jh210021-MDHI

High resolution simulation of cardiac electrophysiology on realistic whole-heart geometries

Kengo Nakajima（The University of Tokyo）

Abstract

This international JHPCN project has an overall aim of enabling efficient simulations of cardiac electrophysiology on realistic whole-heart geometries, so that high-resolution and biologically-detailed simulations can help advancing the scientific understanding of the heart. The project has combined the expertise of The University of Tokyo in HPC and the expertise of Simula Research Laboratory (Norway) in cardiac modeling. Starting from an existing medium-scale simulator that used a simple numerical strategy, the project collaborators have worked on further improving the simulator in several aspects during FY2021. First, some of the basic components of the simulator have been the target for performance optimization. These include a combination of using SIMD vectorization and lookup tables for further improving the performance of the ODE part of the simulator, reordering of mesh entities for securing good data reuse in the caches, and proper delegation of the OpenMP threads of each MPI process for overlapping computation with communication (which is relevant for the PDE part of the simulator). Second, a new numerical strategy that is based on implicit time integration has been implemented. This more advanced numerical scheme is numerically more stable, and thus allows larger time steps. An AMG preconditioner has been experimented in connection with the new numerical strategy. Third, the enhanced simulator has been tested for large and huge scale simulations using many nodes on three supercomputers: Oakforest-PACS, Oakbridge-CX, and Wisteria/Odyssey. Fourth, an initial study has been performed to compare the performance obtained on Wisteria/Odyssey with that obtained on Oakforest-PACS and Oakbridge-CX. This aspect has the purpose of identifying new programming and optimization challenges/strategies that are specific for Wisteria/Odyssey.

1. Basic Information

(1) Collaborating JHPCN Centers

Tokyo

(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

Kengo Nakajima (U Tokyo): Project administration, numerical algorithms and HPC.

Xing Cai (Simula/Norway): Numerical algorithms, HPC & project coordination with Prof. Nakajima.

Akihiro Ida (JAMSTEC): Numerical algorithms and parallel programming.

Toshihiro Hanawa (U Tokyo): Code parallelization, profiling and optimization.

Masatoshi Kawai (U Tokyo): Numerical algorithms and parallel programming.

Tetsuya Hoshino (U Tokyo): Code parallelization, profiling and optimization.

Masaharu Matsumoto (Fukushima Univ): Numerical algorithms and parallel programming.

Glenn Terje Lines (Simula/Norway): Cardiac electrophysiology, mathematical modeling.

Johannes Langguth (Simula/Norway): Parallel programming and code optimization.

Jonas van den Brink (Simula/Norway): Cardiac electrophysiology, mathematical modeling, and simulations.

Kristian Gregorius Hustad (Simula/Norway): Parallel programming, code optimization, and simulations.

Hermenegild Arevalo (Simula/Norway):

Cardiac electrophysiology, mathematical modeling and simulations.

James Trotter (Simula/Norway): Parallel programming and code optimization.

Aadarsh Bussooa (Simula/Norway): Parallel programming and cardiac simulations.

María Hernández Mesa (Simula/Norway): Cardiac electrophysiology and numerical simulations.

Lena Myklebust (Simula/Norway): Cardiac electrophysiology and numerical simulations.

Lisa Pankewitz (Simula/Norway): Cardiac electrophysiology and numerical simulations.

2. Purpose and Significance of Research

Coordinated electrical activities are vital for the heart to function normally and effectively. Here, many questions about cardiac electrophysiology are not yet answered. We thus need simulation codes that can effectively use modern supercomputers for “in-silico” experiments. This project aims to extend the existing expertise of the project partners on very large-scale simulations over simple computational meshes, into the regime of unstructured meshes. Successful results from this project will help to advance the field of cardiac electrophysiology. Moreover, experience obtained on effectively using the latest processor architectures are applicable to other scenarios of computation over unstructured meshes. The goal of FY2021 has four main aspects. 1. To further improve the scale-out capability (i.e., efficiently using many nodes of a supercomputer) of an existing cardiac simulator based on explicit time integration. 2. To develop an enhanced simulator based on implicit time integration, for better numerical stability. 3. To port the codes to the Wisteria system, which uses the Fujitsu A64FX architecture, for making the codes more versatile and gathering new performance optimization experience. 4. To run realistic cardiac simulations using the optimized and enhanced simulators to investigate some hypotheses in cardiac electrophysiology. Oakforest-PACS (OFP), Oakbridge-CX (OBCX), and Wisteria/BDEC-01 (Odyssey) at the University of Tokyo are used.

3. Significance as JHPCN Joint Research Project

The significance of this JHPCN joint research project is due to two aspects. First, UTokyo has world-leading expertise in implementing and optimizing advanced numerical code. This expertise has been built up via developing real-world applications for running on cutting-edge supercomputers at UTokyo. Such hands-on experience on supercomputing is lacking for the Norwegian partner whose main expertise is on cardiac modeling and simulation. Second, the Oakforest-PACS and Oakbridge-CX systems, as well as the upcoming Wisteria/BDEC-01 system, are of a suitable size for achieving the ambitious goal of this project, whereas access to world-leading supercomputers has been very scarce for the Norwegian partner. The chosen hardware systems, in particular Wisteria/BDEC-01, also provide realistic and future-oriented testbeds for the performance portability of the simulation codes to be developed.

4. Outline of Research Achievements up to FY2020

This project is a continuation of project jh200036. During FY2020, SIMD vectorization of the simulator was automatically enabled by the compiler, with the help of suitable OpenMP directives. Another approach was tested to use the VCL library, which offers a portable way of programming with SIMD intrinsics. It was found that the automated compiler-supported approach was to be preferred. In addition, OpenMP

programming was combined with MPI programming to achieve a hybrid parallelization, which was seen to produce better performance than a pure MPI approach. Moreover, some initial tests of running the simulator on multiple nodes of Oakbridge-CX (strong scaling) were done. Work was also started on testing a multigrid preconditioner, together with several choices of the storage formats for the sparse matrix.

5. Details of FY2021 Research Achievements

(1) Optimization of the Existing Explicit-Method based Simulator

Work was continued in FY2021 to further improve the existing simulator that is based on an explicit time integration scheme. We remark that the computational work per time step consists of two parts. The first part is to solve a nonlinear ODE system at each mesh element. The second "PDE" part is a sparse matrix-vector multiplication (SpMV) that explicitly integrates a 3D diffusion equation by one time step.

Combining SIMD vectorization with lookup tables. Since the ODE system involves quite a few evaluations of special mathematical functions, such as exp(), log() and pow(), a strategy of further speeding up the ODE solver is to pre-evaluate these special mathematical functions, or related expressions, and store them in tabular forms. Use of such lookup tables is to avoid repeated evaluations of expensive mathematical functions, at a slight reduction of accuracy (but still at an acceptable level). The challenge is that automated compiler vectorization will cease to work, when the lookup tables are inserted. We have found out that the following code restructuring is needed. Namely, the computational work of the ODE solver has to be split into two parts, one involving lookup tables, the other without. Then, the part without lookup tables can be auto-vectorized by the compiler (with the help of suitable OpenMP directives). Thus, both SIMD vectorization and lookup tables (LUT) can contribute to performance improvement. Table 1 shows such an experiment and more details can be found in [2].

Table 1 The performance of the ODE solver (measured in million cell-steps per second) when using a single node on Oakforest-PACS (OFP) or Oakbridge-CX (OBCX)

	Naive	SIMD	SIMD+LUT
OFP	46.9	354.3	423.5
OBCX	96.0	374.3	552.4

Re-ordering of mesh entities.

Mixed MPI-OpenMP programming was studied for the PDE part of the simulator, which executes a parallel SpMV per time step during the explicit time integration. The work per SpMV consists of a "boundary part" (computing the values that are needed by neighboring MPI processes) and an "interior part". The boundary part can be expensive due to more irregular memory accesses. A remedy has been devised as an appropriate re-ordering of the boundary part to improve cache utilization (Fig.1). For an extreme case of using 272 OpenMP threads per Oakforest-PACS node, we observed that boundary-reordering can decrease the SpMV time from 16 ms to 6.3 ms.

Computation Communication (CC) Overlapping.

Another effort was on improving communication. Especially, we are investigating the effects of dynamic loop scheduling, where one OpenMP thread is used for MPI communication, while the other threads can immediately start computing the interior part. This strategy (CC-Overlapping with Dynamic Loop Scheduling (DLS)) is to allow the communication thread to join the computation

when the communication is done. Dynamic thread scheduling has been found to be very important.

We evaluated the performance of GeoFEM/Cube with CC-Overlapping on Oakforest-PACS (OFP) and Wisteria/BDEC-01 (Odyssey) (ODY). GeoFEM/Cube solves 3D linear elasticity problems in simple cube geometries using a parallel finite element method (FEM). Tri-linear hexahedral elements are used for the discretization. Material properties are defined as homogeneous, where Poisson's ratio is set to 0.30 for all elements and Young's modulus is 1.00. The boundary conditions are described in Fig.2.

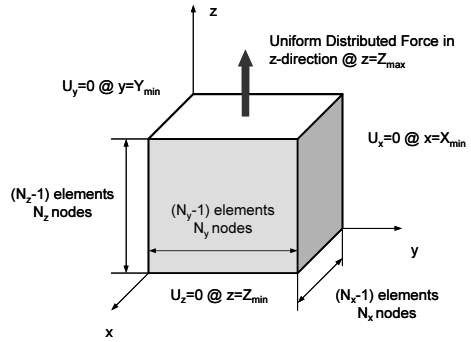

Fig.2 Simple Cube Geometry for 3D Linear Elasticity Problems in GeoFEM/Cube [6]

Derived linear equations with sparse matrices are solved by preconditioned Conjugate Gradient (CG) method. The preconditioner by block diagonal LU factorization utilizes the mathematical and physical properties of GeoFEM/Cube. Block diagonal LU factorization preconditioning applies full LU factorization to each 3x3 diagonal block of the coefficient matrix and applies forward/backward substitution to each block at the preconditioning stage. This preconditioning is easy to parallelize and more robust than Point-Jacobi preconditioning.

We evaluated the performance of GeoFEM/Cube using up to 2,048 nodes of OFP (131,072 cores) and ODY (98,304 cores). Because the most expensive procedure is SpMV, this evaluation is useful for optimization of methods based on explicit time marching scheme. Hybrid MxN (HB MxN) parallel programming models were applied, where "M" denotes the number of OpenMP threads for each MPI process, and "N" is the number of MPI processes on each node. In the present work, HB 16x4 for OFP, and HB 12x4 for ODY were evaluated. For each case, the following three implementation were evaluated:

- Original: Original code without any CC-Overlapping. Local computation of SpMV starts after completion of a halo exchange.
- Static: CC-Overlapping with static loop scheduling (Fig.2).
- Dynamic: CC-Overlapping with dynamic loop scheduling (Fig.3). The chunk size was varied from 100 to 1,000

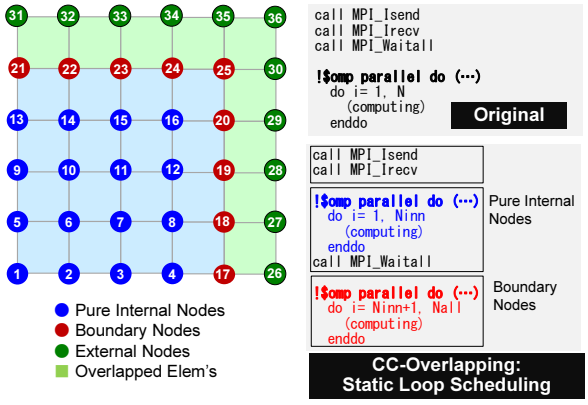

Fig.2 Overlapping communications in halo exchanges with SpMV computation (Ninn = number of pure internal nodes/meshes; Nall = total number of internal nodes/meshes) [6]

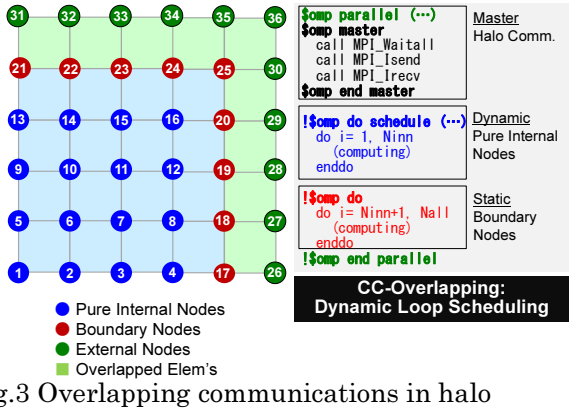

Fig.3 Overlapping communications in halo exchanges with SpMV computations using dynamic loop scheduling by OpenMP [6]

Fig.2 describes communication pattern of *Original* implementation and that with CC-Overlapping. This type of CC-Overlapping is called CC-Overlapping with *Static* Loop Scheduling. Furthermore, dynamic loop scheduling of OpenMP was introduced for CC-Overlapping in halo exchanges based on the method in [6]. In this approach, halo exchanges, including transfers of buffer copies, are done by the master thread, and dynamic loop scheduling is applied to the computations for pure internal nodes, as shown in Fig.3. The computations for pure internal nodes (blue letters in Fig.3) start without the master thread, which performs communication (red letters in Fig.3). The master thread can join the computations for pure internal nodes after completing the communication.

Measurements were repeated five times, and the fastest one of the five measurements are presented as the results for each case. Performance was evaluated at computation time for each iteration of CG solver. Following three types of problem size were considered. The largest problem size at 2,048 nodes is 2.4576x1010 DOF:

- Small (S): 96x96x48 FEM nodes/node (1,327,104 DOF/node)
- Medium (M): 128x128x64 FEM nodes/node (3,145,728 DOF/node)
- Large (L): 200x200x100 FEM nodes/node (12,000,000 DOF/node)

Fig. 4(a) and 4(b) show improvement of performance over "original" implementation in Figure 7 at 2,048 nodes for each of OFP and Odyssey (ODY). Generally, performance improvement is 10-15% at OFP, and 25-30% at Odyssey. In both of OFP and ODY, this improvement is significant for Medium and Small cases. Effects of dynamic loop scheduling is rather smaller for Large cases, because effects of communication overhead by halo exchange are smaller. This feature is more significant in ODY. Optimum chunk size for dynamic loop scheduling is 700 for Medium and Small, and is 1,000 for Large cases on both systems. Improvement of performance in static implementation is rather smaller on OFP, but it is very large on ODY, and this is very significant in Large cases.

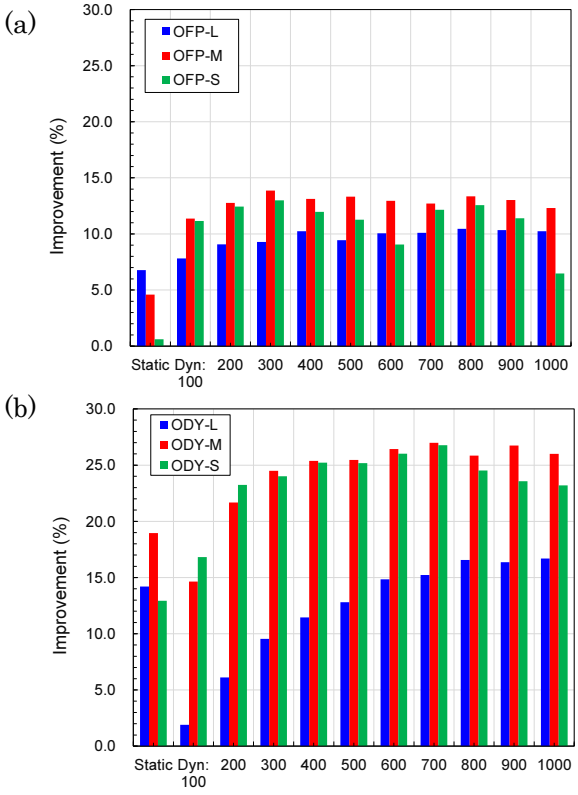

Fig.4 Improvement by Static and Dynamic (Dyn) CC-Overlapping over "Original" implementations at 2,048 nodes (a) Oakforest-PACS (OFP), (b) Wisteria/BDEC-01 (Odyssey) [6]

Fig.5(a) and 5(b) show the best case of performance improvement over Original implementation for each number of nodes of OFP and Odyssey (ODY). Generally, results show similar features in Fig.4(a) and 4(b), but it is clear that performance improvement is decreasing in Medium and Small cases for OFP, as number of node increases. Similar features are also observed

in Fig.5(b) for ODY, although decreasing is very slight. This is mainly because of overhead by MPI_Allreduce for dot products in CG solver.

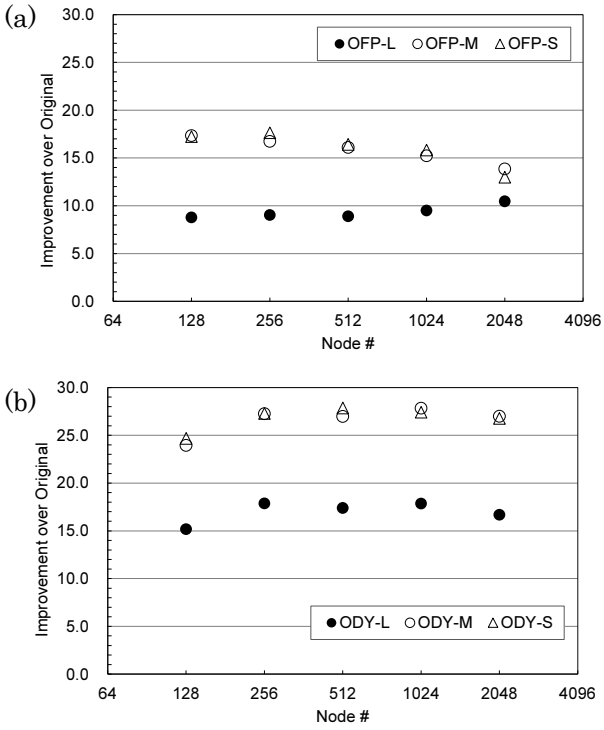

Fig.5 Improvement by Static and Dynamic CC-Overlapping over "Original", best case for each number of nodes (a) OFP, (b) Odyssey

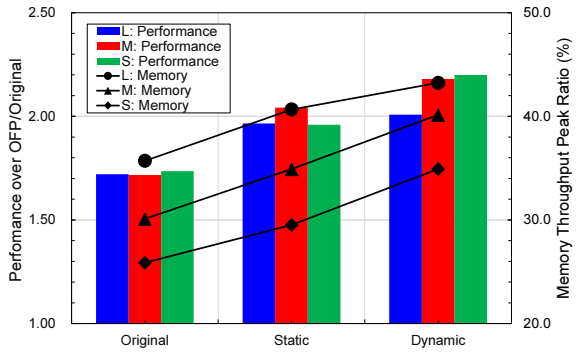

Fig.6 Performance Analyses on Odyssey using Detailed Profiler, Bars: Relative Performance compared to Original implementation of OFP, Lines & Symbols: Memory Throughput Peak Ratio

Moreover, Fig.6 compares results of Original, Static, and Best Case for Dynamic at 2,048 nodes of ODY. Three bars are relative performance compared to the Original implementation of OFP, and lines/symbols show memory throughput peak ratio (%) measured by Fujitsu's detailed profiler. Both of performance and memory throughput are gradually improving from *Original* through *Static* to *Dynamic*.

(2) Development & Optimization of the Implicit-Method based Simulator

We have implemented a preliminary version of a new simulator based on an implicit time integration scheme. Here, a linear system involving a sparse matrix needs to be solved by a Krylov iterative solver per time step. This version allows using larger time steps, but its overall computational efficiency hinges upon the effectiveness of a parallel preconditioner. For this purpose, we have investigated a parallel algebraic multigrid (AMG) preconditioner, including many of its internal parameters and strategies, for example, matrices that arise from discretizing the PDE part using a cell-centered finite volume scheme over unstructured tetrahedral meshes.

AMG is use a characteristics that a high-frequency error is easy to dump, but a low-frequency error is hard to reduce. To reduce low-frequency error, we map the low-frequency error to coaser grid space. Here, we can reduce the low-frequency error on the coarse grids with lower computational cost. It is because the number of elements of coarse grid is less than the fine grid, and the low-frequency error on the fine grid is seen as a higher frequency error on coarser grid. Therefore, the performance and effectiveness of AMG depend on the construction of multi-level coarse grid spaces (coarse grid collection). To construct the coarse grid collection, we focus on the relationships between diagonal and off-diagonal elements in a given matrix. If the off-diagonal elements are enough larger than the diagonal element, there are handled as "strong

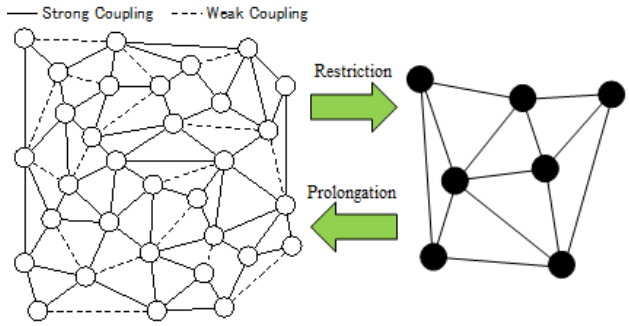

Fig.7 Example of how to construct coarse grid space from a given grid space

coupling". Then, elements with the strong-couplings map to an element on the coarse grid (Figure.7). Deciding a threshold of strong coupling is often difficult, and it depends on the problems. In addition, the best threshold on the SLEs derived from the PDE part depends on time step size. It is because the diagonal elements are changed by the time step size. Therefore, it is required a scheme to decide the threshold, dynamically. In this study, we investigated a method to dynamically determine the threshold by focusing on the properties of the matrix. In detail, we check all the couplings initially and introduce a parameter α that determines the threshold to make arbitrary ratio couplings strong-coupling.

Table2 shows the result of our approach. To check the efficiency of the AMG method, we compared the number of iterations between BiCGSTAB and a CG method with the AMG preconditioner(AMG-CG). In the numerical evaluations, we decide the ratio $\alpha = 30\%$. On the numerical evaluations, AMG-CG shows better convergence than BiCGSTAB and the differential becomes larger with a larger time step size(dt). This differential will also become larger with larger DOF.

Table 2 Number of iterations compared with BiCGSTAB and AMG-CG

Problem name	Time step size	Number of iterations	
	dt	BiCGSTAB	AMG-CG
Patient_1	1	14	7
	10	51	9
	100	183	12
Patient_20	1	14	7
	10	49	9
	100	163	11
Patient_Pat01	1	16	7
	10	51	9
	100	199	11
Cuboid_500	1	73	11
	10	7	6
	100	25	9
Cuboid_200	1	19	8
	10	58	11
	100	192	15
Cuboid_100	1	381	21
	10	40	9
	100	105	13

(3) Large-scale Simulations

Performance of the improved simulator (explicit-time-integration) was studied on Oakforest-PACS, Oakbridge-CX and Wisteria/Odyssey. An unstructured computational mesh containing 55 million tetrahedra was used for this strong-scaling study. The following tables show the average time usage [ms] per time step (and the time breakdown) on the three systems. It can be seen that Oakbridge-CX is the most powerful system. The performance of Wisteria/Odyssey is better than Oakforest-PACS, but needs further study. Moreover, simulation results obtained on Oakforest-PACS and Oakbridge-CX were included in two journal submissions that are currently under review.

Table 3 Average time usage and breakdown measured on Oakforest-PACS (#cores = #nodes x 4 MPI x 17 threads)

#nodes	#cores	Total	PDE	ODE	MPI
32	2176	222.1	26.6	188.7	6.8
64	4352	110.3	10.8	94.5	5.0
128	8704	55.1	4.2	47.3	3.6
256	17408	28.4	1.4	23.7	3.4
512	34816	15.2	0.5	11.9	2.9
1024	69632	9.4	0.3	6.0	3.1
2048	139264	6.4	0.6	3.0	2.7

Table 4 Average time usage and breakdown measured on Oakbridge-CX (#cores = #nodes x 2 MPI x 28 threads)

#nodes	#cores	Total	PDE	ODE	MPI
4	224	48.0	17.90	26.06	3.99
8	448	25.6	9.82	13.39	2.40
16	896	13.6	5.01	6.95	1.66
32	1792	7.2	2.58	3.48	1.17
64	3584	3.8	1.36	1.71	0.69
128	7168	2.3	0.56	0.86	0.88
256	14336	1.1	0.14	0.43	0.51

Table 5 Average time usage and breakdown measured on Wisteria/Odyssey (#cores = #nodes x 4 MPI x 12 threads)

#nodes	#cores	Total	PDE	ODE	MPI
32	1536	48.1	1.99	43.84	2.29
64	3072	25.9	0.96	21.79	3.12
128	6144	14.1	0.45	10.78	2.82
256	12288	6.8	0.22	5.40	1.14
512	24576	3.7	0.10	2.70	0.91
1024	49152	1.8	0.05	1.35	0.41
2048	98304	1.1	0.03	0.68	0.39

(4) Porting to Wisteria/BDEC-01

We have made an initial porting of the explicit simulator to the Wisteria/BDEC-01 (Odyssey) system. Both the ODE part and PDE part were verified to produce the correct numerical results. A separate test concerning the "CC-Overlapping with DLS" (Fig.1) approach was also successful, and 20+% performance improvement has been achieved for a sample benchmark with 3D FEM application using up to 6,144 nodes (294,912 cores) of Odyssey (Fig.2). The achievable performance will need to be examined with more details.

6. Progress during FY2021 and Future Prospects

Compared with the project plan made at the beginning of FY2021, it can be said that the research activities have been carried out accordingly during FY2021. In particular, the work of FY2021 also covered one task (single-node optimization of the explicit-time integration based simulator) that should have been finished during FY2020, but was delayed to the pandemic.

For the upcoming work during FY2022, the focus will be given to the tuning and incorporation of the parallel AMG preconditioner into the new simulator that is based on implicit time integration. Further performance optimization and tuning will also be applied to the overlap of computation with communication, vectorization of the PDE part, and use of mixed precision in the calculations. Adoption of GPUs (on the Aquarius system) will also be investigated.

7. List of Publications and Presentations

(1) Journal Papers (Refereed)

[1] Trotter D. J.(+), Cai X.(+), Funke S. W., On memory traffic and optimisations for low-order finite element assembly algorithms on multi-core CPUs, ACM Transactions on Mathematical Software, https://doi.org/10.1145/3503925, 2021

[2] Hustad K. G.(+), Cai X.(+),Resource-efficient use of modern processor architectures for numerically solving cardiac ionic cell models.

Frontiers in Physiology, accepted for publication, 2022

(2) Proceedings of International Conferences (Refereed)

[3] Nakajima K., Ogita T., Kawai, M., Efficient Parallel Multigrid Methods on Manycore Clusters with Double/Single Precision Computing, IEEE Proceedings of iWAPT 2021 in conjunction with IPDPS 2021, May 2021

[4] Yoda R., Bolten M.(+), Nakajima K., Fujii A., Assignment of idle processors to spatial redistributed domains on coarse levels in multigrid reduction in time, The International Conference on High Performance Computing in Asia-Pacific Region (HPC Asia 2022), 2022

[5] Kawai M., Nakajima K., Low/Adaptive Precision Computation in Preconditioned Iterative Solvers for Ill-Conditioned Problems, The International Conference on High Performance Computing in Asia-Pacific Region (HPC Asia 2022), 2022

[6] Nakajima, K., Gerofi, B., Horikoshi, M., Ishikawa, Y., Communication-Computation Overlapping for Preconditioned Iterative Solvers by Dynamic Loop Scheduling, ACM Proceedings of IWAHPCE22 Workshop in conjunction with HPC Asia 2022, 2022

(3) International conference Papers (Non-refereed)

(4) Presentations at domestic conference (Non-refereed)

[7] 河合直聡，中島研吾， 低精度浮動小数点数を適用した ICCG 法の性能評価，情報処理学会研究報告（2021-HPC-180-23），2021

[8] 中島研吾，河合直聡，Wisteria/BDEC-01(Odyssey)における大規模前処理付き反復法ソルバーの性能評価，日本応用数理学会 「行列・固有値問題の解法とその応用」研究部会（MEPA），2021 年並列／分散／協調処理に関するサマー・ワークショップ (SWoPP2021)，2021 年７月

[9] 中島研吾，河合直聡，Wisteria/BDEC-01(Odyssey)における並列多重格子法ソルバーの開発と性能評価，日本応用数理学会年会 2021，先進的環境における数値計算と関連 HPC 技術，2021 年 9 月

(5) Published library and relating data

(6) Other (patents, press releases, books and so on)