivkin, G.E. Karniadakis, Accurate coarse-grained modeling of red blood cells, Phys. Rev. Lett. 101 (11) (2008 September) 118105.
[28] D.A. Fedosov, B. Caswell, G.E. Karniadakis, Systematic coarse-graining of spectrin-level red blood cell models, Comput. Methods Appl. Mech. Eng. 199 (29–32) (2010 June) 1937–1948.
[29] D.A. Reasor, J.R. Clausen, C.K. Aidun, Coupling the lattice-Boltzmann and spectrin-link methods for the direct numerical simulation of cellular blood flow, Int. J. Numer. Methods Fluids 68 (6) (2012 February) 767–781.
[30] M.V. Kameneva, M.J. Watach, H.S. Borovetz, Gender difference in rheologic properties of blood and risk of cardiovascular diseases, Clin. Hemorheol. Microcirc. 21 (3) (1999 January) 357–363.
[31] P. Allen, D.J. Tildesley, Computer Simulation of Liquids, Oxford Science Publications, Clarendon Press, 1987.
[32] Cartesius@SURFsara. Retrieved from: http://surfsara.nl/systems/cartesius (accessed 08.01.15).
[33] Fermi@cineca. Retrieved from: http://www.cineca.it/en/content/fermi-bgq (accessed 08.01.15).
[34] J.R. Clausen, D.A. Reasor, C.K. Aidun, Parallel performance of a lattice-Boltzmann/finite element cellular blood flow solver on the IBM blue Gene/P architecture, Comput. Phys. Commun. 181 (6) (2010 June) 1013–1020.
[35] D.A. Fedosov, B. Caswell, A.S. Popel, G.E. Karniadakis, Blood flow and cell-free layer in microvessels, Microcirculation 17 (8) (2010) 615–628.
[36] V. Narsimhan, H. Zhao, E.S.G. Shaqfeh, Coarse-grained theory to predict the concentration distribution of red blood cells in wall-bounded Couette flow at zero Reynolds number, Phys. Fluids 25 (6) (2013 June) 061901.