Developments for the Efficient Self-coupling of HemeLB

Patronis, A.[1], McCullough, J.W.S.[1], Richardson, R.A.[1], Halver, R.[2], Marshall, R.[3], Ruefenacht, M.[3], Wylie, B.J.N.[2], Odaker, T.[4], Wiedemann, M.[4], Lloyd, B.[5], Neufeld, E.[5], Sutmann, G.[2,6], Skjellum, A.[3], Kranzlmüller, D.[4], and Coveney, P.V.[1,7]

[1]Centre for Computational Science, University College London, UK
[2]Jülich Supercomputing Centre, Jülich-Rechenzentrum, Germany
[3]SimCenter, University of Tennessee at Chattanooga, USA
[4]Leibniz Supercomputing Centre, Leibniz-Rechenzentrum, Germany
[5]IT'IS Foundation, Switzerland
[6]ICAMS, Ruhr-University Bochum, Germany
[7]Institute of Informatics, University of Amsterdam, The Netherlands

May 21, 2019

1 Introduction

The aim of this paper is to document recent methodological advancements that enable extreme scale simulation by HemeLB [1], our present lattice-Boltzmann (LB) based blood-flow solver. From pre-processing to simulation and finally post-processing, we demonstrate the entire workflow on SuperMUC-NG, a state-of-the-art high performance computing platform. Pre-processing involves the voxelisation of patient-specific geometries to form a large lattice consisting of up to tens of billions of sites. Our ultimate goal is to enable the simulation of virtual humans, or digital replicas, with HemeLB simulating the full arterial and venous trees and exchanging information with simulation tools responsible for capturing the behaviour of other organ systems. To create accurate digital patients, we rely on some of the recent advancements discussed here.

2 Methodological Advancements

The first phase of HemeLB's execution, referred to as initialisation, and before simulation, loads and decomposes the lattice generated during pre-processing on hundreds of thousands of cores. A number of notable enhancements have been made to allow for efficient initialisation at this scale:

1. The development of a coupling scheme to communicate flow variables between separate instances of HemeLB, each simulating a unique (and physically disconnected) domain.

2. Conversion of large vector containers (i.e. `std::vector`), better suited for dense geometries, to associative containers (i.e. `std::map`) for a drastically reduced memory footprint.

3. Incorporation of our own 64-bit clean MPI (referred to here as BigMPI) to load data from file in an efficient manner. Current implementations of MPI use a 32-bit signed integer to specify a count variable, restricting the number of elements that can be operated upon (in a single MPI call) to 2^{31}. This limits our ability to efficiently read large data sets from file. BigMPI addresses this shortcoming by supporting counting to 2^{63} elements. Alternative strategies to read large datasets (e.g. to read more than two billion bytes ($2^{31} - 1$ to be precise) using `MPI_File_read_at`) involve the use of derived data types or '31-bit data chunking' [2], by which multiple MPI calls read data in smaller chunks; these methods of reading data are suboptimal.

4. To quickly and efficiently decompose large lattices over hundreds of thousands of MPI ranks, we resort to a new load-balancing library, simply known as ALL (A Load-balancing Library).

We assess the capabilities of ALL relative to HemeLB's own 'basic decomposition' method. Basic decomposition has been very successful to now, but it is expected that ALL will provide a better work distribution, and therefore improved load balance.

The improvements listed above are all necessary as we push to deploy on multi-petascale and emerging exascale machines. The available memory per core is set to fall dramatically, emphasising the need for great care when developing memory-intensive applications. In preparing HemeLB to fully exploit the largest multi-core machines available, we have come to appreciate the critical task of memory optimisation. In some cases we revert to bit packing and shifting to reduce data size, concepts that are fundamental to low-level hardware or embedded programming where available memory is severely limited – this serves to demonstrate the need for memory efficient code.

2.1 BigMPI

HemeLB is a demonstrator of some concepts and technologies set to define the new MPI 4.0 standard. BigMPI represents one aspect of this, but we have also identified issues relating to the allocation of internal MPI buffers when communicating large amounts of data from a subset of cores. Work to address these issues is ongoing. While the various MPI tuning knobs of the specific MPI implementation in use can be invoked to alleviate such issues, and in fact should be used to improve efficiency at extreme scales, tuning is usually time consuming and not portable; tuning options tend to be specific to the machine/implementation combination. We will be presenting these concerns to the MPI Forum for consideration and potential addressal by the MPI 4.0 standard.

2.2 Self-coupling

At number eight of the 52nd edition of the TOP500 list (November 2018), SuperMUC-NG bundles compute nodes into 8 domains (islands). Within one island, the OmniPath network topology is a 'fat tree' for highly efficient communication. The OmniPath connection between the islands is pruned (pruning factor 1:4). Tests will be performed to assess the impact of scaling over multiple islands. If performance is found to suffer, as with the previous generation of SuperMUC machine hosted at the Leibniz-Rechenzentrum, we can resort to multiscaling: a strategy by which we couple multiple HemeLB instances and communicate boundary values every n simulation steps. This concept represents a step towards realising the goals of the Virtual Human project. For the purposes of this paper we adopt a coupled approach to modelling on SuperMUC-NG – we couple 3D blood-flow simulation in the human arterial and venous trees segmented from cryosection images from the Visible Korean female [5], with each simulation running on a (single) separate island. As stated previously, a bi-directional exchange of information will occur every n steps, reducing (potentially limiting) communication between islands. We are effectively designing our simulation to make best use of SuperMUC-NG's unique topology. To make use of all islands, while respecting that inter-island communication may incur a performance penalty, other modelling tools will be used and confined to execution on any single island. For example, simulation of the human heart by Alya Red [3] may also be included and coupled to the inlet/outlets of the arterial/venous tree.

2.3 Extreme Scale Performance

Although we do not attempt to run a single gargantuan job on SuperMUC-NG, by which we may utilise all 311,040 compute cores across 8 islands, HemeLB has been shown to scale (strong scaling) very well (to within almost 80% of linear scaling) to 288,000 cores (18,000 XE nodes of NCSA Blue

Waters). Blue Waters (BW) [4] is composed of AMD 6276 Bulldozer-based Interlagos processors, with each consisting of 8 (floating point) cores. On the XE nodes there are 2 Interlagos processors. BW employs the Cray Gemini interconnect, which implements a configuration (3D torus topology) that has been shown to enable extremely large simulations on hundreds of thousands of traditional CPUs. Approximately 10.3 billion lattice sites were used to simulate blood flow in the circle of Willis (corresponding to a lattice spacing of approximately 6.3 microns). Figure 1 shows the scaling behaviour of HemeLB on BW. We see a drastic drop in performance between 18,000 and 21,000 XE nodes; at close to full-machine utilisation ($\sim$ 93% of 22,640 XE nodes), performance is highly susceptible to, for example, any network congestion, operating system (OS) jitter, or any other performance issues of any single node. Synchronisation across all ranks occurs at every simulation step, and so simulation progress is limited by any hardware performance variation or failure – an issue that will worsen as machines grow in complexity and the number of hardware components increases.

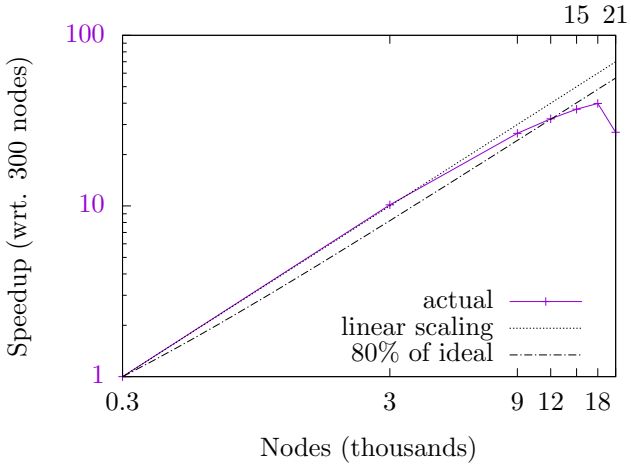

Figure 1: Strong scaling of HemeLB on NCSA Blue Waters. One node consists of 16 (floating point) cores (which translates to 16 MPI ranks), and offers 64 GB of memory.

3 Arterial-venous Coupling

In order to couple the arterial and venous trees, a physical and computational strategy has been developed to ensure that the boundary conditions of both domains can be adequately fulfilled. Firstly it was noted that, in the available geometries, there are many more inlets to the venous domain than there are outlets of the arterial domain. To accommodate this, a mapping was developed to link inlets and outlets. This links all arterial outlets to the nearest venous inlet, and ensures that no outlets are left uncoupled. Following this, the remaining inlets are coupled to the nearest outlet. As part of this mapping, the distance between each pair, and their respective areas, are noted.

From a physical perspective, two phenomena must be accounted for when representing the absent capillary networks between trees. The first is that the mass of fluid leaving the arterial tree and entering the venous tree is conserved, i.e. mass conservation. The second is that the pressure drop through the narrow capillaries is represented appropriately. This is achieved by scaling the outlet (arterial) velocity and pressure to those required at the paired (venous) inlets.

The coupling strategy is verified through multiple test cases of various connected networks. The first case of one outlet coupled to one inlet demonstrates that mass is conserved through a simple pipe, and that a desired pressure drop can be enforced. The further cases of one-to-many and many-to-many further illustrate that these fundamental properties are maintained in cases that better represent a coupling between arterial and venous trees. A full-body network of vessels can be seen in Figure 2; here, we simulate blood flow from the base of the spine (at the bifurcation) to the feet.

4 Conclusion

In a push to accelerate scientific discovery by deploying HemeLB on multi-petaflop computing platforms, we have developed and implemented a number of enhancements to support the simulation of large cardiovascular flows, performed with up to $\mathcal{O}(10^{10})$ lattice sites on $\mathcal{O}(10^5)$ processors. We demonstrate the latest capabilities by application to a coupled-flow problem: an arterial-venous coupling where we communicate flow behaviour between arterial outlets and venous inlets in a subsection of the "Yoon-sun" model.

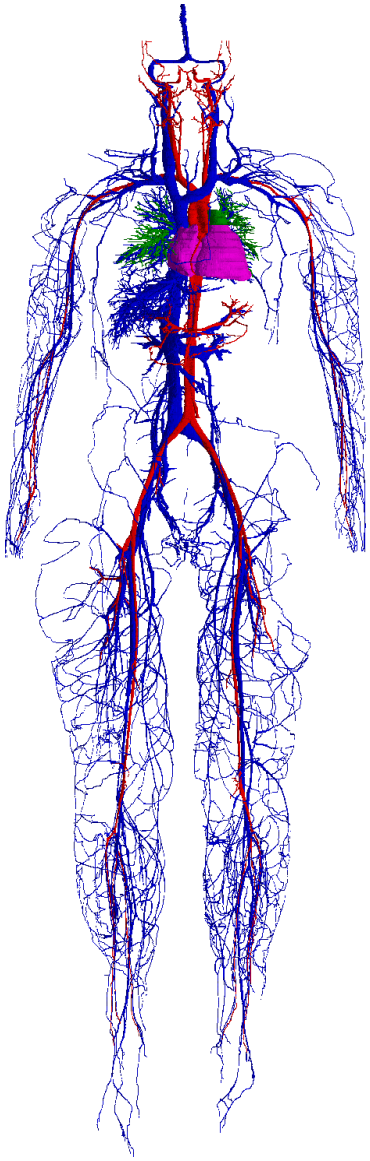

Figure 2: Network of systemic arteries (red), pulmonary arteries (green), veins (blue) and heart (pink) in the Virtual Population model "Yoon-sun" (https://itis.swiss/virtual-population/virtual-population/vip3/yoon-sun/; based on [5]). The intracranial vasculature is not included in this network.

5 Acknowledgements

We acknowledge funding support from CompBioMed (Grant No. 675451). Access to NCSA Blue Waters provided by NSF Grant No. 1713749.

References

[1] Derek Groen, James Hetherington, Hywel B. Carver, Rupert W. Nash, Miguel O. Bernabeu, and Peter V. Coveney. Analysing and Modelling the Performance of the HemeLB Lattice-Boltzmann Simulation Environment. *Journal of Computational Science*, 4(5):412–422, 2013.

[2] Jeff Hammond, Andreas Schäfer, and Rob Latham. To INT_MAX... and beyond! Exploring large-count support in MPI. *Proceedings of the Workshop on Exascale MPI*, 2014.

[3] Guillermo Marin, Fernando Cucchietti, Mariano Vázquez, Carlos Tripiana, Guillaume Houzeaux, Ruth Arís, Pierre Lafortune, and Jazmin Aguado-Sierra. Alya Red: A Computational Heart. *Science*, 339:518–519, 2013.

[4] Celso L. Mendes, Brett Bode, Gregory H. Bauer, Jeremy Enos, Cristina Beldica, and William T. Kramer. Deploying a Large Petascale System: The Blue Waters Experience. *Procedia Computer Science*, 29:198–209, 2014.

[5] Jin Seo Park, Min Suk Chung, Sung Bae Hwang, Yong Sook Lee, Dong-Hwan Har, and Hyung Seon Park. Visible Korean Human: Improved serially sectioned images of the entire body. *IEEE Transactions on Medical Imaging*, 24(3):352–360, 2005.