Jh160055-NAHI

Cerebrospinal Fluid Flow Analysis in Subarachnoid Space

Ryusuke Egawa（Cyberscience Center, Tohoku University）

Abstract
A better understanding of the hydrodynamics of the cerebrospinal fluid (CSF) is important to understand the pathophysiology of various diseases related to the central nervous system (CNS). This project aims to simulate the CSF flow in subject specific subarachnoid spaces (SAS) of healthy volunteers and patients suffering from Chiari malformation. The simulations at high spatial and temporal resolutions are conducted using the Lattice Boltzmann Method (LBM) based solver Musubi, which is ported on the SX-ACE for efficient and scalable direct numerical simulation of the CSF. In this report, implementation and performance analysis of MUSUBI on SX-ACE are presented.

1. Basic Information

(1) Collaborating JHPCN Centers

Cyberscience Center, Tohoku University.

(2) Research Areas

- ■ Very large-scale numerical computation
- □ Very large-scale data processing
- □ Very large capacity network technology
- □ Very large-scale information systems

(3) Roles of Project Members

This joint research is composed of Tohoku University and Siegen University (Germany). The members of Tohoku University mainly work for the performance analysis and optimization, and the members of Siegen University work for code development of a CFD simulation code based on LBM to analyze the flow of cerebrospinal fluid in subarachnoid spaces. All members and their tasks in this project are listed as follows:

Tohoku University (JAPAN)

- Ryusuke Egawa (Performance Analysis / Code optimization)
- Hiroyuki Takizawa (Code Optimization)
- Kazuhiko Komatsu (Performance Analysis / Code optimization)
- Hiroaki Kobayashi (Performance Analysis)

Universität Siegen (Germany)

- Sabine Roller (Code Design)
- Harald Klimach (Performance Analysis / Code Design)
- Kartik Jain (Code Design)
- Jiaxing Qi (Performance Analysis / Code Design)

2. Purpose and Significance of the Research

The overall goal of this research is the simulation of Cerebrospinal fluid (CSF) in subject specific subarachnoid spaces (SAS), with employment of high spatial and temporal resolutions. To perform a direct numerical simulation of hydrodynamics of the CSF in large geometries of SAS, a discretization that results in up to two billion cells is needed. Therefore, the simulation requires large computational resources and an efficient implementation of the underlying algorithms.

We are using the Lattice-Boltzmann method to simulate the incompressible flow. This method has some favorable features for large-scale, transient, incompressible flow simulations in

complex geometries. It offers a relatively simple kernel with few operations that is good to vectorize. However, it comes also with a relatively high Bytes/FLOP ratio. This typically results in memory-bound computations on usual x86 based architectures and an implementation on modern vector supercomputer SX-ACE would offer a more efficient execution.

A detailed understanding of the hydrodynamics of the CSF has been elusive and computational methods when high performance supercomputers can help in gaining better insights into the pathophysiology of disorders related to the CNS. Chiari malformation type I (CMI) is a condition characterized by herniation of cerebellar tonsils in the spinal canal through the foramen magnum. The crowding of tissue resulted by CMI causes obstruction to CSF outflow.

Computational Fluid Dynamics (CFD) has evolved as an important tool for the simulation of such complex physiological flows. The CFD studies, however, due to the low Reynolds number (typically 100 ~ 300 due to varying diameters along the SAS) of CSF flow, rely on the assumption of laminar CSF flow. The complexity of the SAS, constriction of the cranio-vertebral junction as a result of Chiari malformation, and the oscillating behavior of the CSF, however, rejects this common assumption and have motivated us to look into detailed CSF hydrodynamics using highly resolved direct numerical simulations.

Further research on this topic by incorporation of varying flow rates and larger parts of the spinal canal would be necessary to understand the phenomena better. Besides, this study will provide improved understanding to clinicians, and help them make decision for intervention.

The Lattice-Boltzmann method is an important tool to investigate incompressible transient flows, in complex geometries. In contrast to most classical approaches to incompressible flows, this stencil-based method does not need implicit time integration and thereby avoids the need to solve a linear equation system. This time integration enables a highly efficient computation well suited for massively parallel systems.

Another major advantage of this method is the ease to describe complex geometries. The Lattice-Boltzmann method relies on discrete space directions, and boundaries need to be defined only at these one-dimensional lines. Such line intersections with arbitrary complex surface descriptions can be generated with relative ease. Advancing this method and increasing its flexibility can help a wide range of CFD applications.

3. Significance as a JHPCN Joint Research Project

From the scientific aspects, we can indeed obtain significant results as described in the previous section by accomplishing this project. However, to carry out the project, the active collaboration between computational and computer scientists is mandatory to exploit the potentials of supercomputers. Therefore, JHPCN can provide us an excellent opportunity to conduct our joint interdisciplinary research.

Besides, Germany has different supercomputing systems with JAPAN such as SuperMUC, JUGENE, etc. Currently, there are no large-scale vector systems that can accelerate our target code in Germany. Hence, this project makes to analyze their performance quantitatively possible, and these kinds of analysis are necessary for application

developments and also useful to design future supercomputing systems.

4. Outline of the Research Achievements up to FY 2015

N/A

5. Details of FY 2016 Research Achievements

5.1 MUSUBI

MUSUBI is an incompressible fluid flow solver based on the LBM. Instead of discretizing the Navier-Stokes equation directly, the LBM emerges from a highly simplified gas-kinetic description. It operates on structured Cartesian meshes and requires information only from the nearest neighbors. Such data locality makes LBM a good candidate for parallelization.

Moreover, the simple link-wise boundary treatment makes LBM very suitable for handling complex geometries. Strategies and techniques for the large-scale LBM parallelization have gained significant attention in recent years. The flow solver MUSUBI is a part of the APES simulation framework that is designed and implemented by Vice-PI Sabine Roller's group.

APES is based on a distributed octree mesh representation library TreElM. Using a dedicated mesh representation, the common geometrical and topological operations on octree are encapsulated into the library. Together with the mesh generator and the post-processing tool, the APES framework provides users an end-to-end high-performance simulation tool chain. MUSUBI has been successfully applied in complex engineering applications like the flow of liquid mixtures through a complex spacer as well as clinical applications like blood flow in intracranial aneurysms. More detailed information regarding APEs can be found in the following reference.

[Reference] **H. Klimach, K. Jain, and S, Roller, "End-to-end parallel simulations with apes," In Parallel Computing: Accelerating Computational Science and Engineering (CSE), volume 25 of Advances in Parallel Computing, pages 703-711, Munich, Germany, September 2014. IOS Press.**

5.2 MUSUBI on SX-ACE

NEC SX-ACE vector system has installed at Cyberscience Center, Tohoku University. The whole system consists of 2,560 4-core nodes and 64 GB total memory in each node. With four cores, each node can deliver a peak performance of 256 GFLOPS along with a peak memory bandwidth of 256 GB/s, resulting in a high machine balance B/F (Byte to FLOPS ratio) of one. The SX-ACE CPU also has a cache similar design called Assignable Data Buffer (ADB).

The NEC sxf03 rev.020 Fortran compiler was used to compile our code. However, since the compiler cannot fully vectorize the code, we made some code optimizations for improving the vector operation ratio.

For example, intermediate arrays to store results within larger loops prevent efficient vectorization. Regarding the data size, the compiler attempts to vectorize over small arrays. So by eliminating short arrays and adding compiler directives for explicating the data dependency, we can achieve a certain level of vector operation ratios and vector length. As a result, the kernels perform well as we expected.

5.2.1 Single Node Performance

For the performance evaluation, a fully periodic cubic geometry without any boundary

conditions is selected as the test bed. Performances under various scenarios (intra-node and inter-nodes) are examined. The million lattice updates per second (MLUPS) is chosen as the performance metric. For a given compute kernel, this number can be converted directly into the number of floating point operations. The BGK relaxation scheme of the LBM and the D3Q19 layout were chosen throughout for the performance evaluation. The use of the D3Q19 layout implies 19 8-byte real numbers for each cell. To implement double buffering and to store adjacency information of 1D 4-byte integer array, a total of (8 x 3 x 19 + 4 x 19=) 532 bytes data volume per element are required for every time step. These data to be transferred over the memory interface, assuming a write allocate cache strategy. MUSUBI have not used any turbulence modeling and all the spatial and temporal scales have been resolved down to the Kolmogorov micro-scales. The MRT collision scheme of the LBM is used for higher stability in complex oscillating flow.

After applying compiler optimizations, we measured that the BGK kernel requires 160 floating point operations (FLOPS) per element at every time step. The Bytes to FLOPS ratio of the generated code reaches at 3.325, which indicates the performance is limited by main memory bandwidth.

Figure 1 depicts the single node performances for two different problem sizes 128^3 and 256^3. The SX-ACE CPU has fewer cores than general scalar processors. However, each core behaves like a strong core: a single core achieves a performance of more than 100 MLUPS, far better than the CPUs on other systems. The maximum performance within a single node is 310 MLUPS, which corresponds to a memory bandwidth of 165 GB/s, i.e. 64.4% of the theoretical peak memory bandwidth.

5.2.2 Multi Node Performance

Figure 2 shows inter-node performance of MUSUBI on SX-ACE when all the 4 cores within a node are utilized. For small problem sizes, the non-computation overheads dominate the simulation, resulting in low performance. The performance curves for 8 up to 256 nodes are closer to each other portraying an improved scaling behavior.

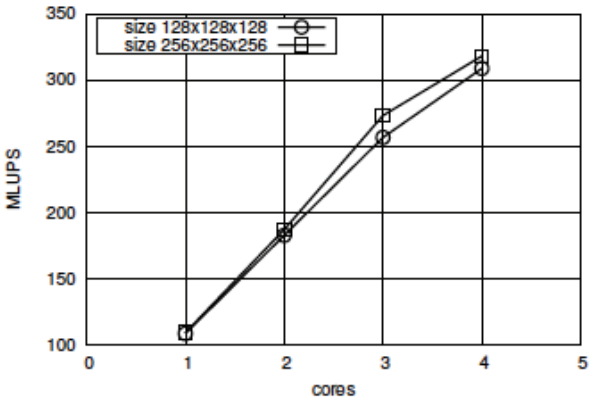

Fig.1 Node Performance of MUSUBI on SX-ACE.

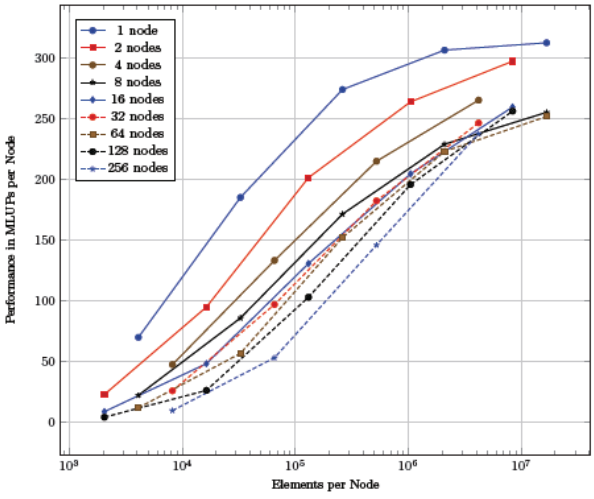

Fig.2 Parallel Performance of MUSUBI on SX-ACE.

In order to analyze the low performance in the case of small problem sizes, Figures 3(a) and 3(b) show the communication costs of the

main iteration part of MUSUBI with different problem sizes. For small problem sizes, the communication overhead inhibits scalability as shown in Figure 3(a), unlike larger problem sizes as shown in Figure 3(b).

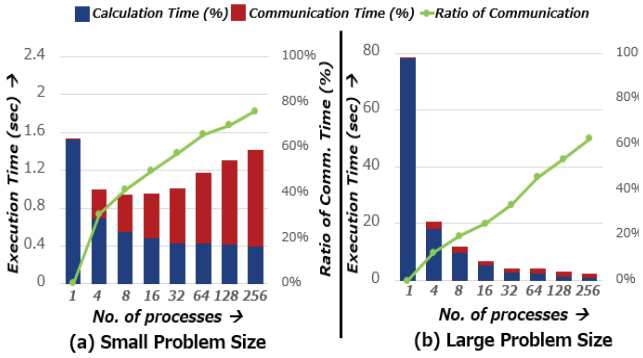

Fig.3 Communication cost for the iteration part.

Figure 4 shows the memory footprint size required for each MPI process. This figure shows that the communication overhead for the small problem size is prominent because the minimum memory required by each process is almost close to the total problem size. This fact leads the situation that the data does not distribute linearly with the increasing number of MPI processes.

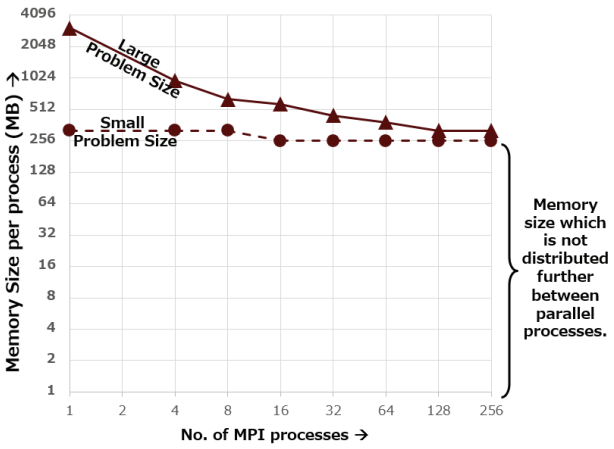

Fig.4 Memory Size per process

Figure 5 shows the breakdown of the execution time. This figure shows that the MPI overhead for communication of data among processes increases with increasing number of processes. Consequently, the performance gets better upon increasing the problem sizes as the computation to communication ratio increases.

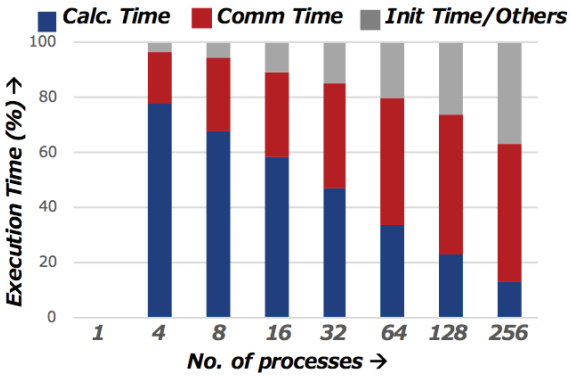

Fig.5 Breakdown of the execution time

Figure 6 shows the scalability when the number of processes changes. This figure indicates that the scalability of the iteration part also improves with the increasing problem sizes. While the small problem sizes show poor scalability due to the earlier discussed reasons, as the problem size becomes larger, the program tends to show better scalability.

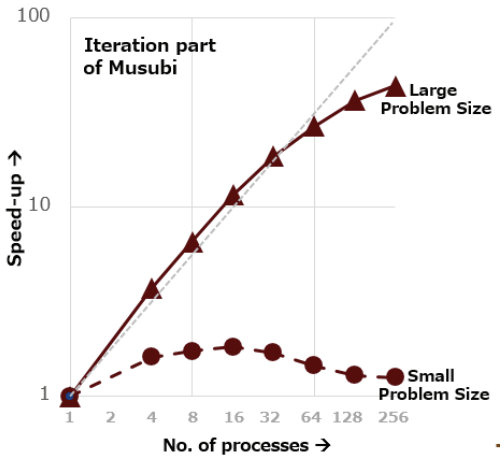

Fig.6 Speed-up of iteration part

For the iteration part, the communication among MPI processes are well balanced. Table 1 shows the communication costs in the man

iteration part with four and eight parallel processes cases. The memory size of the communicated data is similar for all the process ranks and the number of MPI communications are also similar for the shown parallel processes. Therefore, the execution time per process is also observed to be similar.

The evaluation results show that SX-ACE effectively accelerates the kernel of MUSUBI through high vector ratio and average vector length. However, through the detailed analysis, it is clarified that the initialization and communication turned out to be inhibitive expensive for larger problems.

Table 1.Snapshot of the communication cost for iteration part

No. of Processes	PROCESS RANK	Elapsed Time (Sec)	MPI communication count	Size of Data (MB)
4	Rank0	0.946	28001	617.5
	Rank1	0.976	28001	617.5
	Rank2	0.976	28001	617.5
	Rank3	0.981	28001	617.5
8	Rank0	0.951	52001	457.3
	Rank1	0.988	52001	457.3
	Rank2	0.981	52001	457.3
	Rank3	0.981	52001	457.3
	Rank4	0.986	52001	457.3
	Rank5	0.979	52001	457.3
	Rank6	0.98	52001	457.3
	Rank7	0.981	52001	457.3

The initialization overhead in the large-scale simulation is mainly due to the wide usage of a dynamic data structure. The overhead comes from costs of automatic array reallocation with amortized allocation and handling sorted arrays with insertion for searching.

In addition to the initialization, IO that included in “others” is another bottleneck of this simulation. This is because that loading meshes (a simple list of integers) is very slow, and it takes several minutes for a file of 32MB in a serial way. Further performance optimizations for improving parallel efficiency should be considered as our future work.

Besides, for the partitioning, we make use of a space-filling curve partitioning, which allows the direct parallel reading of the mesh. This matches nicely with the octree mesh structure that we use and which is provided by the corresponding mesh-generation tool Seeder within the APES framework. However as one reviewer commented in the mid-term evaluation, we would like to consider the possibility of introducing AMR to improve the parallel performance.

5.2.3 Performance comparison among various HPC Systems

It is clear from the preceding sections that MUSUBI scales reasonably in the weak sense with a fixed problem size per node on SX-ACE. Although more detailed performance analysis and improvement SX-ACE are needed as described in the previous sub-section, we have compared the strong scaling over four HPC systems (SX-ACE@Tohoku Univ., SuperMUC, JUGENE, Hornet@HLRS).

Figure 7 indicates the performance achieved by MUSUBI in TFLOP/s against the theoretical peak performance for various fractions of the four systems. It captures the strong scaling behavior and compares the scale-up from the smallest fraction of the machine where the fixed problem size fits up to the full machine or some limit of scalability. A fixed problem size of 88 elements is chosen for this plot comparison, which fits on a single node of the machines except for JUQUEEN. Double logarithmic axes are used to capture the wide scaling range.

Whereas nearly 20% of peak performance is

achieved by MUSUBI on a single node of SX-ACE, it achieves around 4% of the peak performance on the other systems. For the scalar systems a super-linear speed-up can be observed with a large number of nodes, where the problem size per node fits into the cache.

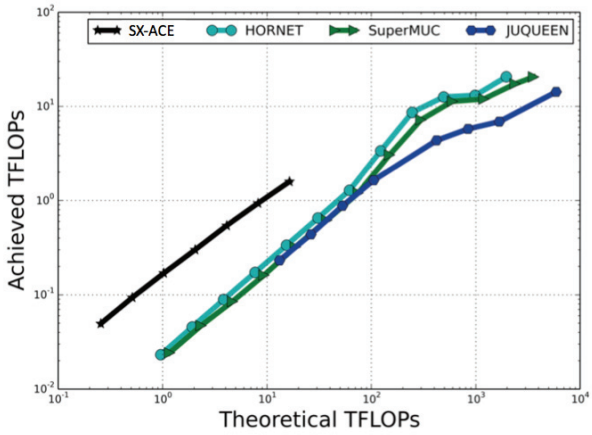

Fig.7 Perfomance Comparison among 4 HPC Systems

6. Progress of FY 2016 and Future Prospects

First, from the viewpoints of computational sciences, the resolutions in our simulations are at the order of Kolmogorov microscales to resolve all the spatial and temporal scales that appear in a transitional/turbulent flow. Our previous results have suggested that the flow remains laminar in a healthy subject whereas it transitions to a regime that exhibits high-frequency fluctuations in patients suffering from Chiari I malformation.

Figure 8 shows the turbulence intensity across axial planes in the SAS of a healthy volunteer and two Chiari patients. Unlike previous results reported in [Jain], we have increased the flow rates by 40% here to observe if the flow in healthy volunteer ever transitions to turbulence within a Physiologically realistic range of Reynolds numbers. As can be observed, the flow is evenly distributed in the volunteer; minor fluctuations develop in the Patient1 that are highly localized in the Patient2, due to impingement of flow in specific locations as a result of narrowing of the cranio-vertebral junction due to Chiari I malformation.

Therefore, we believe that the observed behavior is indeed a flow phenomenon that only appears with the malformation and is absent in healthy spinal canal geometries.

[Jaim] Kartik Jain, Transition to Turbulence in Physiological Flows: Direct Numerical Simulation of Hemodynamics in Intracranial Aneurysms and Cerebrospinal Fluid Hydrodynamics in the Spinal Canal, Ph.D Thesis, University of Siegen, 2016.

Through the detailed performance tuning for SX-ACE, and performance evaluation using several computing systems, we clarify the

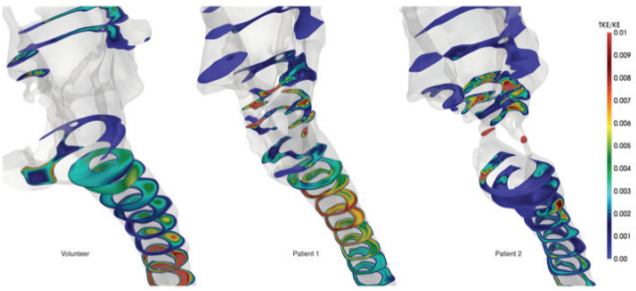

Fig. 8 Turbulence intensity across axial planes in the SAS of a healthy volunteer and two Chiari patients.

performance bottlenecks of our code on SX-ACE. Especially, initialization and communication overheads on MUSUBI turned out to be inhibitive expensive for larger problems.

In addition to these bottlenecks, IO becomes another bottleneck of this simulation. This is because that loading meshes (a simple list of integers) is very slow on SX-ACE, and it takes several minutes for a file of 32MB in a serial way. To overcome this situation, we are currently planning to replace Fortran Direct IO with MPI-IO.

This IO optimization and more comprehensive performance evaluation and analysis using

different types of supercomputers remain as our future work. We are also planning to evaluate MUSUBI using other supercomputers such as FX series, accelerator system in future.

In summary, although there are rooms for parallel performance improvements, the evaluation results show that SX-ACE provides a higher sustained performance than scalar-based systems. In particular, the evaluation results using multiple nodes indicate that the high sustained performance per core with a high sustained memory bandwidth given by powerful off-chip memory bandwidth and advanced memory subsystems enables the SX-ACE system to achieve outstanding performances that are unreachable by simply increasing the number of fine-grain scalar processor cores. We believe that this fact shows a new direction for developing a future large-scale, highly productive supercomputers.

7. List of Publications and Presentations

(1) Journal Papers

1. Ryusuke Egawa, Kazuhiko Komatsu, Shintaro Momose, Yoko Isobe, Akihiro Musa, Hiroyuki Takizawa and Hiroaki Kobayashi, "Potential of a modern vector supercomputer for practical applications: performance evaluation of SX-ACE," The Journal of Supercomputing, pp 1 - 29, doi:10.1007/s11227-017-1993-y.

2. Kartik Jain, Sabine Roller, Kent-André Mardal, Transitional flow in intracranial aneurysms – A space and time refinement study below the Kolmogorov scales using Lattice Boltzmann Method, Computers & Fluids, Volume 127, Pages 36-46, ISSN 0045-7930, https://doi.org/10.1016/j.compfluid.2016.12.011.

(2) Conference Papers

1. Jiaxing Qi, Kartik Jain, Harald Klimach, Sabine Roller, “Performance Evaluation of the LBM Solver Musubi on Various HPC Architectures,” Advances in Parallel Computing, pp.807 – 816, doi: 10.3233/978-1-61499-621-7-807

(3) Conference Presentations (Oral, Poster, etc.)

(4) Others (Patents, Press releases, books, etc.)