The In-Silico Lab-on-a-Chip: Petascale and High-Throughput Simulations of Microfluidics at Cell Resolution

Diego Rossinelli[1], Yu-Hang Tang[2,*], Kirill Lykov[3,*], Dmitry Alexeev[1],
Massimo Bernaschi[4], Panagiotis Hadjidoukas[1], Mauro Bisson[5], Wayne Joubert[6], Christian Conti[1],
George Karniadakis[2], Massimiliano Fatica[5], Igor Pivkin[3], Petros Koumoutsakos[1]
[1]ETH Zürich, Switzerland, [2]Brown University, U.S.A., [3]USI, Switzerland,
[4]CNR, Italy, [5]NVIDIA Corporation [6]Oak Ridge National Laboratory (ORNL), U.S.A.
[*]Equally credited authors

ABSTRACT

We present simulations of blood and cancer cell separation in complex microfluidic channels with subcellular resolution, demonstrating unprecedented time to solution, performing at 65.5% of the available 39.4 PetaInstructions/s in the 18, 688 nodes of the Titan supercomputer.

These simulations outperform by one to three orders of magnitude the current state of the art in terms of numbers of simulated cells and computational elements. The computational setup emulates the conditions and the geometric complexity of microfluidic experiments and our results reproduce the experimental findings. These simulations provide sub-micron resolution while accessing time scales relevant to engineering designs.

We demonstrate an improvement of up to 45X over competing state-of-the-art solvers, thus establishing the frontiers of simulations by particle based methods. Our simulations redefine the role of computational science for the development of microfluidics – a technology that is becoming as important to medicine as integrated circuits have been to computers.

Keywords

Supercomputing, time-to-solution, peak performance, irregular codes, dissipative particle dynamics, microfluidics simulations, blood analytics, circulating tumor cells

1. INTRODUCTION

The quest towards miniaturisation of devices capable of handling fluids and biological matter at a molecular level is revolutionising medicine and pharmaceutical research. Microfluidic systems, such as *lab-on-a-chip*, transport and manipulate nanoliter quantities of fluids and cells to enable precise analysis and large-scale automation for important biochemical processes. By benefiting from an economy of scales, microfluidics is advancing technologies ranging from clinical diagnostics and cancer detection to regenerative medicine and proteomics [67, 61].

Microfluidic devices exploit the functional properties of fluids confined in microscopic channels to perform tasks such as blood separation, detection of circulating tumor cells, and molecular recognition with high sensitivity and specificity [59, 10]. However, the acquisition of information of the fluid micro-rheology, including phenomena such as cell merging, lysis [45] and biochemical signaling [69] remains a formidable task [52]. Without such information, the unique, and often unexpected, properties of fluids at the micro-scale can be a hindrance for new designs while complex prototypes may be impossible to manufacture at an economy of scales. Fast turnaround times for design are essential for the evolving technologies (3D printing, liquid biopsies, organ-on-a-chip) of microfluidic chips that can assist medical analysis and drug development [57, 43, 12]. Predictive simulations are becoming an invaluable tool in accelerating the design cycle for microfluidics [33].

Today, the majority of simulations for microfluidic systems employs continuum models. Grid based and Boundary Integral solvers of the linear viscous Stokes equations can handle the geometries of the microfluidic channels but they have limited capabilities in resolving essential sub-micron flow biophysics [18]. Continuum models cannot be readily extended to capture cell topological changes, membrane transport and electrochemical interactions between the channel and the fluid constituents. Such processes are essential for important microfluidic applications ranging from drug delivery and immunoassays [10] to micro-robotics for non-invasive surgery [46].

Mesoscale models, such as Lattice Boltzmann (LB) [9] and Stochastic Particle Methods (SPM) ([20, 34] and references therein), use discrete particles to represent flows in microfluidic channels. They overcome the limiting assumptions of continuum models at the expense of requiring larger numbers of computational elements to resolve sub-micron scales

SC '15, November 15-20, 2015, Austin, TX, USA
© 2015 ACM. ISBN 978-1-4503-3723-6/15/11. . . $15.00
DOI: http://dx.doi.org/10.1145/2807591.2807677

and processes. LB methods have been ported to supercomputing architectures to simulate blood flows [6] and protein crowding phenomena [7]. The LB kernel is memory intensive and although very amenable to parallel processing, it hardly achieves a significant percentage of the peak performance on most modern computing platforms [8]. Similar to continuum models, LB approaches need the a priori specification of the diffusion constant. Moreover, their accuracy decreases rapidly and their computational cost increases as the Reynolds number approaches zero, thus limiting their applicability to microfluidics.

Dissipative Particle Dynamics (*DPD*) [27] is a SPM that bridges the gap between Molecular Dynamics and Navier Stokes equations [13]. It has been used extensively to model complex fluids such as colloidal suspensions, emulsions and polymers [26] and has become a key method for the study of the blood microrheology [55, 70]. DPD resolves cells at a sub-micron scale to simulate microfluidic [60] and drug delivery systems [44], reaching time scales that have been previously accessible only to Navier-Stokes solvers [18]. State of the art DPD solvers are based on extensions of Molecular Dynamics (MD) software packages such as LAMMPS [54] and HOOMD-Blue [3] targeting both CPU-only and GPU-accelerated supercomputers. The DPD is a consistent coarse graining of Molecular Dynamics (MD) [25], accessing continuum time scales with sub-micron resolution. However, while MD and DPD share similarities as particle based solvers, the physics represented by their interaction potentials have significant differences [34]. These differences translate into a number of challenges for the effective parallelisation of DPD. As an example, in DPD the overall exposed instruction-level parallelism (ILP) is higher than in MD and it can be extracted by the current CPU and GPU microarchitectures.

GPU-accelerated supercomputers exhibit significant differences from those based solely on CPUs in terms of throughput and latencies of the different instructions. On the CPU, integer instructions are usually the fastest, while double precision (DP) instructions are among the most expensive and single precision (SP) instructions have twice the throughput of the DP instructions. In contrast, on the GPUs of current supercomputers, SP instructions are the fastest. These SP instructions take just one cycle and the throughput imbalance between SP and DP instructions is 3:1, instead of the more common 2:1. Furthermore, on GPUs, many integer instructions are as expensive as the DP ones.

These imbalances indicate that GPUs in current supercomputer have been designed to process SP instructions more efficiently than any other instructions. This leads to rather counter-intuitive observations regarding the "traditional" performance metrics in supercomputing. For example, a compute kernel featuring exclusively DP instructions that is capable of achieving 100% of the nominal DP peak performance, is actually performing at 1/3 of the potential compute power offered by the GPU. This challenges metrics based on DP, since the performance achieved by such a DP kernel is arguably 33% of the nominal throughput. Hence, in this work we use the vector instruction per second [I/s] and instruction per cycle [I/c] (or IPC) as performance metrics.

We present a high-throughput software for DPD simulations based on thoroughly optimized kernels for GPUs. Our code simulates for the first time blood flow and circulating tumor cells in realistic geometries of complex microfluidic channels at sub-micron resolution and times that are relevant to clinical applications. The software maps effectively the DPD method to the 18,688 GPUs of the Titan supercomputer by overcoming the following challenges:

- The irregular computational patterns inherent to DPD are not readily data-parallel. These irregularities prevent the simulations from reaching high fractions of the peak performance.
- DPD involves particle interactions that require hundreds of instructions. This is an order of magnitude more instructions than Lennard-Jones interactions in MD simulations.
- DPD accesses time scales that are several orders of magnitude larger than those accessible by MD. At the same time, in DPD simulations, the particle neighborhood changes practically at each time step. Hence during the evaluation of the interactions around 80% of the instructions are spent in identifying possible neighbors.
- The DPD thermostat (dissipative and random forces) is the most expensive part in the computation of DPD interactions. It consists mostly of integer instructions which do not map efficiently on current GPUs.
- DPD implies challenging inter-node communication patterns. It requires non-trivial CPU-GPU cooperations to pack/unpack messages, and hundreds of MPI point-to-point communications per rank and time step. Message sizes are not known a-priori, preventing a straightforward Computation/Transfer (C/T) overlap.

The present DPD solver effectively exploits the capabilities of the Titan supercomputer and allows for simulations at spatiotemporal scales that are impossible for MD, while maintaining molecular level details that are inaccessible to continuum solvers. The goal of the present effort is to accelerate the design and testing of microfluidics devices by an order of magnitude while resolving phenomena at sub-micron scales. The target time-to-solution for simulations of a full microfluidic device is in the range of 2-6 hours, implying $10^6 - 10^7$ iterations. This goal is met by the present code.

Contributions of this paper.

The present work redefines the frontiers of simulations for microfluidic devices by improving by two orders of magnitude the geometrical detail as well as the number of cells previously resolved using continuum solvers. We present a tool which uses the most accurate and experimentally validated [16, 14], deformable models for *red blood cells (RBC), white blood cells (WBC) and circulating tumor cells (CTC)* and can perform, for the first time *in-silico*, cell separation in microfluidic devices that have been previously tested only *in-vitro*.

Our software outperforms state of the art DPD solvers, deployed on GPU based supercomputers, by *up to 45X*. The software enables unprecedented simulations of cell separation in two different microfluidic devices [32, 42] with sub-micron resolution. In addition to the unprecedented geometric complexity afforded by our code, these simulations provide a *three orders of magnitude improvement* over the current state of the art in terms of numbers of RBCs and time-to-solution.

The paper is structured as follows: in Section 2 we give a brief overview of the current state of the art for DPD software and simulations related to microfluidics and in Section 3 we introduce the mathematical framework for DPD. We describe the computing platforms we used for our simulations in Section 4 while in Section 5 and Section 6 we report details of the design and development of the present software. In Section 7 we present the results of our simulations. Finally, we present our performance results in Section 8 and provide conclusions in Section 9.

2. CURRENT STATE OF THE ART

DPD on supercomputers.

The computational complexity of DPD has hindered its use for large-scale simulations for time scales relevant to engineering designs. Accurate DPD models require at least 500 elements to represent the dynamics of a RBC [53], hence for simulating RBC separation in a blood sample of $1mm^3$ with a hematocrit of 45% over 10^9 particles are required. *To the best of our knowledge, there has never been a simulation of blood flow in complex 3D microfluidics chip with subcellular-level resolution, reaching spatiotemporal scales of realistic devices as shown in this study.*

Currently, software for DPD deployed on supercomputers is based on extensions of code originally developed for MD, such as LAMMPS [54] and HOOMD-Blue [3]. LAMMPS (Large-Scale Atomic/Molecular Massively Parallel Simulator) was originally developed for MD simulations on distributed memory architectures with CPUs featuring Single Instruction Multiple Data (SIMD) [54]. LAMMPS is one of the six projects selected from the Center for Accelerated Application Readiness by the Oak Ridge Leadership Computing Facility (OLCF) and has been extended to perform DPD simulations on Titan [47]. The largest LAMMPS/DPD simulation reported 32-256K particles per node. They simulated 256 Million particles on 1024 Titan nodes with a throughput of 5 - 7.6 Million Particles per second [MP/s][47]. We demonstrate that our implementation of DPD outperforms this Titan optimized version of LAMMPS by up to 45X.

HOOMD-Blue [3] aims at taking advantage of the compute capabilities offered by Single Instruction Multiple Thread (SIMT) architectures [19] and has also been extended to support DPD simulations [51]. The fastest HOOMD-Blue/DPD simulation reported throughput 1.85 [MP/s] on a GeForce GTX 480 [51], which, projected on a Titan node, would correspond to 6.7 [MP/s]. However, in its current version, HOOMD-Blue *is not capable of large-scale simulations using more than 124 Million particles* (see also the Performance Section).

The largest and fastest DPD simulations so far, had been performed with USER-MESO, a LAMMPS fork by Tang and Karniadakis [62]. These simulations employed 1 Billion particles on 1024 Titan nodes with reported throughput of $10 - 30$ [MP/s] per node, for the simulation of spontaneous vesicle formation.

Blood flow simulations.

The first large-scale simulations of RBCs were performed by Rahimian et al. [56]. They used a boundary integral solution of the continuum linear Stokes equation to simulate 262 Million RBCs, each discretized with 84 vertices, totaling 88 billion unknowns. In terms of throughput, they measured about 290 Million unknowns/s, leading to a theoretical throughput of about 2.6 Billion RBC unknowns/s on Titan.

Lattice Boltzmann simulations of the blood flow into a coronary arteries network employed 450 Million non-deformable RBCs observed a throughput of 540 Million unknowns/s and 1 PFLOP/s in double precision on Tsubame [6]. Xu et al. [68] performed blood flow simulations with 50,000 RBCs at the exceptional resolution of 3,300 vertices per RBC. Large-scale rheological simulations, albeit with non-deforming RBCs, have reached 4 Millions RBCs [31].

The first DPD simulations of RBCs transiting in small microfluidics channels [55] predicted the strong dependency between temperature and the dynamics of a RBC due to changes in the membrane viscosity. Methodological improvements and increased resolution enabled DPD to predict, for the first time, the blood viscosity as a function of the shear rate and hematocrit level [16]. These simulations involved about 200 RBCs and provided insight on the magnitude of the adhesive forces acting between cells. In the last few years the use of DPD has burgeoned for the study of cellular systems and in particular for the study blood flow in microfluidic devices. These studies include the first ever combination of experiments and DPD simulations to examine large deformations of RBCs in microfluidic devices [55] and the identification of the governing mechanisms for increased blood viscosity in sickle cell anemia [37].

CTC models and microfluidics devices.

The detection of CTCs in the blood is one of the most potent methods for the early diagnosis of cancer [2] and a key target of liquid biopsies [49]. However, with 1 CTC present for every 10^9 RBCs, this detection is equivalent to finding a "needle in a *flowing* haystack". This challenge has led to a wealth of activity in developing microfluidic devices for high throughput cell separation. At the same time, our understanding of flow mediated interactions between CTCs, blood and microfluidic devices is hindered by the lack of detailed experimental information [35]. The rational design of such devices based on simulations is at its infancy. The Volume of Fluid method was employed to model a CTC passing through a narrow channel [70] while a CTC membrane model based on the LB approximation investigated microfluidics devices exerting large cell deformations [24]. Other studies include flows of suspended RBCs and WBCs past an array of displaced circular obstacles [65], DPD simulations of the bead assay problem [60] and in-vitro and in-silico studies of a passive hydrodynamic cell sorter for leukocyte enrichment [11].

The CTC-iChip [32], arguably one of the most effective microfluidic device to date to separate CTCs from RBCs was designed with the assistance of a Finite Element solver of the continuum Stokes equations. The simulations were used to investigated the shape of a simplified array of obstacles that lead to cell separation by exploiting the concept of the Deterministic Lateral Displacement [28]. The resulting optimal "egg-looking" shape has been used as a building block in the first compartment of the CTC-iChip microfluidics device. This compartment, in its full geometric complexity, is one of the devices studied in the present work.

3. NUMERICAL METHODS

The method of Dissipative Particle Dynamics (DPD) [27, 22] employs discrete particles interacting through pairwise forces. The force $\mathbf{F}_i$ acting on particle i consists of three additive parts:

$$\mathbf{F}_i = \sum_{j \neq i}^{N} (\mathbf{F}_{ij}^C + \mathbf{F}_{ij}^D + \mathbf{F}_{ij}^R) \ , \ i = 1, ..., N \qquad (1)$$

which are non-zero within a cutoff radius ($R_c = 1$). The conservative force $\mathbf{F}_{ij}^C$ is a soft repulsion force, acting along the vector between particles i and j. The dissipative force $\mathbf{F}_{ij}^D$ depends on the relative velocity of the particles and models friction effects, while the random force $\mathbf{F}_{ij}^R$ represents the effect of the suppressed degrees of freedom in the form of thermal fluctuations. The parameters of the random and dissipative forces are related through a fluctuation-dissipation theorem [13]. The time evolution of this N-body problem is described by Newton's equations of motion and a leap-frog algorithm is used to advance the system in time.

We emphasize that, unlike most mesoscale particle methods, *the DPD is rigorously derived from MD through the Mori-Zwanzig coarse-graining procedure* [25]. DPD models explicitly the hydrodynamics forces in the solvent and entails an accurate kinematic description of the flow [13, 17]. The complex interaction between fluids, cells and walls, are modeled with appropriate DPD parameters. DPD is an inherently multiscale particle method, well suited to simulating microfluidics devices allowing for sub-micron resolution and accessing continuum time scales. The computational stages involved in one time step of a DPD flow simulation involving suspended RBCs and CTCs is shown in Figure 2.

Walls and microscale geometries.

The boundaries of the microfluidic channels are discretized by first equilibrating a homogeneous DPD fluid and then freezing particles which are inside the wall geometry. The channel wall geometry is represented implicitly by a Signed Distance Function (SDF) allowing for domains with very high geometric complexity. Boundary conditions are enforced through DPD interactions between wall and solvent particles and a bounce-back mechanism after the update of the particle positions [63].

Cell Modeling.

The RBC membrane has two main components: an elastic spectrin network and an incompressible lipid bilayer. When relaxed, the membrane assumes a biconcave shape with a diameter of $D = 7.6 \mu m$ (for humans). In this work, the RBC membrane is modeled as a mesoscopic viscoelastic membrane. The model adopted here was proposed by Li et al. [39] and later extended by viscous membrane forces [48]. This model is intensively used to study the mechanical properties of individual RBCs [55, 38, 50] as well as blood flows in various conditions[37, 16, 15].

The RBC membrane is discretized on a mesh as depicted in Figure 1. There are three types of bonded forces on each vertex of the mesh: pairwise interactions for all incident edges (black edge, F_{bond}), angle forces for all incident triangles (red triangle, $F_{triangle}$), dihedral forces for all pairs of adjacent incident triangles (orange triangles, $F_{dihedral1}$), and dihedral forces for pairs of adjacent triangles where one triangle is incident to the vertex (yellow triangles, $F_{dihedral2}$) [14]. The number of vertices per membrane should be chosen according to the required level of details. For the present numerical investigations we consider a discretization involving 500 vertices.

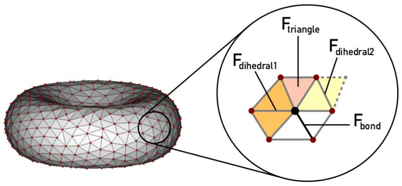

Figure 1: An RBC membrane model discretized by 500 vertices (left) and the bonded force types acting on it (right).

The modeling of WBCs and CTCs employs a recently proposed, experimentally validated, extension of a RBC model [15]. We remark that there is a wide variety of CTC types with disparate diameters ranging from 10 μm to 24 μm and with significantly different mechanical properties [23, 29, 40]. Here we consider MCF-7 breast cancer cells with a diameter of 14 μm by using 2484 vertices as well as RT4 bladder cancer cells with diameter 20 μm and 5220 vertices.

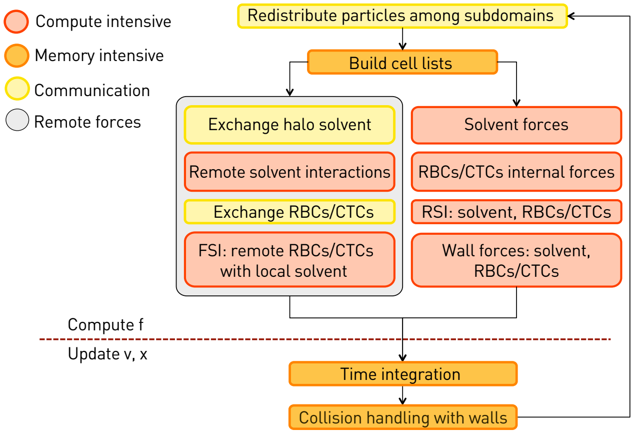
Compute intensive
Memory intensive
Communication
Remote forces
Redistribute particles among subdomains
Build cell lists
Exchange halo solvent
Solvent forces
Remote solvent interactions
RBCs/CTCs internal forces
Exchange RBCs/CTCs
RSI: solvent, RBCs/CTCs
FSI: remote RBCs/CTCs with local solvent
Wall forces: solvent, RBCs/CTCs
Compute f
Update v, x
Time integration
Collision handling with walls

Figure 2: The computational stages involved in one time step of a DPD simulation involving suspended RBCs and CTCs.

4. TARGET PLATFORMS

The present simulation software was developed targeting two supercomputers: *Titan*, the fastest supercomputer *available* at the present time, from Oak Ridge National Laboratory; and *Piz Daint*, the fastest supercomputer in Europe, from the Swiss National Supercomputing Centre (CSCS). These supercomputers have a nominal aggregate CPU-GPU peak rate of 39 and 12 normalized[1] PetaInstructions/s respectively (see Table 1).

The per-node performance for Titan and Piz Daint is primarily attributed to the NVIDIA K20X GPU, a SIMT architecture which dispatches the computation to its 14 Streaming Multiprocessors (SMX), each containing 192 CUDA cores.

[1] #nodes × (#CPU cores × CPU-freq × SIMD-width × 2 [I/c] + $0.732 \cdot 10^9$ [Hz] × 14 SMX × 192 CUDA cores × 1 [I/c]).

Table 1: Target platforms.

	Titan	**Piz Daint**
Model	Cray XK7	Cray XC30
Host/CPU	Opteron 6274	Xeon E5-2670
Device/GPU	K20X	K20X
Nodes	18,688	5,272
Inter-node bw. [GB/s]	2.5	8.1
Aggregate GPU [I/s]	3.7×10^{16}	1.0×10^{16}
Aggregate CPU-GPU [I/s]	3.9×10^{16}	1.2×10^{16}

Type	**Model**	**[GB]**	**[I/s]**	**[I/c]**
GPU	K20X	6	2.0×10^{12}	2.7×10^{3}
CPU	Opteron 6274	32	0.14×10^{12}	64
CPU	Xeon E5-2670	32	0.33×10^{12}	128

Each CUDA core is capable of performing 1 [SP FLOP/c]. The K20X GPU is based on the NVIDIA Kepler microarchitecture, CUDA compute capability 3.5, featuring 6 GB of device memory.

Each SMX can schedule the work of 16 CUDA blocks concurrently, fragmented into warps of 32 threads. At every cycle, the four per-SMX schedulers pick four eligible warps out of a pool of 64, and fetch up to two instructions each, totaling a maximum of six warp-level instructions per SMX. Because of the dual-issue nature of the schedulers, the compute kernel must expose enough Instruction Level Parallelism (ILP) to reach the peak throughput. To maximize the number of eligible warps, and thus maximize the available work per cycle, each thread should require no more than 32 registers. The measured peak device bandwidth is about 180 [GB/s] and the measured peak throughput $1.8 \cdot 10^{12}$ [I/s]. The best measured per-warp throughput was 5.5 [I/c] as opposed to the nominal one of 6 [I/c].

The K20X exhibits a 3-to-1 throughput imbalance of DP versus SP instructions. We remark that in terms of throughput, integer instructions are 1.2X - 6X slower than SP instructions. These observations suggest that the K20X was designed to aggressively process SP instructions, over any other instructions.

SIMT advantages for DPD.

The irregular computational patterns of DPD exhibit a massively parallel, fine-grained heterogeneous workload. This heterogeneity leads to a number of control flow exceptions that is excessively large to be effectively handled explicitly on SIMD architectures. Even if masking instructions alleviate this problem, the burden of dealing with masks inadvertently impairs both coding productivity and kernel performance.

Furthermore, compared to SIMD architectures, SIMT provides a number of unmatched advantages, starting with the ability to discover vectorization at runtime. The detection of exceptional cases in the computation (such as data boundaries, misalignment, etc.) is delegated to the hardware. This dramatically increases the readability of the code and the productivity of the programmer. A second advantage is the ability of scattering/gathering data to/from arbitrary locations in the shared memory. These operations enable effectively to “homogenize” the irregular DPD workload into data-parallel-ready computation. Finally, with control flow instructions at warp-level, the cooperation among the threads within a warp can be dynamically adjusted to the available workload so as to minimize the overhead or the intra-warp imbalance.

5. DESIGN

The complexity of flows involving RBCs and CTCs in microscale geometries impacts adversely the performance of the simulations. In turn, this poses significant challenges in mapping effectively the computational patterns onto GPU accelerated supercomputers.

A number of design choices have been made in order to significantly improve the performance:

- Computation is performed in single precision
- Avoid global particle ids
- Neighbor lists are not stored in off-chip memory
- Fast Random Number Generator (RNG)
- Each kernel operates at most on two particle types

Time steps in DPD simulations are usually between 10^{-3}-10^{-2}, while the magnitude of the forces rarely exceed 10^{2}. In these units it takes hundreds of thousands of time steps to resolve one cycle of the tumbling of a single RBC. This suggests that the single precision mantissa is large enough (10^{6}) to capture the difference in the orders of magnitudes across the unknowns of the simulation. This observation renders the vast majority of the DPD computational patterns amenable for SP computation, which in turn accelerates up to 3X the time-to-solution.

A global particle id may simplify the management and diagnostics of the simulation. However, it represents an unnecessary overhead when involved in the computation of the forces or the dynamic creation of new solvent particles as well as new RBCs or CTCs. Similarly, the overhead of keeping a per-particle neighbor list is hardly amortized if we consider that high flow strain rates impose a complete reconstruction of the neighbor lists at every time step. Furthermore, the cost of the DPD interactions can be straightforwardly hidden by the irregular work required for producing the neighbor list.

The random force of the DPD interactions requires a computationally efficient random number generator (RNG). Some of the algorithmically most efficient RNGs are those based on the Tiny Encryption Algorithms (TEA) [66], such as the SARU algorithm [1]. For the K20X, a performance issue of the TEA-based RNGs is that they mostly feature integer instructions. As described below, we have developed a *novel RNG* based on the Logistic map which features exclusively FMA instructions.

In order to achieve high performance in the compute kernels we restrain them to operate concurrently on just two particle types. This enables us to apply type-specific optimizations when computing the interactions.

Most of the workload imbalance is expected to come from the irregularity of the microscale geometry. Since the geometry is known a priori, we introduce a pre-processing step to provide an optimized rank-to-node mapping to the application-level placement scheduler. The reordering assigns an heterogeneous number of MPI tasks to each node so as to equalize the per-node workload.

Memory layout.

The domain of the present simulations is decomposed in a cartesian fashion in parts of equal size and shape. We employ an Array-of-Structure (AoS) format that provides us

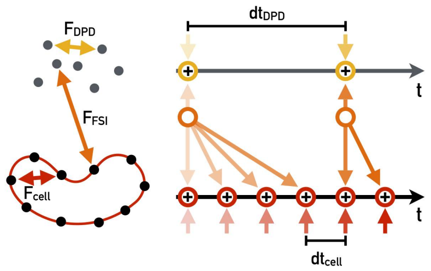

Figure 3: The separation of the temporal scales between solvent and cells allows us to consider multi-timesteps algorithms [58].

with the best tradeoff between spatial locality and effective memory access for DPD simulations. This layout is also reasonably efficient when packing/unpacking MPI messages containing particle information.

The representation of the wall geometry is entirely local to its respective subdomain. Once created the wall data are immutable objects. Hence, after the wall creation the associated information can be exchanged only once across neighboring MPI tasks, in a fashion similar to a halo exchange. This renders the handling of collisions between walls and DPD fluid as well as cells entirely local to each MPI task.

Cluster-level optimizations.

The design choices at this level aim to minimize the amount of communication and achieve maximum C/T-overlap. For the computation of the forces within the solvent, C/T-overlap is achieved by exchanging messages of a predefined size. After packing the messages on the GPU, each MPI task performs non-blocking point-to-point communication with its 26 surrounding tasks while computing the local DPD forces. Since the force evaluation takes $1 - 10$ ms and the communication takes $0.01 - 0.1$ ms, we expect the messages to arrive before the local computation has been completed. We then compute the inter-node DPD interactions by fetching directly the MPI messages from the GPU using zero-copy memory.

In certain cases the number of solvent particles to be communicated exceeds what can be held by a single message. These cases are safely detected and handled by sending a second message in a blocking fashion, whereas the C/T-overlap may be decreased for that timestep.

Node-level optimizations.

At node-level, the simulation code employs multiple CUDA streams to coordinate the execution of the different compute kernels. We rely both on the operating system and MPS to effectively run multiple MPI tasks with heterogeneous workloads on the node. We exploit the separation of the temporal scales between the solvent and the RBC membrane, since the viscosity of the latter is $10 - 100$ times larger. Our time scales separation approach is similar to the one discussed by Schlick [58]. Since the computation of the membrane forces is relatively inexpensive ($1\% - 5\%$ of the total workload), we are able to increase the DPD time step by a factor of 10 and updating just the cells at substeps of $1/10$ of a time step. As illustrated in Figure 3, for a coarse time step, we first calculate the total external forces (from the solvent and wall particles) on the cells, and then evolve the cells by adding internal forces of the cells to the external ones and integrating at finer time steps. These finer steps are entirely local to the node and do not require any MPI communication.

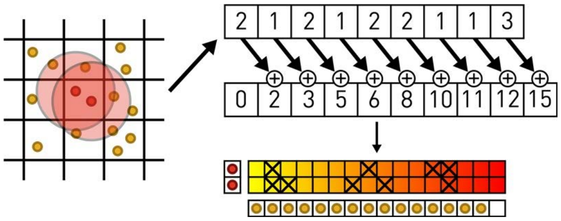

Figure 4: The workload of a cell is mapped to a warp so as to decrease warp divergence and work imbalance. Firstly, the particle count of the surrounding cells is fetched, then the prefix sum is computed.

GPU-level optimizations.

The most time consuming kernels are the ones computing the interaction forces, namely: *DPD, FSI* and *walls*. The *DPD kernel* computes the DPD interactions within each subdomain. The *FSI kernel* computes the interactions between the solvent particles and the particles representing the suspended cells. The *Wall* kernel is responsible for computing the interactions between the DPD particles and the wall particles. These kernels take advantage of Newton's 3rd law and work cooperatively on the cell-lists, minimizing warp divergence and maximizing temporal and spatial locality.

Figure 4 depicts how the computation of a cell list entry is mapped to a warp. A reindexing of the workload is performed through the shared memory. The reindexing consists of a prefix sum of the cell count of the 27 surrounding cells. The result is then shared across the CUDA threads, which are organized into an interaction matrix (16x2 in the case of Figure 4). For every thread, the corresponding source particle is found by performing a trinary search of the prefix sum.

All compute-bound kernels were subject to a "floatization" process, where integer instructions are replaced by FP instructions. This is implemented by reinterpreting integer variables as denormalized SP numbers and perform SP operations at full precision. The correctness of such operations is guaranteed by using inline PTX assemblies with explicit rounding and flushing modifiers.

We propose a novel stateless RNG that consumes mostly FMA instructions by using the Logistic map [21, 64], a nonlinear chaotic recurrence relation defined for real numbers. The series $X_{n+1} = 4X_n(1 - X_n)$, $X_n \in [0, 1]$ maps the interval $[0, 1]$ onto itself with a probability density function $p(x) = \left(\pi\sqrt{x(1-x)}\right)^{-1}, x \in [0, 1]$.

The recurrence relation is chaotic and hence generates unpredictable sequences after sufficiently large number of iterations. The distribution is symmetric on $[0, 1]$ with a mean of $\frac{1}{2}$ and a variance of $\frac{1}{8}$. A random number sequence of zero mean and unit variance can thus be obtained by shifting and scaling X_n.

We introduce Algorithm 1 for generating a *unique* sequence of random numbers for *each* pair of interacting particles. The algorithm combines a time-dependent global seed with particle indices into pairwise-unique seed values, which then get crunched through the Logistic pipeline for further randomization. A serial high quality RNG, in our case KISS [41], generates the global seed since only one value is used for each time step.

Algorithm 1 Logistic based parallel RNG for DPD.

```
// $k = k(t)$ is a time-dependent global random number sequence
// $G$ is the golden ratio $\frac{\sqrt{5}-1}{2} \approx 0.618\cdots$
// $S$ is another irrational number $\sqrt{2}-1 \approx 0.414\cdots$
function MeanOVar1(i,j,k)
    // Low-discrepancy number
    $u \leftarrow$ Min(i,j)
    $v \leftarrow$ Max(i,j)
    $p_{ij} \leftarrow$ Mod($u \times G + v \times S$,1)
    $y \leftarrow k - p_{ij}$

    // Pass through Logistic map
    for N rounds do
        $y \leftarrow 4y(1-y)$
    end for

    // Normalize
    $z \leftarrow$ Normalize($y$)
    return $z$
end function
```

The generator passes 158 out of 160 tests in BigCrush [36] with 18 Logistic iterations and achieves zero mutual information across sequences for different particle pairs.

6. SOFTWARE

The development of the present software[2] started on July 2014. Four main developers were involved for approximately 24 man-months. During this time our team followed the Extreme Programming principles developed by Kent Beck [5], including small releases, continuous integration, and test-driven development. The software is written in CUDA and MPI; it targets NVIDIA GPUs with compute capability 3.5 or higher. The resulting software consists of about 80,000 lines of code. Approximately 10,000 lines of code involve Object-Oriented programming and carry out the CPU-GPU coordination and MPI communication. The cell membrane model code takes about 15,000 lines of code, the code for the optimized DPD interactions takes about 5,000 lines of code and makes heavy use of PTX intrinsics. The remaining 50,000 lines of code consist of unit tests and proof-of-concepts.

The software implements Lustre-aware parallel I/O for data dumps, whose formats are compliant to ParaView [4]. The mesh data generated by the present simulations was rendered with the Mitsuba Renderer [30].

7. LARGE-SCALE SIMULATIONS

We demonstrate the capabilities of our software to perform simulations of cells separation in complex geometries, under different conditions by examining two devices: (I) the CTC-iChip [32] and (II) the Funnel Ratchets [42].

Device I: CTC-iChip.

In this work as a proof of the capabilities of our software we simulate flows of RBCs and CTCs in the CTC-iChip [32]. The CTC-iChip is a microfluidics system, with two modules, that has shown great potential for blood separation and capture of CTCs. The one-to-one geometric correspondence between the CTC-iChip and the geometries studied herein, allows us to confirm these features.

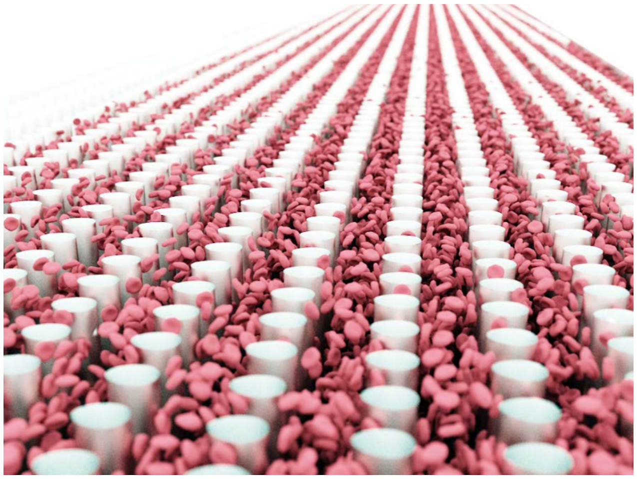

Figure 5: `Device I:` Simulation of the first module of the CTC-iChip with 200,000 RBCs spiked with 1 CTC (not visible - see also Figure 6, top).

In this work we focus on the first compartment of the device (CTC-iChip1), which separates CTCs and WBCs from RBCs by taking advantage of the deterministic lateral displacement effect [28]. The size of the CTC-iChip1 device is 150 μm tall, $\sim$ 750 μm wide and at least 3.8 mm long. Pillared obstacles form a microarray of 13 rows and at least 118 posts along the module, as illustrated in Figure 6 (top). The cross section of the posts is "egg"-shaped of size 17 μm along the X-direction and 24 μm in the Y-direction. This microarray is characterized by a gap of 32 μm between the rows, 15 μm between the columns and a 1.7° slope with respect to X-axis. This geometry affects the deflection of particles larger than 4 μm [28]. A blood sample is injected into the lower rows of the CTC-iChip1 together with a buffer injected in the top part with mass flow rates of 120 ml/min and 630 ml/min, respectively. The filtered product is extracted downstream at the device outlet.

The domain size is $1872\mu m \times 816\mu m \times 144\mu m$ with an array of 767 (59×13) obstacles (see Figure 6, top). The computational box is bounded in the Y and Z directions by flat walls, with periodic boundary conditions applied in X direction . The flow is driven by a uniform pressure gradient along the X-direction.

We initialize $200,000$ RBCs at uniformly random positions in the leftmost part of the domain with respect to Y direction (Figure 5). We insert among the RBCs a single CTC with a diameter of 14 μm. The simulation entails for 3×10^6 timesteps during which the CTC traversed the whole domain in the whole device $\sim$ 7.5 times. We observe that RBCs were primarily located around the obstacles at the bottom in accordance with the experimental snapshot, as depicted in Figure 7. In Figure 8 (left) we report the displacement of the CTC and RBCs in the Y-direction over time in order to assess the effectiveness of the device in separating the CTC from the RBC cells. We note that the CTC was drastically displaced in the Y-direction (Figure 8, left) allowing for its separation from the RBCs in agreement with the experiment.

[2] https://github.com/rossinelli/uDeviceX

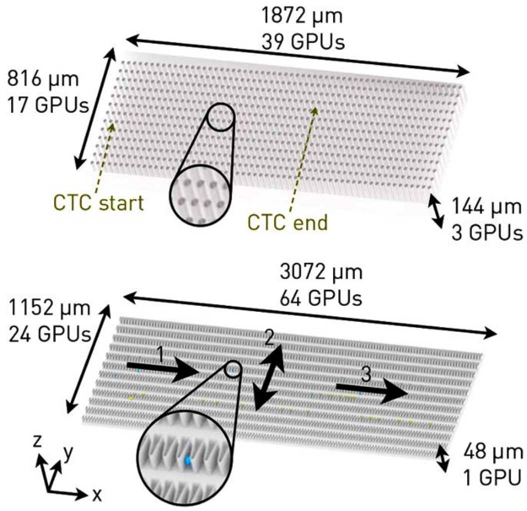

Figure 6: `Device I`: (Top) Stage 1 of the CTC-iChip indicating the initial and final position of the 1 CTC spiked in 200,000 RBCs. `Device II`: (Bottom) Funnel ratchets geometry.

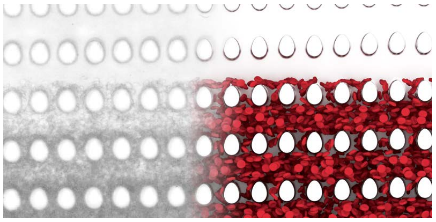

Figure 7: `Device I`, experiment [32] of RBC flows in the CTC-iChip geometry (left) and simulation (right).

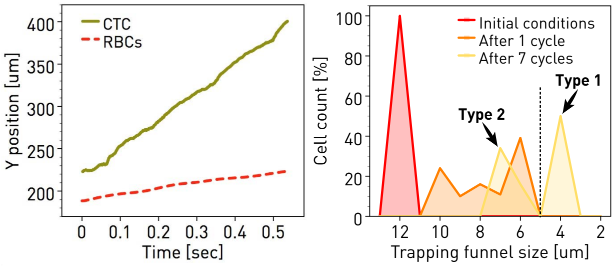

Figure 8: `Device I`, displacement in the Y-direction of the CTC versus time (left) and `Device II`, evolution of the cell distribution.

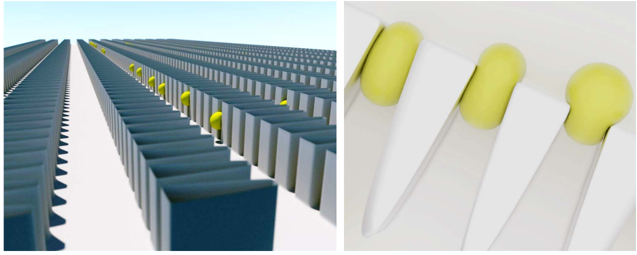

Figure 9: `Device II`, simulation of the funnels ratchet (left), CTCs are squeezing through funnel constrictions (right).

Device II: Funnel ratchets.

Here we focus on a functional stage of a device that sorts cells according to their deformability as well as size, by transporting them through funnel ratchets as discussed by McFaul et al. [42]. The asymmetric design of the device renders the cell-sorting irreversible, and is capable of reaching throughput of 9,000 cells per hour without clogging.

The device consists of microscale ratchets arranged in a 2D array of 12 rows having 128 funnels each (Figure 9). The funnel pore size is constant in each row and decreases by $1\,\mu m$ starting at $14\,\mu m$, reaching $2\,\mu m$ along the Y-axis of the device.

The filtering of CTC's involves three steps. Firstly, the cell sample is infused into the device from the left (see Figure 6, bottom, arrow 1). Then the oscillatory flow is applied along the Y axis of the device (arrow 2). This procedure is repeated several times (2 to 20) with a pressure difference across the device of $7 - 51$ kPa. Due to the asymmetric shape of the funnels, the cells get trapped between certain rows depending on their size and deformability (see Figure 7, right). To finally collect the separated cells, a flow parallel to the X-axis is applied until the cells leave the device (arrow 3, right).

We perform a numerical investigation of this problem by discretizing the domain into $1152 \times 3072 \times 48$ length units. Shape and separation of the tapers are exactly as they appear in the experiment. The box is bounded by flat walls in the Z direction, while periodic boundary conditions are applied in X and Y directions. This simulation does not involve RBCs, it is used to further isolate CTCs from WBCs [42].

In the simulation we initialize ~ 70 cells of two different types uniformly along the X direction in front of the first row of obstacles. Cells of type 1, mimicking WBCs, have a diameter of $14\,\mu m$, while cells of type 2, corresponding to CTCs, have a diameter of $20\,\mu m$. Type 2 cells are about 20% stiffer than type 1 cells.

In the simulation, the period of the oscillations is 1500 DPD time units and the positive force is maintained for 1125 time units, while the remaining 325 time units the force is reversed. We run $2.9M$ timesteps which corresponds to 9 full oscillatory cycles.

We observe cell separation even after the first cycle as illustrated in Figure 8 (right), even though irreversibility is achieved only after the fifth cycle. Interestingly, due to the flow properties and random initial conditions, type 2 cells end up getting trapped between the funnels as shown in Figure 9.

Table 2: Outperforming factor over LAMMPS.

Nodes	Ranks/node	Particles/rank	Outperforming factor	
			$N_R = 4$	$N_R = 1$
1	1	3,538,944	26.7	44.1
48	1	3,538,944	26.7	**44.6**
48	8	442,368	7.40	17.2
18600	1	3,538,944	22.2	**37.9**
18600	8	442,368	4.9	11.4

Table 3: Achieved throughput in terms of instructions and unknowns.

Metric	Measured value
Maximum [I/s]	2.58×10^{16}
Overall [I/s]	1.34×10^{16}
Unknowns/s	1.55×10^{13}

8. PERFORMANCE RESULTS

The present software and simulations have enabled the following achievements:

- *Peak performance:* our software reaches **65.5% of the nominal peak** for the most computationally-intensive kernel and **34% overall** nominal peak performance considering the aggregate CPU and GPU.
- *Time-to-solution:* our software outperforms the current, leading, state-of-the-art software LAMMPS[3] by a factor of up to **45X**.
- *Throughput:* we simulate 1.8 Billion cells/s and 1.55×10^{13} unknowns/s, a **three orders of magnitude improvement over state-of-the-art** blood flow simulations [56].
- *Complexity:* Simulations considered microfluidic devices with a volume of 132 mm^3 with at sub-micron resolution, handling up to **1.43 Billion deforming RBCs (0.15 -0.29 ml of blood for 80%-40% hematocrit respectively)**, each discretised with 500 elements. To the best of our knowledge, there have never been such large-scale simulations of blood flow in realistic geometries of microfluidic channel, at sub-micron resolution, accessing continuum time scales.

These results were collected on Titan with simulations at full scale and include I/O time. The performance was measured throughout the simulation with high-resolution timers and hardware counters with *nvprof.* More precisely, the overall aggregate IPC was measured by counting the total number of executed vector instructions over hundreds of timesteps, divided by the wall-clock time. The profiling was performed in the middle of the simulation.

Table 2 shows the comparison of our code with LAMMPS for a plane Poiseuille flow. The test case consisted of running 10^4 time steps of DPD solvent in a periodic domain with the number density of 4. We have varied the number of nodes, ranks per node and particles per rank as well as the frequency N_R of rebuilding neighbor lists in LAMMPS. The parameter N_R depends on the problem, e.g. flow problems exhibiting high shear strain rates and large time steps require $N_R = 1$. We observed a consistent improvement of time to solution with respect to LAMMPS of up to 45X.

Table 4: Weak scaling efficiency in percent.

Ranks/node	Nodes					
	1	300	1200	4800	10800	18600
1	-	98.8	92.4	89.9	82.1	80.5
2	-	99.6	**99.1**	97.7	94.0	87.9
4	-	99.9	**99.7**	**99.7**	**98.6**	**98.2**
8	-	99.9	**99.6**	**99.6**	**99.7**	**99.7**
16	-	99.9	**99.9**	**99.9**	**99.9**	**99.9**

Table 5: Strong scaling efficiency in percent (Piz Daint).

Nodes	Speedup	Efficiency
625	-	-
1250	1.98	98.9%
2500	3.82	95.4%
5000	7.50	**93.8%**

We benchmarked HOOMD-Blue[4] against the same flow problem. *The current version HOOMD-Blue is not capable of large-scale simulations using more than 124 Million particles.* Therefore the benchmark is limited to a maximum of 280 nodes for a constant workload of at least 442,368 particles per rank. Under this limitation, we report a constant gain from 3X to 4X over HOOMD-Blue.

Table 3 shows the peak performance of our code in terms of instructions and unknowns throughput. Our software reaches 65% of the nominal IPC peak performance of Titan. This leads to a rate of 13.4 PetaInstructions/s, effectively solving $15.5 \cdot 10^{12}$ [Unknowns/s].

A *close-to-perfect weak scaling* of our software is depicted in Table 4. As the number of ranks per node increases, we observe an excellent weak efficiency of more than 98% on the 18,688 nodes of Titan. Table 5 reports a 94% strong scaling efficiency on Piz Daint from 625 nodes to 5000 nodes.

Table 6 shows a summary of a series of large-scale simulations performed on Titan. The four cases considered are simulations involving solvent, RBCs and CTCs in complex geometry with volumes of up to 150 mm^3 and 1.43 Billion deforming cells, where we observe a computational throughput of 1.8 Billion cells/s.

GPU kernels.

We carried out a detailed analysis of the four most demanding CUDA kernels which take approximately 85% of the execution time on GPU (Figure 10). The analysis is based on the event counters and metrics supported by nvprof.

The *DPD* kernel computes the interactions and accounts

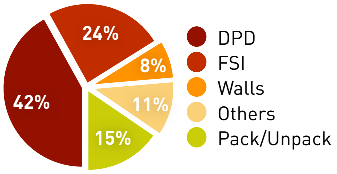

Figure 10: Distribution of the GPU execution time.

[3] LAMMPS GPU package, compiled for SP arithmetic.

[4] HOOMD-Blue was compiled for SP arithmetic.

Table 6: Summary of the large-scale simulations on Titan.

	Funnels ratchet	CTC-iChip1	Suspended RBCs
Simulation volume [mm^3]	33	39	94
Total cells	–	3.4×10^7	1.4×10^9
Total unknowns	5.9×10^{11}	8.6×10^{11}	6.0×10^{12}
RBC/s	–	1.7×10^8	1.8×10^9
Unknowns/s	4.1×10^{12}	4.3×10^{12}	7.5×10^{12}

for 35% of the total GPU time. The kernel reaches an executed IPC of 2.7, corresponding to 45% of the nominal GPU peak performance. The kernel performance limiters are instruction level parallelism and texture fetch latency. The execution dependency and texture fetch account for 33% and 25% of the latencies, respectively.

The FSI kernel computes the interactions between cells and solvent particles and accounts for 17% of the total GPU time. The IPC for this kernel is 2.5, corresponding to 43% of the nominal GPU peak performance. The performance limiter here is memory and texture bandwidth, the kernel exhibits 300 [GB/s] and 200 [GB/s] of L2 cache and texture bandwidth, respectively.

The computation of the interactions between the DPD particles and the wall achieves an IPC of 3.2, corresponding to 53% of the nominal peak. The performance of this computational stage is latency-bound. Latencies are significant due the irregularities of the wall geometry, even if the occupancy is above 84%. About 40% of stall reasons are attributed to memory dependencies due to the indirect access to the wall particles.

Time scales separation.

Taking advantage of the separation of the temporal scales within the simulation leads to a $5.5X$ gain in time-to-solution. The secondary benefit of time scales separation is an increase in the GPU utilization: the IPC is observed to increase up to **40% of the nominal peak performance overall**.

9. CONCLUSIONS

We present a software for large-scale simulations of flows in complex geometries of microfluidic devices involving RBCs, CTCs and WBCs. The software is based on the DPD method to simulate microscale flow phenomena at sub-cellular resolution and accesses time scales that so far have been accessible only by continuum solvers. In terms of time-to-solution, we outperform the current state-of-the-art DPD solvers by 38X-45X. The software demonstrates the potential to accelerate by an order of magnitude the design cycle for microfluidic systems for medical diagnosis and drug design.

Despite the irregular nature of the computational patterns involved in DPD simulations, the SIMT architecture was shown to deliver a significant performance gain. We describe a set of strategies and techniques to map DPD simulations on GPU-accelerated supercomputers with unprecedented efficiency. The unique features of the present software lead to full-scale simulations reaching peaks of up to 65%, and 34% in average, of aggregated nominal CPU-GPU peak performance. The present software redefines the frontier capabilities of simulations using multiscale particle based methods.

With the present software we were able to consider flow simulations involving up to 1.43 Billion deformable RBCs at sub-micron resolution. Domain sizes up to tens of cubic millimeters were simulated covering the *entire* functional compartments of microfluidics devices. We considered in particular two microfluidics devices: the CTC-iChip, which isolates CTCs from hundreds of thousands RBCs, and the funnels ratchet, which sorts CTCs from WBCs according to their size and deformability. Our simulations confirm experimental findings and help assess the operational efficiency of the two devices.

Ongoing work in with the present *in-silico Lab-on-a-Chip* includes shape optimization to improve the effectiveness of devices targeting the isolation of CTCs. Such optimizations are complemented with experiments and uncertainty quantification studies to obtain predictive simulations for a wide range of cell types. We believe that the present software can help revolutionise the rational design of devices critical to medical analysis and pharmaceutical research.

10. ACKNOWLEDGMENTS

We thank Gilles Fourestey (Swiss National Supercomputing Centre-CSCS) for his support throughout the project. We are grateful to Maria Grazia Giuffreda, Jean-Guillaume Piccinali and Thomas Schulthess (CSCS) for their invaluable assistance, to John Stone (UIUC) for his inspiring visualizations, Sauro Succi (CNR) for many discussions on mesoscale modeling and Panagiotis Angelikopoulos (ETHZ) for setting up Mitsuba. We thank Don Maxwell and Jack Wells as well as all the Titan support staff at the Oak Ridge National Laboratory (ORNL) that made this possible. PK acknowledges support by the European Research Council (ERC) Advanced Investigator Award (No.2-73985-14) and by the CSCS projects s436 and s448. GK acknowledges support by the DOE INCITE project bip118 and by the National Institutes of Health (NIH) Grant U01HL114476. This research used resources of the Oak Ridge Leadership Computing Facility at the ORNL, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

11. REFERENCES

[1] Y. Afshar, F. Schmid, A. Pishevar, and S. Worley. Exploiting seeding of random number generators for efficient domain decomposition parallelization of dissipative particle dynamics. *Comput. Phys. Commun.*, 184(4):1119–1128, 2013.

[2] C. Alix-Panabières and K. Pantel. Challenges in circulating tumour cell research. *Nat. Rev. Cancer*, 14(9):623–631, 2014.

[3] J. Anderson, C. Lorenz, and A. Travesset. General purpose molecular dynamics simulations fully implemented on graphics processing units. *J. Comput. Phys.*, 227(10):5342–5359, 2008.

[4] U. Ayachit. *The ParaView Guide: A Parallel Visualization Application*. Kitware, 2015.

[5] K. Beck. *Extreme Programming Explained: Embrace Change*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2000.

[6] M. Bernaschi, M. Bisson, T. Endo, S. Matsuoka, M. Fatica, and S. Melchionna. Petaflop biofluidics simulations on a two million-core system. In *Proc. of 2011 Intl. Conf. for High Perf. Computing, Networking, Storage and Analysis*, SC '11, pages 1–12, New York, NY, USA, 2011. ACM.

[7] M. Bernaschi, M. Bisson, M. Fatica, and S. Melchionna. 20 petaflops simulation of proteins suspensions in crowding conditions. In *Proc. of 2013 Intl. Conf. for High Perf. Computing, Networking, Storage and Analysis*, SC '13, New York, NY, USA, 2013. ACM.

[8] M. Bernaschi, M. Bisson, M. Fatica, S. Melchionna, and S. Succi. Petaflop hydrokinetic simulations of complex flows on massive GPU clusters. *Comput. Phys. Commun.*, 184(2):329–341, 2013.

[9] S. Chen and G. D. Doolen. Lattice Boltzmann Method for Fluid Flows. *Annu. Rev. Fluid Mech.*, 30:329–364, 1998.

[10] K. Choi, A. H. Ng, R. Fobel, and A. R. Wheeler. Digital Microfluidics. *Annu. Rev. Anal. Chem.*, 5(1):413–440, 2012.

[11] C. Cupelli, T. Borchardt, T. Steiner, N. Paust, R. Zengerle, and M. Santer. Leukocyte enrichment based on a modified pinched flow fractionation approach. *Microfluid Nanofluidics*, 14(3-4):551–563, 2013.

[12] E. Esch, A. Bahinski, and D. Huh. Organs-on-chips at the frontiers of drug discovery. *Nat. Rev. Drug Discov.*, 14(4):248–260, 2015.

[13] P. Español. Hydrodynamics from dissipative particle dynamics. *Phys. Rev. E*, 52:1734–1742, Aug 1995.

[14] D. Fedosov, B. Caswell, and G. E. Karniadakis. Wall shear stress-based model for adhesive dynamics of red blood cells in malaria. *Biophys. J.*, 100(9):2084–2093, 2011.

[15] D. Fedosov and G. Gompper. White blood cell margination in microcirculation. *Soft Matter*, 10:2961–70, 2014.

[16] D. Fedosov, W. Pan, B. Caswell, G. Gompper, and G. E. Karniadakis. Predicting human blood viscosity in silico. *Proc. Natl. Acad. Sci. U.S.A*, 108(29):11772–11777, 2011.

[17] D. Fedosov, I. V. Pivkin, and G. E. Karniadakis. Velocity limit in DPD simulations of wall-bounded flows. *J. Comput. Phys.*, 227(4):2540–2559, 2008.

[18] J. Freund. Numerical simulation of flowing blood cells. *Annu. Rev. Fluid Mech.*, 46(1):67–95, 2014.

[19] J. Glaser, T. Nguyen, J. Anderson, P. Lui, F. Spiga, J. Millan, D. Morse, and S. Glotzer. Strong scaling of general-purpose molecular dynamics simulations on gpus. *Comput. Phys. Commun.*, 192:97–107, 2015.

[20] S. C. Glotzer and W. Paul. Molecular and mesoscale simulation methods for polymer materials. *Annual Review of Materials Research*, 32(1):401–436, 2002.

[21] J. Gonzalez and R. Pino. A random number generator based on unpredictable chaotic functions. *Comput. Phys. Commun.*, 120(2–3):109–114, 1999.

[22] R. Groot and P. Warren. Dissipative particle dynamics: Bridging the gap between atomistic and mesoscopic simulation. *J. Chem. Phys.*, 107(11):4423–4435, 1997.

[23] Q. Guo, S. Park, and H. Ma. Microfluidic micropipette aspiration for measuring the deformability of single cells. *Lab Chip*, 12:2687–2695, 2012.

[24] M. Gusenbauer, I. Cimrak, S. Bance, L. Exl, F. Reichel, H. Oezelt, and T. Schrefl. A tunable cancer cell filter using magnetic beads: cellular and fluid dynamic simulations. *Manag. Sci.*, 2011.

[25] C. Hijon, P. Espanol, E. Vanden-Eijnden, and R. Delgado-Buscalioni. Mori-zwanzig formalism as a practical computational tool. *Faraday Discuss.*, 144:301–322, 2010.

[26] D. F. Hinz, A. Panchenko, T.-Y. Kim, and E. Fried. Motility versus fluctuations in mixtures of self-motile and passive agents. *Soft Matter*, 10:9082–9089, 2014.

[27] P. Hoogerbrugge and J. Koelman. Simulating microscopic hydrodynamic phenomena with dissipative particle dynamics. *EPL (Europhys. Lett.)*, 19(3):155, 1992.

[28] L. R. Huang, E. C. Cox, R. H. Austin, and J. C. Sturm. Continuous particle separation through deterministic lateral displacement. *Science*, 304(5673):987–990, 2004.

[29] S. Hur, N. Henderson-MacLennan, E. McCabe, and D. Di Carlo. Deformability-based cell classification and enrichment using inertial microfluidics. *Lab Chip*, 11:912–920, 2011.

[30] W. Jakob. Mitsuba renderer, 2010. http://www.mitsuba-renderer.org.

[31] F. Janoschek, F. Toschi, and J. Harting. Simulations of blood flow in plain cylindrical and constricted vessels with single cell resolution. *Macromol. Theory Simul.*, 20(7):562–570, 2011.

[32] N. Karabacak, P. Spuhler, F. Fachin, E. Lim, V. Pai, E. Ozkumur, J. Martel, N. Kojic, K. Smith, P. Chen, J. Yang, H. Hwang, B. Morgan, J. Trautwein, T. Barber, S. Stott, S. Maheswaran, R. Kapur, D. Haber, and M. Toner. Microfluidic, marker-free isolation of circulating tumor cells from blood samples. *Nat. Protoc.*, 9(3):694–710, 2014.

[33] G. Karniadakis, A. Beskok, and N. Aluru. *Microflows and nanoflows: fundamentals and simulation*, volume 29. Springer Science & Business Media, 2005.

[34] P. Koumoutsakos. Multiscale Flow Simulations Using Particles. *Annu. Rev. Fluid Mech.*, 37(1):457–487, 2005.

[35] P. Koumoutsakos, I. V. Pivkin, and F. Milde. The fluid mechanics of cancer and its therapy. *Annu. Rev. Fluid Mech.*, 45(1):325–355, 2013.

[36] P. L'Ecuyer and R. Simard. Testu01: Ac library for empirical testing of random number generators. *ACM Trans. Math. Software (TOMS)*, 33(4):22, 2007.

[37] H. Lei, D. Fedosov, B. Caswell, and G. E. Karniadakis. Blood flow in small tubes: quantifying the transition to the non-continuum regime. *J. Fluid Mech.*, 722:214–239, 2013.

[38] H. Lei and G. E. Karniadakis. Quantifying the rheological and hemodynamic characteristics of sickle cell anemia. *Biophys. J.*, 102(2):185–194, 2012.

[39] J. Li, M. Dao, C. Lim, and S. Suresh. Spectrin-level modeling of the cytoskeleton and optical tweezers stretching of the erythrocyte. *Biophys. J.*, 88(5):3707–3719, 2005.

[40] H. Liu, Q. Tan, W. Geddie, M. Jewett, N. Phillips, D. Ke, C. Simmons, and Y. Sun. Biophysical characterization of bladder cancer cells with different metastatic potential. *Cell Biochem. Biophys.*, 68(2):241–246, 2014.

[41] G. Marsaglia and A. Zaman. The KISS generator. Technical report, Dept. of Statistics, University of Florida, 1993.

[42] S. McFaul, B. Lin, and H. Ma. Cell separation based on size and deformability using microfluidic funnel ratchets. *Lab Chip*, 12:2369–2376, 2012.

[43] D. T. Miyamoto, L. V. Sequist, and R. J. Lee. Circulating tumour cells:monitoring treatment response in prostate cancer. *Nature Reviews Clinical Oncology*, 11(7):401–412, 07 2014.

[44] K. Muller, D. Fedosov, and G. Gompper. Margination of micro- and nano-particles in blood flow and its effect on drug delivery. *Sci. Rep.*, 4, 2014.

[45] L. Nan, Z. Jiang, and X. Wei. Emerging microfluidic devices for cell lysis: a review. *Lab on a Chip*, 14:1060–1073, 2014.

[46] B. J. Nelson, I. K. Kaliakatsos, and J. J. Abbott. Microrobots for minimally invasive medicine. *Annu. Rev. Biomed. Eng.*, 12:55–85, 2010.

[47] T. Nguyen and S. Plimpton. Accelerating dissipative particle dynamics simulations for soft matter systems. *Comput. Mater. Sci.*, 100:173–180, 2015.

[48] W. Pan, I. V. Pivkin, and G. E. Karniadakis. Single-particle hydrodynamics in dpd: A new formulation. *EPL (Europhys. Lett.)*, 84(1):10012, 2008.

[49] K. Pantel and M. R. Speicher. The biology of circulating tumor cells. *Oncogene*, pages –, 06 2015.

[50] Z. Peng, X. Li, I. V. Pivkin, M. Dao, G. E. Karniadakis, and S. Suresh. Lipid bilayer and cytoskeletal interactions in a red blood cell. *Proc. Natl. Acad. Sci. U.S.A*, 110:13356–13361, 2013.

[51] C. Phillips, J. Anderson, and S. Glotzer. Pseudo-random number generation for Brownian Dynamics and Dissipative Particle Dynamics simulations on GPU devices. *J. Comput. Phys.*, 230(19):7191–7201, 2011.

[52] D. Pinho, T. Yaginuma, and R. Lima. A microfluidic device for partial cell separation and deformability assessment. *Biochip J.*, 7:367–374, 2013.

[53] I. V. Pivkin and G. E. Karniadakis. Accurate coarse-grained modeling of red blood cells. *Phys. Rev. Lett.*, 101(11):1–4, 2008.

[54] S. Plimpton. Fast parallel algorithms for short-range molecular dynamics. *J. Comput. Phys.*, 117(1):1–19, 1995.

[55] D. Quinn, I. V. Pivkin, S. Wong, K. Chiam, M. Dao, G. E. Karniadakis, and S. Suresh. Combined simulation and experimental study of large deformation of red blood cells in microfluidic systems. *Ann. Biomed. Eng.*, 39(2):1041–1050, 2011.

[56] A. Rahimian, I. Lashuk, S. Veerapaneni, A. Chandramowlishwaran, D. Malhotra, L. Moon, R. Sampath, A. Shringarpure, J. Vetter, R. Vuduc, D. Zorin, and G. Biros. Petascale direct numerical simulation of blood flow on 200k cores and heterogeneous architectures. In *Proc. of the 2010 ACM/IEEE Intl. Conf. for High Perf. Computing, Networking, Storage and Analysis*, SC '10, pages 1–11, Washington, DC, USA, 2010. IEEE Computer Society.

[57] J. W. Scannell, A. Blanckley, H. Boldon, and B. Warrington. Diagnosing the decline in pharmaceutical r&d efficiency. *Nat Rev Drug Discov*, 11(3):191–200, 03 2012.

[58] T. Schlick. Time-trimming tricks for dynamic simulations: splitting force updates to reduce computational work. *Structure*, 9(4), 2001.

[59] T. M. Squires and S. R. Quake. Microfluidics: Fluid physics at the nanoliter scale. *Rev. Mod. Phys.*, 77:977–1026, 2005.

[60] T. Steiner, C. Cupelli, R. Zengerle, and M. Santer. Simulation of advanced microfluidic systems with dissipative particle dynamics. *Microfluid Nanofluidics*, 7(3):307–323, 2009.

[61] A. Streets and Y. Huang. Chip in a lab: Microfluidics for next generation life science research. *Biomicrofluidics*, 7(1), 2013.

[62] Y. Tang and G. E. Karniadakis. Accelerating dissipative particle dynamics simulations on gpus: Algorithms, numerics and applications. *Comput. Phys. Commun.*, 185(11):2809–2822, 2014.

[63] D. Visser, H. Hoefsloot, and P. Iedema. Comprehensive boundary method for solid walls in dissipative particle dynamics. *J. Comput. Phys.*, 205(2):626–639, 2005.

[64] N. Wagner. The logistic lattice in random number generation. In *Proc. of the Annual Conf. on Communication Control and Computing*, volume 30, pages 922–922. University of Illinois, 1992.

[65] Q. Wei, Y. Xu, F. Tian, T. Gao, X. Tang, and W. Zu. Ib-lbm simulation on blood cell sorting with a micro-fence structure. *Bio-Med. Mater. Eng.*, 24:475–481, 2014.

[66] D. J. Wheeler and R. M. Needham. Tea, a tiny encryption algorithm. In *Fast Software Encryption*, pages 363–366. Springer, 1995.

[67] G. Whitesides. The origins and the future of microfluidics. *Nature*, 442(7101):368–372, 2006.

[68] D. Xu, E. Kaliviotis, A. Munjiza, E. Avital, C. Ji, and J. Williams. Large scale simulation of red blood cell aggregation in shear flows. *J. Biomech.*, 46(11):1810–1817, 2013.

[69] C. Yung, J. Fiering, A. Mueller, and D. Ingber. Micromagnetic-microfluidic blood cleansing device. *Lab on a Chip*, 9:1171–1177, 2009.

[70] P. Zhang, N. Zhang, Y. Deng, and D. Bluestein. A multiple time stepping algorithm for efficient multiscale modeling of platelets flowing in blood plasma. *J. Comput. Phys.*, 284:668–686, 2015.