Parallel Left Ventricle Simulation Using the FEniCS Framework

Timofei Epanchintsev[1,2,3] and Vladimir Zverev[3,1]

[1] Krasovskii Institute of Mathematics and Mechanics, Yekaterinburg, Russia
[2] Institute of Immunology and Physiology of the Ural Branch of the Russian Academy of Sciences, Yekaterinburg, Russia
[3] Ural Federal University, Yekaterinburg, Russia

Abstract. Heart simulation is complex task that requires multiscale modeling on cell, tissue and organ levels. Such structure makes difficult high performance code development and its maintenance. In this paper, we evaluate how scientific software could be used for heart simulation. An overview of existing frameworks for automated scientific computing is presented. The FEniCS framework was chosen since it supports automated solution of differential equations by the finite element method of parallel computing systems, provides near-mathematical notation, uses high performance backend and has comprehensive documentation. FEniCS performance was evaluated by simulation the space propagation of membrane potential alternation over a cardiac left ventricle using the electrophysiological model of a left ventricle and the Ekaterinburg-Oxford cell model. The FEniCS framework showed good performance and near-linear scalability up to 240 CPU cores.

Keywords: FEniCS · heart simulation · parallel computing · finite element method

1 Introduction

Heart simulation is a complex task that requires multiscale modeling on cell, tissue, and organ levels [1]. Such problems are computationally intensive and require parallel computing, including the use of modern computational accelerators such as GPU and Xeon Phi. However, porting a complicated multilevel simulation code to a new computational architecture requires a long time (often 3–5 years) during which the architecture may become obsolete. In addition, adaptation for parallel computing architectures often leads to significant changes of code. Consequently, it is very difficult to determine which mathematical models and numerical methods are used in the optimized code. Hence, reflecting the changes of a mathematical model in the optimized code can be very complicated. As a result, complex multiscale simulation software is rarely adapted to modern parallel computing architectures.

An alternative approach is based on using automated scientific computing frameworks. Such frameworks allow the development of simulation software using

programming languages with near-mathematical notation. Traditionally, a significant disadvantage of such frameworks was their low simulation performance. However, some modern implementations use advanced tools such as highly efficient mathematical libraries, just-in-time compilers, parallel execution, and so on, which improve their performance. Still, it is not clear if the modern automated scientific computing frameworks are efficient enough for real multiscale simulation tasks such as heart simulation.

In this paper, we evaluate how the automated scientific computing frameworks could be used for heart simulation on parallel computing systems. Heart simulation is an attractive task for evaluation of the performance of automated scientific computing frameworks because numerical simulations can be used as a virtual environment for testing and predicting tissue behavior in the cases where experimental techniques can not be applied [1, 2]. As a benchmark problem, we chose the investigation of the space propagation of the membrane potential alternation over the left ventricle of human heart.

2 Automated Scientific Computing Frameworks

Nowadays, finite element method is the most popular method for numerical investigation of systems with complex geometry, such as human heart. Several frameworks for automated scientific computing using finite element method exist. The most popular among them are OpenFOAM, OpenCMISS, Chaste, and FEniCS.

OpenFOAM (Open Source Field Operation and Manipulation) [3] is a free Computational Fluid Dynamics (CFD) software designed for solving problems in continuum mechanics. OpenFOAM is a C++ library that provides numerical schemes implemented in the traditional finite volume framework with solvers that are known to be efficient for continuum mechanics problems. OpenFOAM uses domain decomposition in order to provide parallel execution based on MPI.

OpenCMISS is a part of the global international project Physiome [4]. OpenCMISS is a mathematical modeling environment that enables the application of finite element analysis techniques to a variety of complex bioengineering problems. For distributed and parallel computing, OpenCMISS uses MPI, OpenMP, and ParMETIS.However, the OpenCMISS project is still under development, its documentation is incomplete, and it lacks examples.

Chaste (Cancer, Heart, and Soft Tissue Environment) is a general purpose simulation package aimed at multiscale, computationally intensive problems arising in biology and physiology [5, 6]. Current functionality includes the tissue and cell level electrophysiology, the discrete tissue modeling, and the soft tissue modeling. Chaste uses solvers from PETSc, mesh distribution algorithms from ParMETIS, and provides the ability to run parallel simulations using MPI. Chaste has already implemented models and methods, but to modify them a developer has to deal with sophisticated internal structure. Hence their further extension is complicated.

FEniCS [7] is a collaborative project for the development of the tools for automated scientific computing with a particular focus on the automated solution of differential equations by the finite element method. Implementation of a finite element method consists of several stages (obtaining equations' weak form, discretizing the equations in the weak form, assembling the values calculated on each element, and solving of the system of algebraic equations). Each stage is covered by a separate component of the FEniCS framework. It uses third-party high-performance libraries such as PETSc, Trilinos, uBLAS, or Intel MKL. In addition, FEniCS allows parallel simulation using MPI.

For our heart simulation task, we chose the FEniCS framework because it provides the automated solution of differential equations by finite element methods, supports automatic parallel execution using MPI, and has a good documentation.

3 Description of the Heart Model

We simulated the electrical activity of a human heart, which is a result of spatial and temporal propagation of the electrical signal from each cardiac cell. We used the Ekaterinburg-Oxford cell model [8] with a detailed description of electrical, chemical, and mechanical processes. On the intracellular level, the electrical potential arises from a very complicated interaction among ionic currents and cell organelles (organized structures in cells), as presented in Fig. 1 (left). The trans-membrane potential is shown in Fig. 1 (right). Scaled windows presents fast processes in period between 0 and 0.015 seconds.

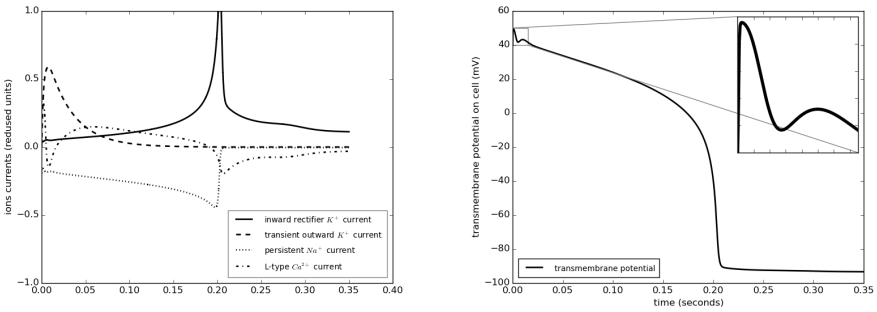

Fig. 1. The schemes of the electric potential and some of the underlying ionic current

The electrical block of the model contains the equations of the membrane potential and dynamic parameters describing the opening and closing of ion channels. The chemical block describes the kinetics of intracellular concentrations of calcium, sodium and potassium, extracellular potassium, the kinetics of calcium complexes, and calcium kinetics in the organelles. The mechanical block

of the model includes equations describing the voltage of the cell that depends on its length. The differential equations of the electrical, chemical, and mechanical processes can be presented in the following simplified form:

$$\frac{\partial S}{\partial t} = g(V, S), \tag{1}$$

where V is the electrical potential, S is the vector of model variables that govern the ion currents, and g is the vector-valued function that describes the time evolution of each variable. The dimension of the vectors S and g is 30. System (1) is defined at each point of the heart tissue, and, consequently, we should solve it for each node of the computational mesh. The space and time propagation of the electrical potential is governed by the "reaction-diffusion equation" [9]:

$$\frac{\partial V}{\partial t} = D\nabla^2 V + I_{ions}(V, S), \tag{2}$$

where D is the diffusion coefficient, ∇^2 is the Laplace operator, and I_{ions} is the sum of the ionic currents related to the capacitance of cell membrane. Boundary conditions correspond to the condition of electrical isolation.

This model is a nonlinear system of differential equations that can not be solved analytically and is a very computationally intensive task due to the large amount of variables in the 3D domain.

4 Benchmark Problem

During the experiments, we simulated the electrical activity of the human heart left ventricle (LV). We used the asymmetric model of LV that was previously developed in our group [10]. The important feature of the model is the ability to vary the size of the mesh elements. The personalized model parameters were captured with the help of ultrasound imaging. An example of 3D mesh for LV is presented in Fig. 2.

In order to solve model (1)–(2), we use the operator splitting scheme of first order (Marchuk–Yanenko method) [11]. Let us consider time domain $t \in [0, T]$ and the uniform grid $t_n = h_t n$, where $h_t = T/N$ and n is an integer that counts time level, $0 <= n <= N$. We denote a quantity at time t_n as V_n. The scheme of computing V_n and S_n consists of two steps. Let us assume that we have already calculated the values of $V(t)$ and $S(t)$ for $t < t_n$. At the first step, we solve the following partial differential equation:

$$\frac{V_n^* - V_{n-1}^*}{h_t} = D\nabla^2 V^*, V^*(t = t_{n-1}) = V(t_{n-1}), t \in [t_{n-1}, t_n]. \tag{3}$$

At the second step, we should solve the following system of ordinary equations:

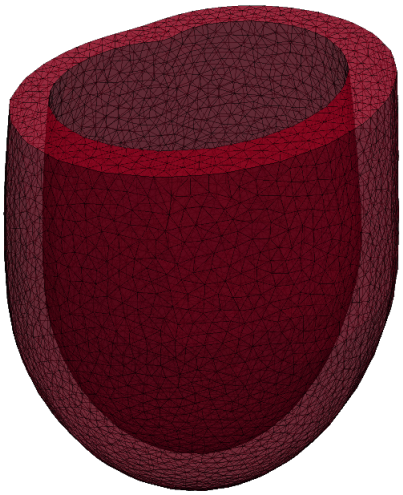

Fig. 2. An example of 3D mesh of left ventricle (asymmetric model)

$$\begin{aligned} &\frac{V_n^{**} - V_{n-1}^{**}}{h_t} = I_{ions}, V^{**}(t = t_{n-1}) = V^*(t_n), \\ &\frac{S_n^{**} - S_{n-1}^{**}}{h_t} = g(V^*, S^{**}), S^{**}(t = t_{n-1}) = S(t_{n-1}). \end{aligned} \tag{4}$$

The solution of (4) gives us the values of $V(t_n)$ and $S(t_n)$ according to the rules $V(t_n) = V^{**}(t_n)$ and $S(t_n) = S^{**}(t_n)$. This method allows us to tackle task (3) using the implicit method and use the explicit time-scheme for task (4). In addition, we avoid the Newton-like iterations. The disadvantage of such approach is that we have to use a very small integration time step, in order to capture the fast electrochemical processes (Fig. 1).

For testing purposes, we chose activation of an entire LV (the potential is greater than 40 millivolt) as our initial condition. The simulation duration period was 0.3 seconds of physical time because the electrical activity tends to the equilibrium state without an external stimulus approximately after this period.

We use the tetrahedral mesh that was generated by the GMSH software [12]. The minimal length of the tetrahedrons was set to 2mm and maximal to 4mm. As a result, the mesh contained 7178 points and 26156 tetrahedrons.

5 Performance Evaluation

In order to estimate the performance and scalability of the LV simulation using the FEniCS framework, a series of experiments was performed. We used the *Uran* supercomputer of the Krasovskii Institute of Mathematics and Mechanics. The

configuration parameters of the computational nodes are presented in Table 1. The FEniCS version 1.6.0 was used.

Table 1. Configuration of computational nodes

Configuration parameter	Value
CPU	2 x Intel(R) Xeon(R) CPU X5675 @ 3.07GHz
RAM	192 GB
Interconnect	Infiniband DDR (20 Gbit)
Operating System	CentOS Linux 7.2

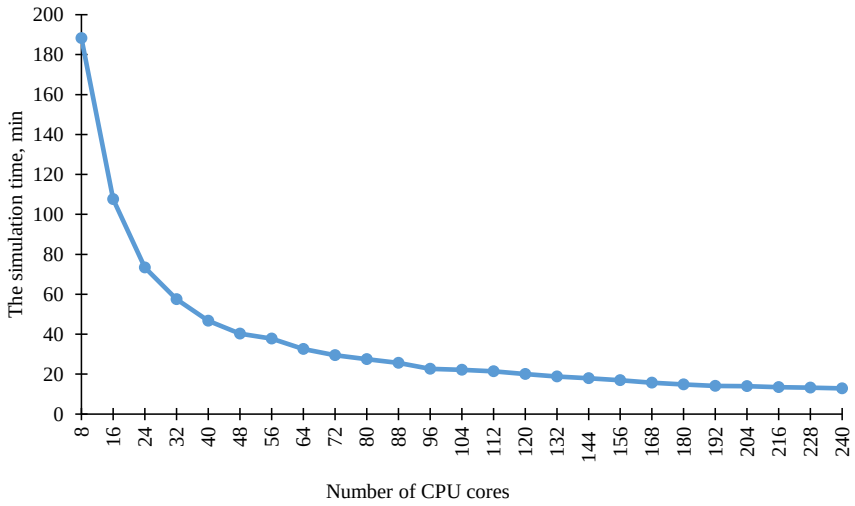

Fig. 3. Simulation time depending on the number of CPU cores

We estimated the scalability by conducting the simulation on varying numbers of CPU cores, from 1 to 240. Fig. 3 shows the simulation time using different numbers of CPU cores and Fig. 4 presents the achieved speedup.

6 Discussion

The FEniCS framework demonstrated good performance and near-linear scalability due to the data parallelism. The mesh was distributed among the computational nodes before the launch of the simulation. Almost all computations were performed independently except for the transfer of boundary values of each mesh fragment between the nodes.

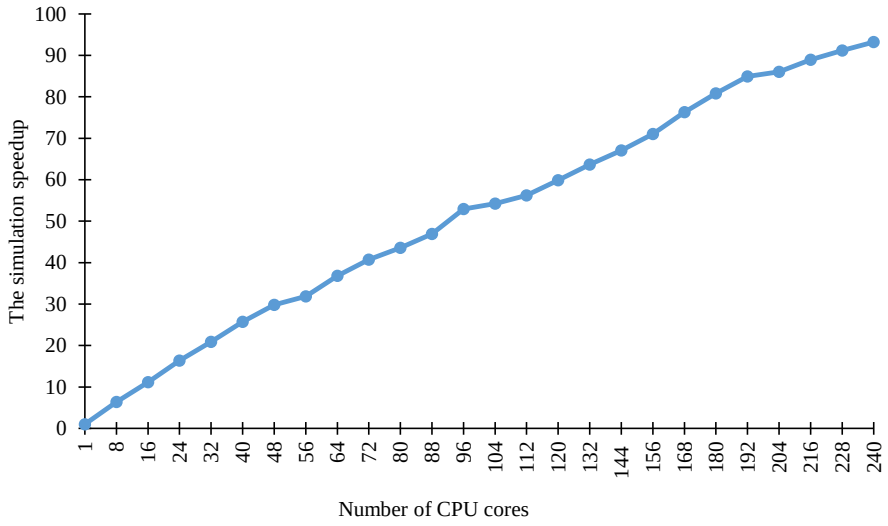

Fig. 4. Simulation speedup depending on the number of CPU cores

Our previous manual implementation of the same model in the LeVen system [13], which uses the C language and OpenMP for parallelization, provides approximately the same performance. However, the FEniCS-based solution has better scalability: it scales up to 240 CPU cores while LeVen scales only up to 8 cores. In addition, the FEniCS implementation can be easily modified since it has near–mathematical notation.

Another problem we faced was the import of the model's description from CellML [14] into FEniCS. CellML is a language created within the Physiome Project to describe mathematical models of cellular biological functions in order to aid distribution and reuse of the models. We use the CellML description of the Ekaterinburg-Oxford model as a basis for our code. However, the standard tool for converting CellML descriptions to the simulation program from the Physiome Project generates non human-readable code and, therefore, is unsuitable for further use. We found a workaround by using the tools from the Gotran Project [15] and converting the CellML model description to the UFL language. However, we had to manually edit output files on each stage due to the complexity of models and the limitations of the just-in-time compilation algorithm of FEniCS. For example, we had to replace expressions such as π^x by $e^{(ln(\pi)x)}$ because FEniCS does not support raising to a fractional power.

Conclusion and Future Work

The FEniCS framework is an efficient tool for heart simulation on parallel computing systems because it provides convenient near-mathematical notation, high simulation performance, and scales well. In comparison to our previous manual

implementation FEniCS provides better scalability and can be easily utilized by biologists, chemists or physicists.

Possible directions of future work include:

- Testing the scalability of FEniCS on thousands of CPU cores.
- Applying FEniCS for real tasks such as simulation of scroll wave dynamics.
- Evaluating the ability of FEniCS and other automated scientific computing frameworks to use modern computational accelerators such as GPU and Xeon Phi.
- Developing tools for automatic import of the CellML models to FEniCS.

Acknowledgments. This work was supported by the Russian Science Foundation (grant no. 14-35-00005). Our study was performed using the *Uran* supercomputer of the Krasovskii Institute of Mathematics and Mechanics.

References

1. Kerckhoffs, R.C.P., Healy, S.N., Usyk, T.P., McCulloch, A.D.: Computational methods for cardiac electromechanics. Proceedings of the IEEE **94** (2006) 769–783
2. Plank, G., Burton, R.A., Hales, P., Bishop, M., Mansoori, T., Bernabeu, M.O., Garny, A., Prassl, A.J., Bollensdorff, C., Mason, F., et al.: Generation of histo-anatomically representative models of the individual heart: tools and application. Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences **367**(1896) (2009) 2257–2292
3. Jasak, H., Jemcov, A., Tukovic, Z.: OpenFOAM: a C++ library for complex physics simulations. In: International workshop on coupled methods in numerical dynamics. Volume 1000. (2007) 1–20
4. Crampin, E.J., Halstead, M., Hunter, P., Nielsen, P., Noble, D., Smith, N., Tawhai, M.: Computational physiology and the physiome project. Experimental Physiology **89**(1) (2004) 1–26
5. Pitt-Francis, J., Pathmanathan, P., Bernabeu, M.O., Bordas, R., Cooper, J., Fletcher, A.G., Mirams, G.R., Murray, P., Osborne, J.M., Walter, A., et al.: Chaste: a test-driven approach to software development for biological modelling. Computer Physics Communications **180**(12) (2009) 2452–2471
6. Mirams, G.R., Arthurs, C.J., Bernabeu, M.O., Bordas, R., Cooper, J., Corrias, A., Davit, Y., Dunn, S.J., Fletcher, A.G., Harvey, D.G., et al.: Chaste: an open source C++ library for computational physiology and biology. PLoS Comput Biol **9**(3) (2013) e1002970
7. Alnæs, M.S., Blechta, J., Hake, J., Johansson, A., Kehlet, B., Logg, A., Richardson, C., Ring, J., Rognes, M.E., Wells, G.N.: The FEniCS project version 1.5. Archive of Numerical Software **3**(100) (2015)
8. Solovyova, O., Vikulova, N., Katsnelson, L.B., Markhasin, V.S., Noble, P., Garny, A., Kohl, P., Noble, D.: Mechanical interaction of heterogeneous cardiac muscle segments in silico: effects on Ca 2+ handling and action potential. International Journal of Bifurcation and Chaos **13**(12) (2003) 3757–3782
9. Katsnelson, L.B., Vikulova, N.A., Kursanov, A.G., Solovyova, O.E., Markhasin, V.S.: Electro-mechanical coupling in a one-dimensional model of heart muscle fiber. Russian Journal of Numerical Analysis and Mathematical Modelling **29**(5) (2014) 275–284

10. Pravdin, S.: Non-axisymmetric mathematical model of the cardiac left ventricle anatomy. Russian Journal of Biomechanics **17**(62) (2013) 75–94
11. Li, Y., Chen, C.: An efficient split-operator scheme for 2-D advection-diffusion simulations using finite elements and characteristics. Applied Mathematical Modelling **13**(4) (1989) 248–253
12. Geuzaine, C., Remacle, J.F.: Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. International Journal for Numerical Methods in Engineering **79**(11) (2009) 1309–1331
13. Sozykin, A., Pravdin, S., Koshelev, A., Zverev, V., Ushenin, K., Solovyova, O.: LeVen - a parallel system for simulation of the heart left ventricle. 9th International Conference on Application of Information and Communication Technologies, AICT 2015 Proceedings (2015) 249–252
14. Cuellar, A.A., Lloyd, C.M., Nielsen, P.F., Bullivant, D.P., Nickerson, D.P., Hunter, P.J.: An overview of CellML 1.1, a biological model description language. SIMULATION **79**(12) (2003) 740–747
15. Hake, J.E.: A general ODE translator (gotran): Towards a versatile tool for general ODEs. CBC and CCI Workshop on Advancing Numerical Technologies in the Cardiac Domain, May 15 (May 2013)