jh190040

Physiologically realistic study of subcellular calcium dynamics with nanometer resolution

Kengo Nakajima（The University of Tokyo）

Abstract
This project has the aim of combining advanced mathematical modeling and very large-scale numerical simulations for studying subcellular calcium dynamics, which are of vital importance for the function of the heart. By incorporating realistic geometries of the calcium release units and their microanatomical structures, the numerical experimentation that is enabled by this joint Japan-Norway JHPCN project can improve the realism of state-of-the-art mathematical models of subcellular calcium signaling, thereby promoting detailed studies of how disease-driven structural changes affect the excitation-contraction coupling in the heart. At the same time, the associated challenge of computational capacity and efficiency is addressed by algorithmic improvements and optimization of the involved parallel software on various coding levels. The obtained results with respect to cardiac electrophysiology include a quantitative understanding of the impact of sarcoplasmic reticulum load on the process of calcium induced calcium release, validating the mathematical model adopted. Moreover, an old parallel simulator which existed prior to this project has undergone a substantial restructuring and improvement process, including reduction of memory footprint and traffic, explicit code vectorization, optimization of MPI communication and mixed OpenMP-MPI programming. Finally, effects of pipelined CG methods were evaluated on the Oakforest-PACS system.

1. Basic Information

(1) Collaborating JHPCN Centers

Information Technology Center, Univ. 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 parallel programming.
- Xing Cai (Simula/Norway): Numerical algorithms, code parallelization and optimization, as well as project coordination together with Prof. Nakajima.
- Glenn Terje Lines (Simula/Norway): Cardiac electrophysiology, mathematical modeling and running simulations.
- Akihiro Ida (U Tokyo): 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.
- Chad Jarvis (Simula/Norway): Code parallelization, profiling and optimization, as well as running simulations.
- Johannes Langguth (Simula/Norway): Code parallelization, profiling and optimization.
- Jonas van den Brink (Simula/Norway): Preparation of geometries and physiological parameters for subcellular simulations.

2. Purpose and Significance of Research

With respect to HPC, this project aims to greatly improve an experimental subcellular simulator previously developed by the Norwegian partner (Simula). The old simulator was inefficient due to inefficient data structures, performance unfriendly loop nests and suboptimal parallelization. It thus urgently needed code restructuring and optimization. With an efficient simulator (developed in this project) to execute very large-scale simulations, this project will consolidate a multi-scale mathematical model that gives a physiologically accurate description of healthy and pathological calcium releases, thereby advancing the scientific understanding of subcellular calcium dynamics.

For this FY, the work related to further optimizing the subcellular simulator will also produce new knowledge about efficiently coding and parallelizing multiple inter-tangled stencil computations for the Knights Landing and Skylake architectures. Consequently, large-scale simulations of subcellular calcium handling, using realistic geometries and distributions of calcium release units, can be carried out to further validate the mathematical model, as well as quantifying the impact of disease-driven structural changes on the excitation-contraction coupling in the heart. Moreover, experience will be sought on using high-speed file cache systems for in-situ data analytics.

3. Significance as JHPCN Joint Research Project

The significance of this JHPCN joint research project has two aspects. First, UTokyo has world-leading expertise in implementing and optimizing advanced numerical code for real-world applications to run on cutting-edge supercomputers. Such hands-on experience in supercomputing is lacking for the Norwegian partner. Second, the Oakforest-PACS and Oakbridge-CX systems are of a suitable size for achieving the ambitious goal of this project, whereas access to world-leading supercomputers has traditionally been very scarce for the Norwegian partner. The high-speed file cache systems available at UTokyo also provide new possibilities of in-situ huge-scale data analysis.

4. Outline of Research Achievements up to FY2018

OpenMP parallelization

The existing old simulator had only MPI parallelization implemented. The lack of shared memory-based parallelization (via threads) is a clear obstacle to fully utilizing the Oakforest-PACS and Oakbridge-CX systems. Our first code improvement was thus focused on incorporating OpenMP parallelization into the various computational tasks. We also painstakingly removed pitfalls such as race conditions and false sharing. The largest benefit of the OpenMP version is avoiding the overhead due to explicit inter-process MPI communication. For example, for a 3D solution domain of size $(2\mu m)^3$, using a 168x168x168 computational mesh, the OpenMP-enabled version needs 64.05 seconds (256 threads on one Knights Landing node), in comparison with 79.70 seconds needed by the old pure-MPI version (256 MPI processes).

Reducing memory footprint

The main computational kernel for simulating subcellular calcium signaling is a set of 3D diffusion-reaction equations. To model the realistic compartments and geometries inside the calcium release units, different diffusion constants are valid in different physiological regions. A simple strategy, which was used by the old simulator, is to adopt a variable diffusion coefficient throughout the domain. This strategy is easy to code but has an extensive memory overhead in that each diffusive species will need three additional 3D arrays. In order to reduce the memory footprint, a lookup table approach was adopted. The lookup table, which is of a tiny size, can be used to accurately incorporate the flux/no-flux interactions between each pair of neighboring computational voxels, therefore completely eliminating the additional 3D arrays. Take for instance the mesh of size 168x168x168. Four diffusive species would have needed 4x3x168x168x168x8=455MB (in double precision) for extra memory usage, which can now be saved.

Explicit code vectorization with AVX-512

The "lookup table" approach, however, requires elaborate coding. It induces more instructions to be carried out on the processor level. To offset this potential performance disadvantage, we explicitly vectorized the diffusion and reaction computations using AVX-512 intrinsics. (This is because experiments show that compiler-enabled auto vectorization is not sufficiently effective.) For example, on a 672x672x168 mesh, using 256 MPI processes, the time needed by the manually vectorized "lookup table" version is 26.1 seconds (for 100 time steps), versus 41.7 seconds needed by the manually vectorized "coefficient array" counterpart.

Quantitative study of the impact of SR load

Many factors can affect the calcium-induced calcium release process, including geometrical configuration, RyR sensitivity and initial SR concentration. As an example, we investigated the impact of varying the SR concentration (also called SR load). Thanks to the new simulator and the Oakforest-PACS system, we were able to carry out thousands of simulations needed to produce a quantitative analysis about how cells behave differently under different SR loads. These simulations are necessary for validating the mathematical model.

Simulations involving many CRUs

We also studied the impact due to disease-driven changes of the microanatomical structures inside the cardiac cells. For example, chronic heart failure can lead to a disintegration of the calcium release units (CRUs), in terms of a change in the number, distribution and calcium release channels. For this research topic, it is very important to adopt simulations that involve many CRUs with realistic geometries and microanatomical details. Simulations involving 93 CRUs that were done on Oakforest-PACS showed that a calcium wave can indeed occur under pathological conditions.

5. Details of FY2019 Research Achievements

Overview

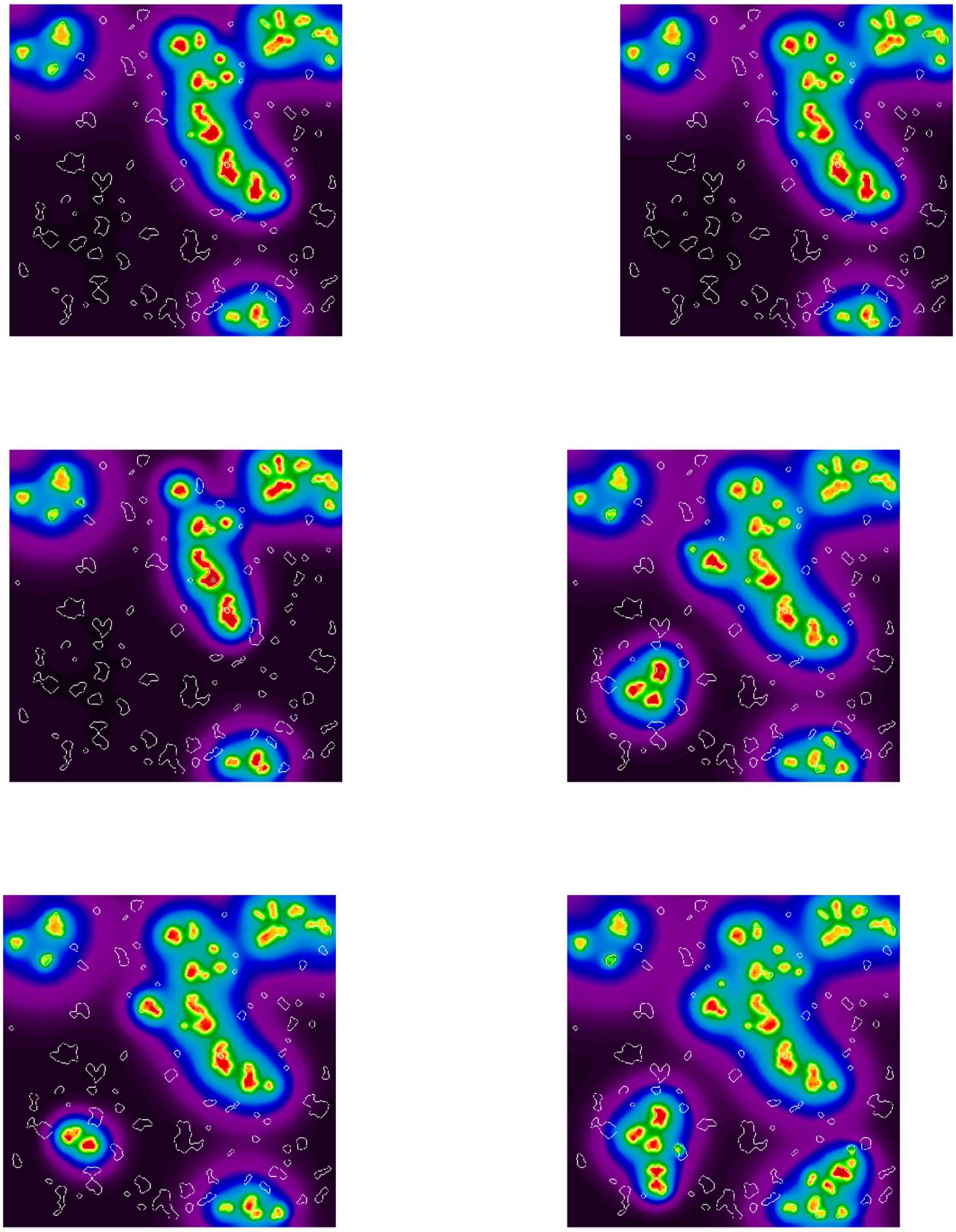
New efforts have been invested in FY2019 to further optimize the parallel simulator of subcellular calcium handling. The concrete topics that have been studied include

- Reducing OpenMP overhead;
- Appropriate choice of number of OpenMP threads per MPI process.
- Improving overlap of MPI communication with computation;
- Preliminary Investigation towards Implicit Time-Marching Scheme

Also, in preparation for running simulations on the Oakbridge-CX system, the code optimization work done in FY2018 has been tuned and measured on a 4-socket server with Xeon Skylake Gold 16-core CPUs, which is available at Simula. Table 1 confirms that the code optimization solutions (e.g. using AVX-512 intrinsics) adopted for the Oakforest-PACS system also work on the Xeon architecture.

Table 1: Time measurements (obtained on a 4-socket Skylake server) of five MPI implementations of the subcellular simulator; computational mesh: 672 × 672 × 168, time steps: 1000

Version	CA_auto		LUT_auto		CA_man		LUT_man1		LUT_man2	
MPI procs	T_R	T_D	T_R	T_D	T_R	T_D	T_R	T_D	T_R	T_D
$2 \times 2 \times 2 = 8$	41.0	164.6	40.7	166.3	17.7	131.0	17.7	85.2	17.8	63.2
$4 \times 2 \times 2 = 16$	20.5	83.8	20.4	84.6	10.0	67.9	10.1	43.9	10.1	33.5
$4 \times 4 \times 2 = 32$	10.4	49.5	10.4	45.2	7.5	45.2	7.5	27.9	7.5	23.2
$4 \times 4 \times 4 = 64$	6.1	51.3	6.2	38.8	5.8	50.9	6.1	36.8	6.1	28.1
$8 \times 4 \times 4 = 128$	6.4	52.5	6.8	37.8	6.2	53.5	6.7	35.6	6.6	32.4



In Table 1, CA denotes the original “coefficient array” approach, whereas LUT denotes the new “lookup table” approach. The subscripts auto and man denote, respectively, compiler-enable vectorization and manual vectorization using AVX-512 intrinsics. Moreover, some snapshots of a multi-CRU simulation are included.

Krylov Iterative Solvers

Because we are solving very nonlinear phenomena, we adopted fully explicit time-marching as the scheme for integration in time direction. Generally, explicit time-marching requires huge number of time steps due to constraint of CFL conditions. We are also considering implicit time-marching scheme where longer time steps are allowed. We need to solve large scale linear equations if we adopt implicit time-marching scheme. In FY.2019, we evaluated performance of preconditioned Krylov Iterative solvers on the Oakforest-PACS (OFP) system. In the present work, we solved large-scale linear equations with sparse coefficient matrices derived from 3D FEM applications for static-linear solid mechanics [5]. Linear equations are solved by Conjugate Gradient method (CG) preconditioned by SGS (Symmetric Gauss-Seidel).

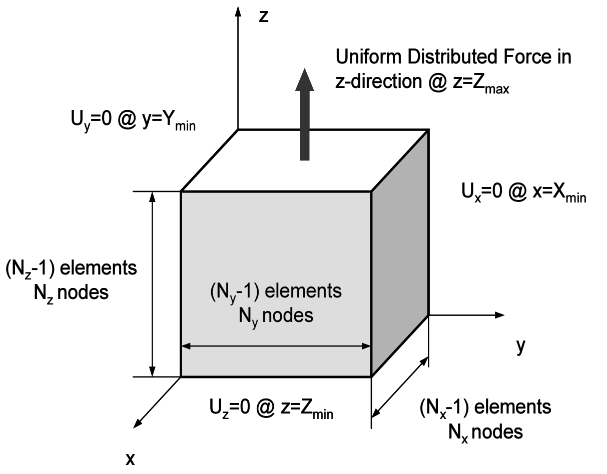
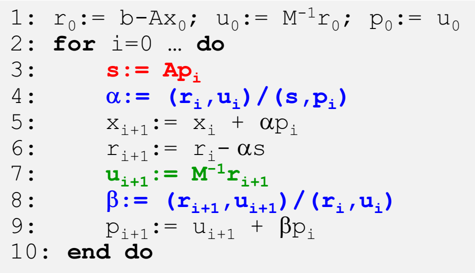


Figure 1: 3D FEM Appl. for Solid Mechanics [5]

Scalability of Krylov subspace methods, such as CG method, suffers from costly global synchronization steps that arise in dot-products and norm calculations on parallel machines. In this work, a modified preconditioned CG method is presented that removes the costly global synchronization steps from the standard CG algorithm by only performing a single non-blocking reduction per iteration based on the work by P. Ghysels et al in 2014 [P. Ghysels, W. Vanroose, Hiding global synchronization latency in the preconditioned Conjugate Gradient algorithm, Parallel Computing 40-7, 224-238, 2014 [Ghysels 2014]]. This global communication phase can be overlapped by the matrix–vector product, which typically only requires local communication. The resulting algorithm will be referred to as pipelined CG. These methods are compared to a recently proposed asynchronous CG algorithm by Gropp [Ghysels 2014].



```
1:  $r_0 := b - Ax_0$; $u_0 := M^{-1}r_0$; $p_0 := u_0$
2:  for i=0 … do
3:      $s := Ap_i$
4:      $\alpha := (r_i, u_i)/(s, p_i)$
5:      $x_{i+1} := x_i + \alpha p_i$
6:      $r_{i+1} := r_i - \alpha s$
7:      $u_{i+1} := M^{-1} r_{i+1}$
8:      $\beta := (r_{i+1}, u_{i+1})/(r_i, u_i)$
9:      $p_{i+1} := u_{i+1} + \beta p_i$
10: end do
```

Figure 2: Algorithm of Preconditioned CG Method (PCG) [Ghysels 2014]

In the original CG, results of dot-products are used just after they are calculated, as shown in lines 4-5 (α), and lines 8-9 (β) in Figure 2. Therefore, this algorithm is significantly affected by the communication overhead in the dot-products using MPI_Allreduce on massively parallel supercomputers.

In [Ghysels 2014], they proposed Pipelined CG method (Fig.3), where procedures of expensive computing (e.g. SpMV, Preconditioning) are inserted just after dot-products. In Fig.3, dot-products are calculated on line-3 and line-4, and before the results are used in line-7 and line-8, preconditioning (line-5) and SpMV (line-6) are conducted. In [Ghysels 2014], they introduced MPI_Iallreduce for Asynchronous Collective Communication which is supported in MPI 3 or later, instead of MPI_Allreduce. Thus, communications by MPI_Iallreduce are overlapped with computation on line-5 and line-6 in Figure 3. Figure 4 describes Gropp's algorithm [Ghysels 2014], where asynchronous collective communication could overlap communications and computations for line 3/4 and line 9/10.

```
1: $r_0 := b-Ax_0;\ u_0 := M^{-1}r_0;\ w_0 := Au_0$
2: for i=0 … do
3:     $\gamma_i := (r_i, u_i)$
4:     $\delta := (w_i, u_i)$
5:     $m_i := M^{-1}w_i$
6:     $n_i := Am_i$
7:     if i>0 then
8:     $\beta_i := \gamma_i / \gamma_{i-1};\ \alpha_i := \gamma_i / (\delta - \beta_i \gamma_i / \alpha_{i-1})$
9:     else
10:    $\beta_i := 0;\ \alpha_i := \gamma_i / \delta$
11:    end if
12:    $z_i := n_i + \beta_i z_{i-1}$
13:    $q_i := m_i + \beta_i q_{i-1}$
14:    $s_i := w_i + \beta_i s_{i-1}$
15:    $p_i := u_i + \beta_i p_{i-1}$
16:    $x_{i+1} := x_i + \alpha_i p_i$
17:    $r_{i+1} := r_i - \alpha_i s_i$
18:    $u_{i+1} := u_i - \alpha_i q_i$
19:    $w_{i+1} := w_i - \alpha_i z_i$
20: end for
```

Figure 3: Pipelined CG [Ghysels 2014]

```
1: $r_0 := b-Ax_0;\ u_0 := M^{-1}r_0;\ p_0 := u_0;\ s_0 := Ap_0;$
$\gamma_0 := (r_0, u_0)$
2: for i=0 … do
3:     $\delta := (p_i, s_i)$
4:     $q_i := M^{-1}s_i$
5:     $\alpha_i := \gamma_i / \delta$
6:     $x_{i+1} := x_i + \alpha_i p_i$
7:     $r_{i+1} := r_i - \alpha_i s_i$
8:     $u_{i+1} := u_i - \alpha_i q_i$
9:     $\gamma_{i+1} := (r_{i+1}, u_{i+1})$
10:    $w_{i+1} := Au_{i+1}$
11:    $\beta_{i+1} := \gamma_{i+1} / \gamma_i$
12:    $p_{i+1} := u_{i+1} + \beta_{i+1} p_i$
13:    $s_{i+1} := w_{i+1} + \beta_{i+1} s_i$
14: end for
```

Figure 4: Gropps'a CG [Ghysels 2014]

In the previous work [塙敏博，中島研吾，大島聡史，星野哲也，伊田明弘，パイプライン型共役勾配法の性能評価，情報処理学会研究報告（2016-HPC-157-6 ）， 2016 [Hanawa 2016]], we implemented these algorism to Reedbush-U in ITC/U.Tokyo using up to 384 nodes (12,288 cores). Figure 5 shows the results for strong scaling.

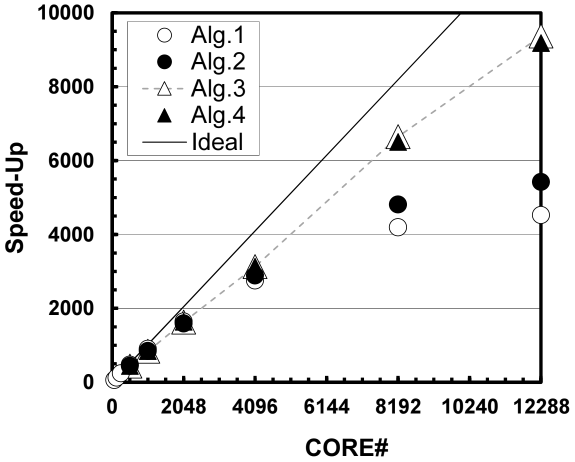


Figure 5: Results for Strong Scaling on Reedbush-U System using up to 384 nodes (12,288 cores), Alg.1: Original CG, Alg.2: Chronopoulos/Gear, Alg.3: Pipelined CG, Alg.4: Gropp's CG [Hanawa 2016]

While performance of original algorithm (Alg.1) is saturated at 8,192 cores and at 12,288 cores, Alg,3 (Pipelined CG (Fig.3)) and Alg.4 (Gropp's CG (Fig.4)) provide excellent scalability.

These results in Fig.5 were obtained in July 2016 using Intel MPI Library 2015 or 2016. Intel Compiler and Intel MPI Library have been improved significantly in recent years, therefore performance of MPI_Allreduce is much better now. If the performance is evaluated using Intel MPI Library 2017 or later, difference among Alg.1, Alg.3 and Alg.4 is small. Moreover, Alg.3 and Alg.4 are rather slower because they need more computation compared to Alg.1. Generally, it is difficult to observe effects of asynchronous collective communications without hardware support, such as communication assist cores on Fujitsu's FX100 system.

In Intel MPI Library 2019, performance of Asynchronous Progress Thread has been significantly improved, where efficient asynchronous collective communications are supported by software. Therefore, Communication-Computation Overlapping will be improved by Asynchronous Progress.

Figure 6 and Table 2 show the performance of Alg.1, Alg.3 and Alg.4 in Fig.5 and [Hanawa 2016] using up to 4,096 nodes (262,144 cores) of the Oakforest-PACS system [5]. Performance of Alg.4-IAR (Gropp's CG with MPI_Iallreduce) attained 40% speed-up compared to Alg.1-AR (Original CG) at 4,096 nodes of OFP.

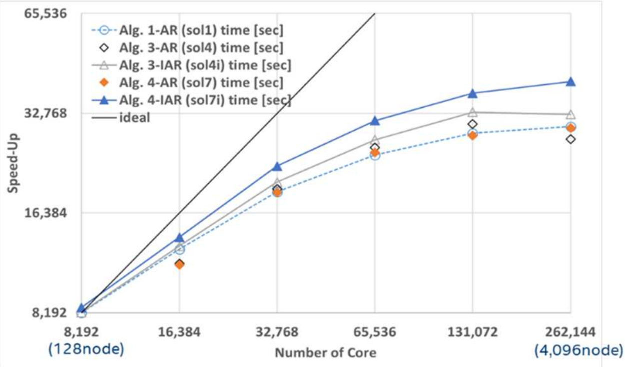


Figure 6: Results for Strong Scaling on Oakforest-PACS (OFP) System using up to 4,096 nodes (262,144 cores), Alg.1-AR: Original CG, Alg.3-AR/3-IAR: Pipelined CG, Alg.4-AR/4-IAR: Gropp's CG, AR: MPI_Allreduce, IAR: MPI_Iallreduce [5]

Table 2: Results for Strong Scaling on Oakforest-PACS (OFP) System using up to 4,096 nodes (262,144 cores), Speed-up of Alg.3-IAR and Alg.4-IAR against Alg.1-AR at each core/node number

Node #	Core #	Alg.3-IAR	Alg.4-IAR
128	8,192	1.03	1.06
256	16,384	1.13	1.21
512	32,768	1.05	1.20
1,024	65,536	1.06	1.25
2,048	131,072	1.08	1.34
4,096	262,144	1.19	1.38

6. Progress during FY2019 and Future Prospects

In FY.2019, we have evaluated performance of pipelined CG methods using up to 4,096 nodes of OFP system with capability of Asynchronous Progress Thread in Intel MPI Library 2019. Pipelined CG based on Gropp's algorithm was 40% faster than the original CG method at 4,096 nodes (262,144 cores) of OFP.

Due to the coronavirus epidemic, the research activities of the project were seriously disrupted during the last two months of FY2019. Specifically, the in-situ data analytics part of the work didn't get started. Moreover, the originally planned work on running huge-scale simulations was only partially initiated.

It is envisioned that our research effort will restart, as soon as the coronavirus situation alleviates to allow an appropriate work mode, for continuing the work on running large-scale simulations. The simulation results will be analyzed so that they will constitute the main physiological content of a journal submission. The HPC findings associated with the large-scale simulations will be collected and drafted as another journal submission.

7. List of Publications and Presentations

(1) Journal Papers (Refereed)

(2) Proceedings of International Conferences (Refereed)

[1] Chad Jarvis, Glenn Terje Lines, Johannes Langguth, *Kengo Nakajima*, Xing Cai. *Combining algorithmic rethinking and AVX-512 intrinsics for efficient simulation of subcellular calcium signaling.*

Proceedings of ICCS 2019 Conference, June 12-14, 2019, Faro, Portugal.

[2] Johannes Langguth, Hermenegild Arevalo, Kristian Hustad, Xing Cai. *Towards detailed real-time simulations of cardiac arrhythmia*. Proceedings of the International Conference in Computing in Cardiology, September 8-11, 2019, Singapore.

(3) International conference Papers (Non-refereed)

[3] Nakajima, K., Parallel Multigrid with Adaptive Multilevel hCGA on Manycore Clusters, Extreme-Scale/Exascale Applications China, Japan, World, ISC High Performance 2019, Frankfurt, Germany, 2019 (Invited Talk)

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

[4] Xing Cai. *Heterogeneous computing for cardiac electrophysiology.* Invited keynote talk at PREAPP workshop on Efficient Frameworks for Compute- and Data-intensive Computing (EFFECT), April 25-26, 2019, Tromsø, Norway.

[5] Masashi Horikoshi, Kengo Nakajima, Balazs Gerofi, Yutaka Ishikawa, Parallel Preconditioned Iterative Solvers on Oakforest-PACS, 2019 年並列／分散／協調処理に関する『北見』サマー・ワークショップ（SWoPP 北見 2018），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）（北見，2019 年 7 月 25 日）

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

N/A