Analysis of the Efficiency PETSc and PETIGA Libraries in Solving the Problem of Crystal Growth

Ilya Starodumov[1], Evgeny Pavlyuk[1], Leonid Klyuev[2], Maxim Kovalenko[3], and Anton Medyankin[1]

[1] Ural Federal University, Yekaterinburg, Russia,
{ilya.starodumov,evgeny.pavluk}@urfu.ru
[2] Immers Ltd., Moscow, Russia,
l.klyuev@immers.ru
[3] The Program Systems Institute of RAS, Pereslavl-Zalessky, Russia,
kovalenko@botik.ru

Abstract. We present an analysis of high performance computational method for solving the problem of crystal grows. The method uses PETSc and PETIGA C-language based libraries and supports parallel computing. The evolution of calculation process was studied in series of special computations are obtained on innovative mobile cluster platform, which provides exclusive system tuning abilities. The results of research confirm the high efficiency of the proposed algorithm on multi-core computer systems and allow us to recommend the use of PETSc and PETIGA for solving high order differential equations.

Keywords: PETSc · PETIGA · parallel · high perfomance · crystal grows · isogeometric analysis

1 Introduction

Problems involving differential operators of order more than two have not historically lent themselves well to finite element analysis[4]. Such applications have the variational statements with second derivatives, requiring the use of a globally C^1- continuous basis. The complexity of the general solution of this problem led to the use of finite-difference and spectral methods, both of which are viable methods, but far more limited than FEA in their scope and flexibility. By the use of isogeometric analysis, we have a higher-order accurate, robust method with great geometric suppleness and compactly supported basis functions. At the same time a higher order continuity is still possible. Thus, it is a convenient technology for the study of equations involving higher-order differential operators[10].

1.1 Phase field models

Two different approaches have been used to describe phase transition phenomena: sharp interface models and phase-field(diffuse-interface) models[9]. Usually,

the evolution of interfaces, such as the liquidsolid interface, has been described by sharp-interface models. That approach needs the resolution of a moving boundary problem, separate differential equations hold in each phase, and certain quantities may suffer jump discontinuities across the interface. Phase-field models provide an alternative description for phase-transition phenomena. It is possible due to approximating the interface as being diffuse such that it does not need to be tracked explicitly.An other description for phase-transition phenomena was provided by phase-field models. Such models can be derived from classical irreversible thermodynamics. Developed by K. R. Elder et al. as recently as 2002[6, 5] the PFC model, which shares many features with the CDFT (the classical density functional theory) of freezing, was presented as an extension of the PF models to study processes with smaller length scales.Essential progress has been made in the simulation of the parabolic PFC-equation[11–13], special efforts are required to solve numerically the modified (hyperbolic) PFC-equation due to the second-order time derivative of the equation. One of the challenges to PFC has been modeling different close-packed crystal structures[7, 2]. Such a task in three-dimensional case will be considered in the current article further.

1.2 The modified phase field crystal problem

Originally, the PFC model has been formulated in a parabolic form. For now it has been extended to allow faster degrees of freedom consistent with inertia in correspondence to transformation propagative regimes. Particularly, a modified or hyperbolic PFC model which includes an inertial term was introduced, and therefore, gives a possibility of the description of both fast and slow dynamics of transition [8]. The modified phase field crystal model describes a continuous field of atomic density $\phi(x,t)$ and it is expressed by the sixth order in space and second order in time equation:

$$\tau\frac{\partial^2\phi}{\partial t^2}+\frac{\partial\phi}{\partial t}=\nabla^2\mu, \tag{1}$$

where t is the time, τ is the relaxation time of the atomic flux to its stationary state, and μ represents the chemical potential, obtained from the free-energy functional

$$\mathcal{F}[\phi,\nabla\phi,\nabla^2\phi]=\int_\Omega\left[f(\phi)-|\nabla\phi|^2+\frac{1}{2}(\nabla^2\phi)^2\right]\mathrm{d}\Omega, \tag{2}$$

associated to the domain Ω. The chemical potential is can be obtained as the variational derivative of the free-energy functional $\mathcal{F}$, namely

$$\mu(\phi)=\frac{\delta\mathcal{F}}{\delta\phi}=f'(\phi)+2\nabla^2\phi+\nabla^4\phi. \tag{3}$$

The function f represents the homogeneous part of the free energy density. It takes on the form

$$f(\phi)=\frac{1-\epsilon}{2}\phi^2+\frac{\alpha}{3}\phi^3+\frac{1}{4}\phi^4. \tag{4}$$

Here, $\epsilon = (T_c - T)/T_c$ is the undercooling, where T and T_c are the temperature and critical temperature of transition, respectively. α is a coefficient which means a measure of metastability.

2 Computational experiments

The modified PFC equation is a hyperbolic differential equation of the sixth order. The solution of this equation from the computational point of view is not an easy task. Therefore, we developed special numerical algorithm using a C-language code based on the PETIGA library[3]. This software can be described as an extension of PETSc[1] that adds the utilization of IGA capability. PETSc is a suite of data structures and routines that provide frames to develop large-scale application codes on parallel computers and consists of parallel linear and nonlinear equation solvers and time integrators[14]. Specification of the computational algorithm is not a subject of current work, but it is represented in [2], where the software allows to get the first numerical results on three dimensional structures predicted by the modified phase field crystal equation. It should be noted that the algorithm takes advantage of the PETSC for parallel computations based on MPI methods. Experience of simulations in [2] shows that the computational complexity of the software is quite high and significant results are impossible without the use of high-performance clusters. This makes it relevant to the optimization problem of the program, which should start with a study of the effectiveness of the already implemented algorithm. The most importaint issues are the parallelization efficiency and amount of required computational resources.

To assess the performance of the computational program, we developed a series of experimental tasks. We made 3 types of experiments for homogeneous HPC cluster. For correct analysis, the configuration of hardware and software computer system must be optimized in order to minimize possible errors. Thus, the researchers had the task to form an experimental computational cluster in a short time with exclusive access to hardware and software. Such an approach could quickly adjust the computer system for each experiment with the requirements to minimize measurement uncertain. The use of traditional large-scale computer systems for this task seems impractical and required a significant amount of effort and time. At the same time, the use of personal workstations is not possible because of their insufficient performance. In this situation, we able to find a solution with innovative Immers technology. Such solutions make it possible to build compact HPC autonomous mobile computational clusters with a form factor close to the personal workstation. In that conditions we got the ability to use the optimum set of system software and hardware settings. Due to this potential sources of measurement error were fixed.

The cluster configuration includes 5 computational nodes connected by Infiniband QDR network of 40 GB/sec. Each compute node consists of two 14-core processor Intel E5-2697 v3 and 64GB of DDR4 RAM. All computational nodes were running under the managment of SLES 11.3 OS. No extra optimization

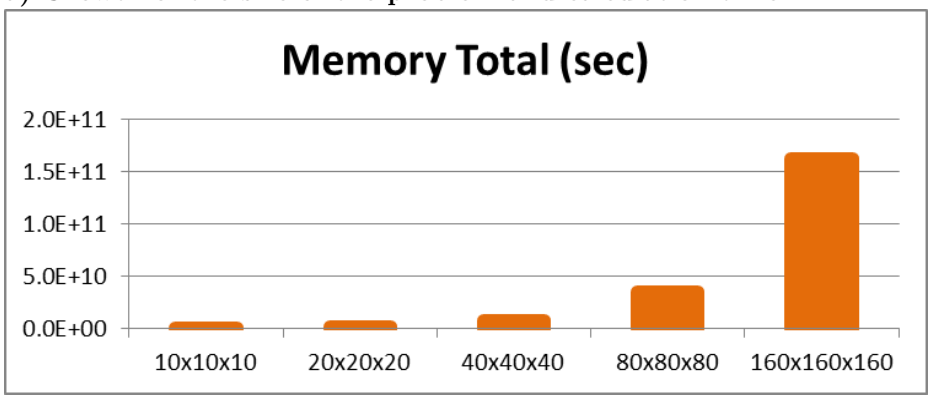
software were installed. In all experiments, the program calculated the task for 1 time step.

Descriptions and the results of experimental calculations are presented below. The purpose of these calculations was to evaluate the amount and specificity of computational resources, to assess the balancing of the program parallelization and estimate the amount of overhead.

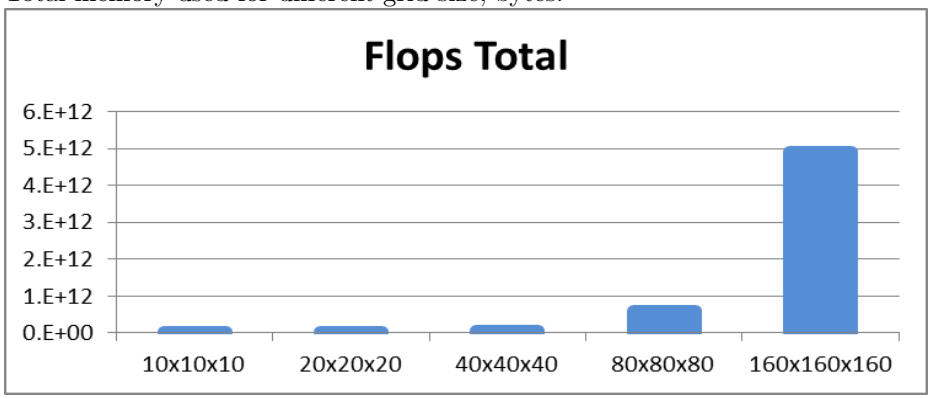
2.1 First experiment

The purpose of the first experiment is evaluation of dependence the complexity of the problem from the amount of finite elements. Fixed parameters for this experiment are: computational domain size 160x160x160 and using of 5 nodes including 28 processor cores on each node. Variable parameter is a grid size: 10x10x10, 20x20x20, 40x40x40, 80x80x80, 160x160x160. Results of the experiment are presented in the following figures:

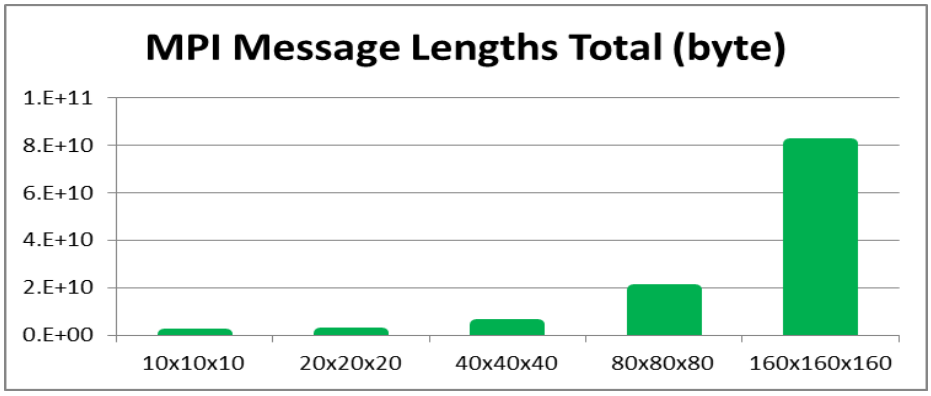
a) Growth of the size of the problem and calculation time



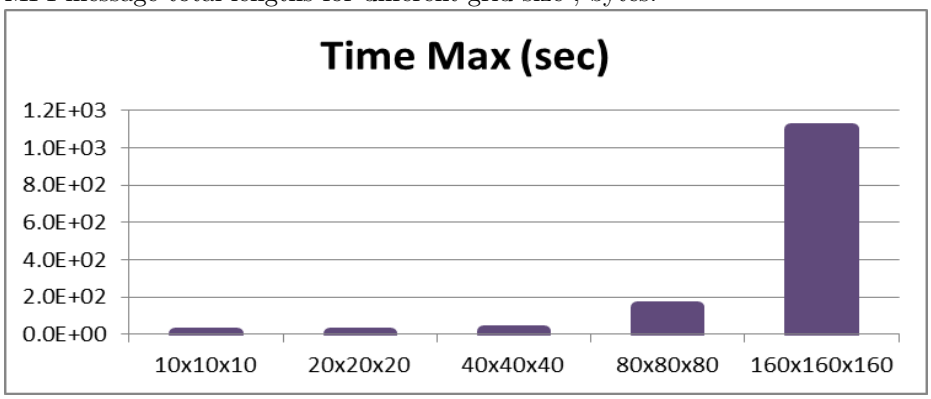
Total memory used for different grid size, bytes.



Total flops for different grid size.

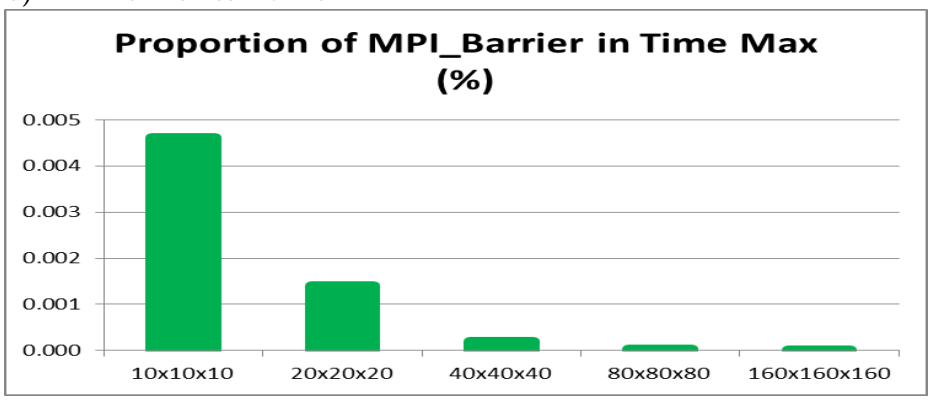


MPI message total lengths for different grid size , bytes.



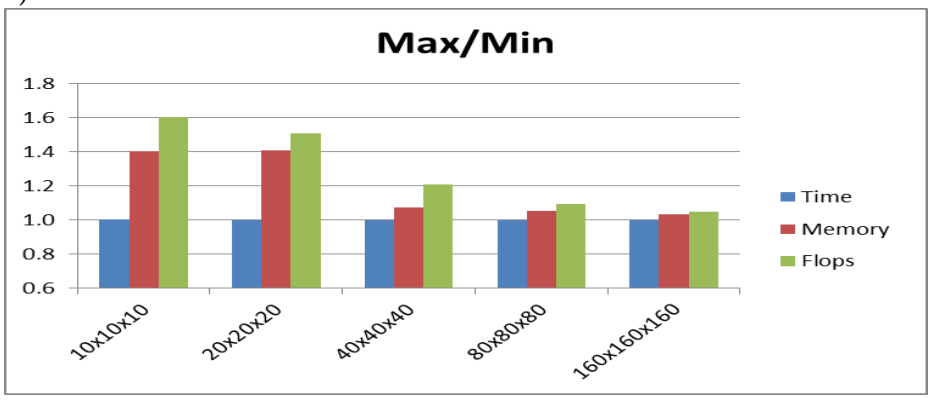
Maximum computational time for a single core in cases of different grid size, sec.

b) MPIBarrier call time



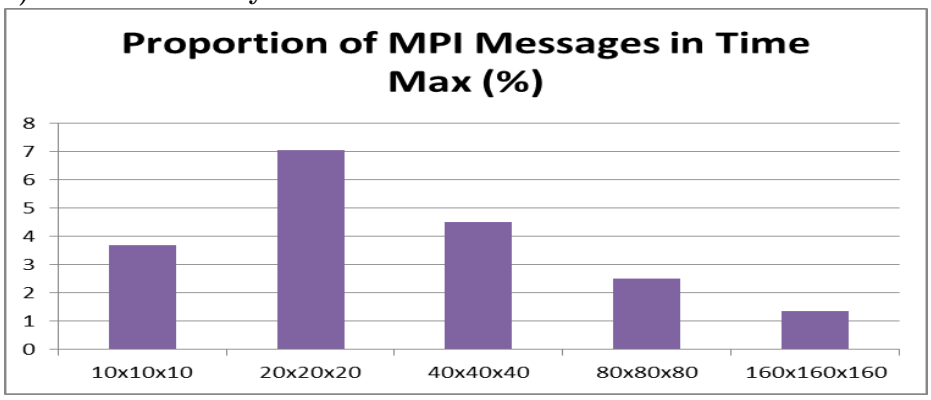
Percentage of MPIBarrier call time in the maximum computation time for a single core in cases of different grid size.

c) Some indicators of the balance



[Blue]The ratio of the maximum time for a single core to a minimum. [Red]The ratio of the maximum memory usage to a minimum during the calculation. [Green]The ratio of the maximum flops indicator to a minimum during the calculation. Diagrams for different grid size.

c) Network activity

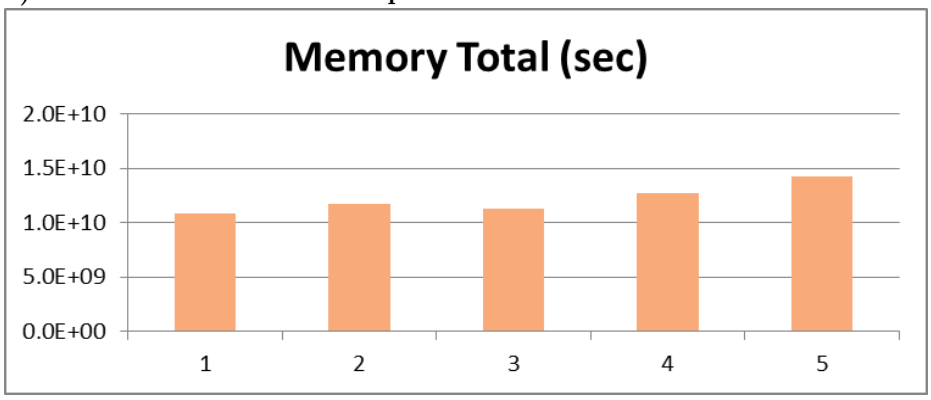


Percentage of MPI messages in maximum computation time for a single core in cases of different grid size.

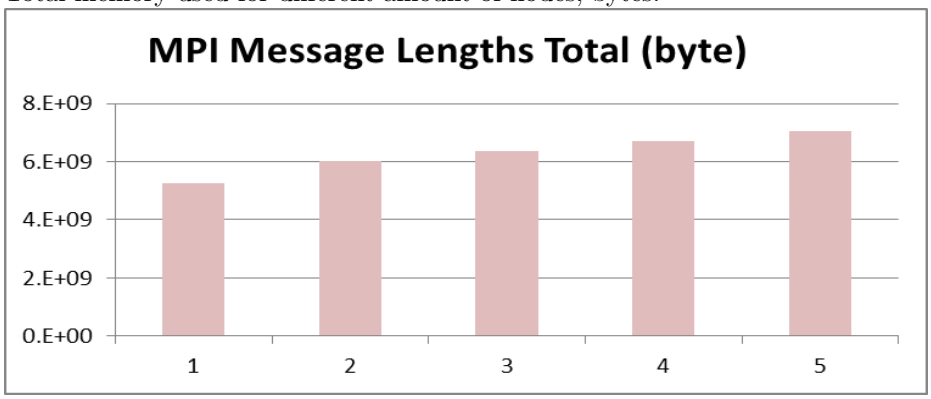
2.2 Second experiment

The purpose of the second experiment is the assessment of the efficiency of the algorithm parallelization by increasing the number of computing nodes. Fixed parameters for this task are: computational domain size 160x160x160 and the grid size 50x50x50. Variable parameter is the number of nodes: from 1 to 5 nodes included 28 processor cores on each node. Results of the experiment are presented in the following figures:

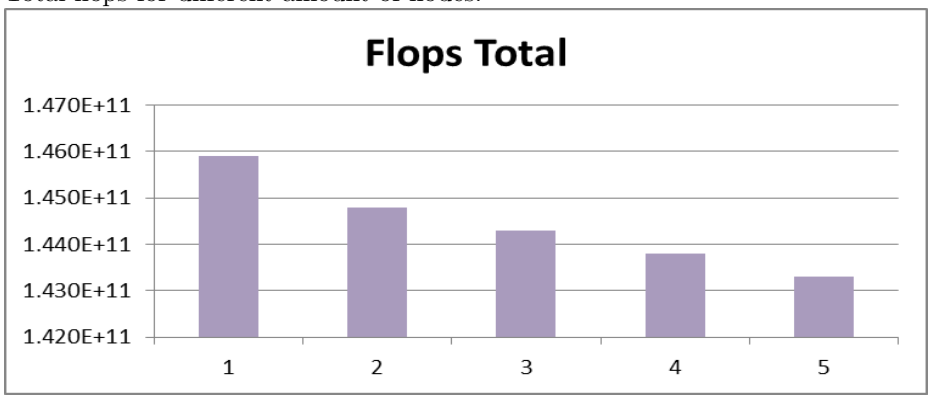
a) Growth of the size of the problem and calculation time



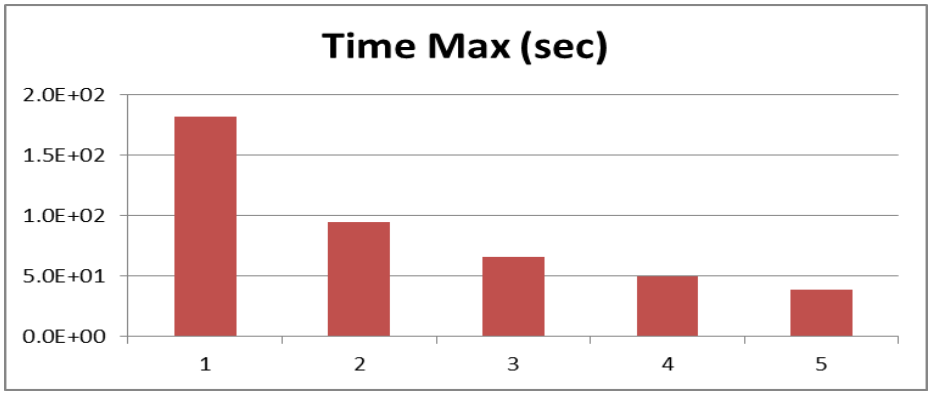
Total memory used for different amount of nodes, bytes.



Total flops for different amount of nodes.

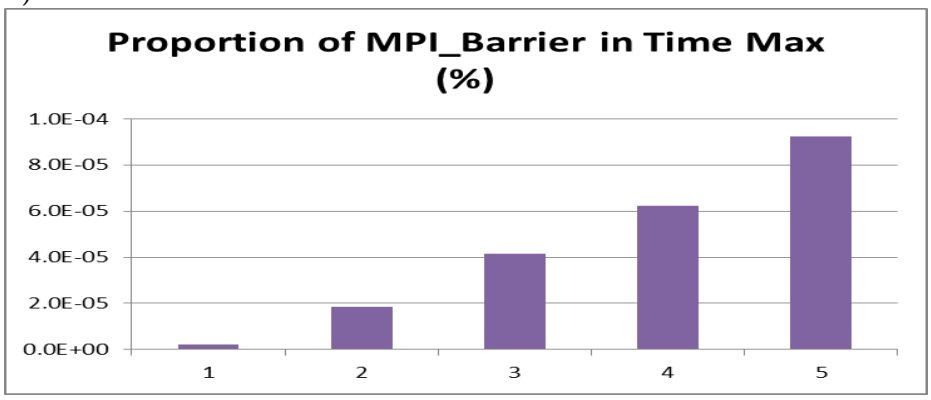


MPI message total lengths for different amount of nodes , bytes.



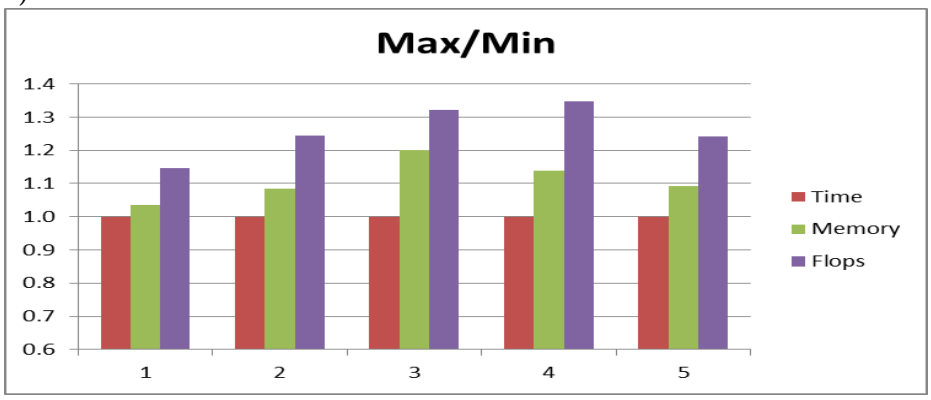
Maximum computational time for a single core in cases of different amount of nodes, sec.

b) MPIBarrier call time



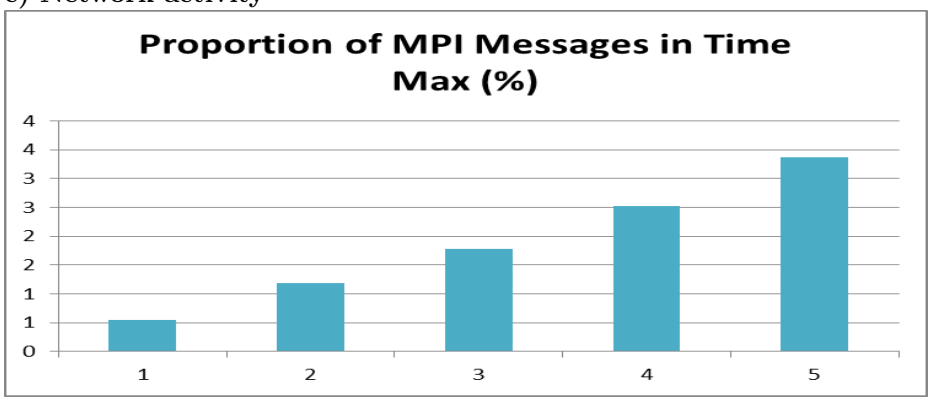
Percentage of MPIBarrier call time in the maximum computation time for a single core in cases of different amount of nodes.

c) Some indicators of the balance



[Red]The ratio of the maximum time for a single core to a minimum. [Green]The ratio of the maximum memory usage to a minimum during the calculation. [Blue]The ratio of the maximum flops indicator to a minimum during the calculation.

c) Network activity

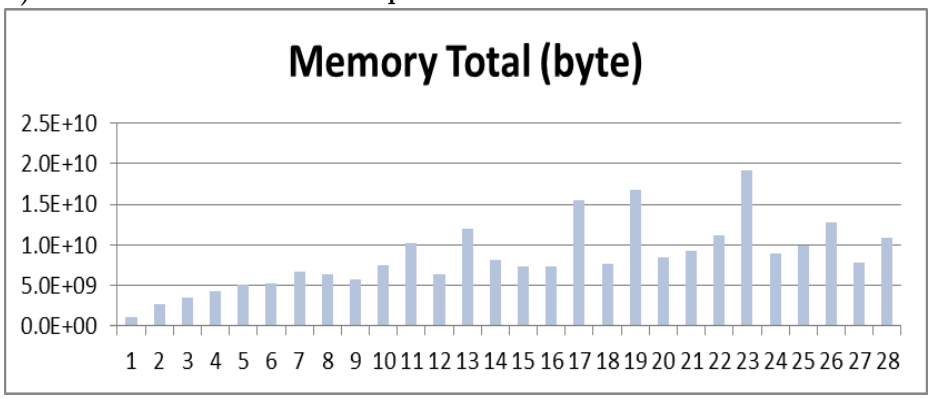


Percentage of MPI messages in maximum computation time for a single core in cases of different amount of nodes.

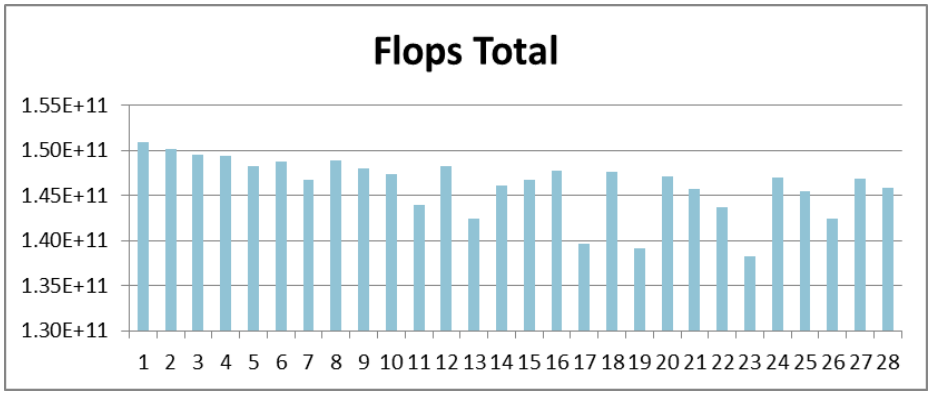
2.3 Third experiment

The purpose of of the third experiment is estimation of efficiency of the algorithm parallelization by increasing the number of cores on single node. Fixed parameters for this task are: only one node, computational domain domain size 160x160x160 and the grid size 50x50x50. Variable parameter is the number of cores on single node: from 1 to 28. Results of the experiment are presented in the following figures:

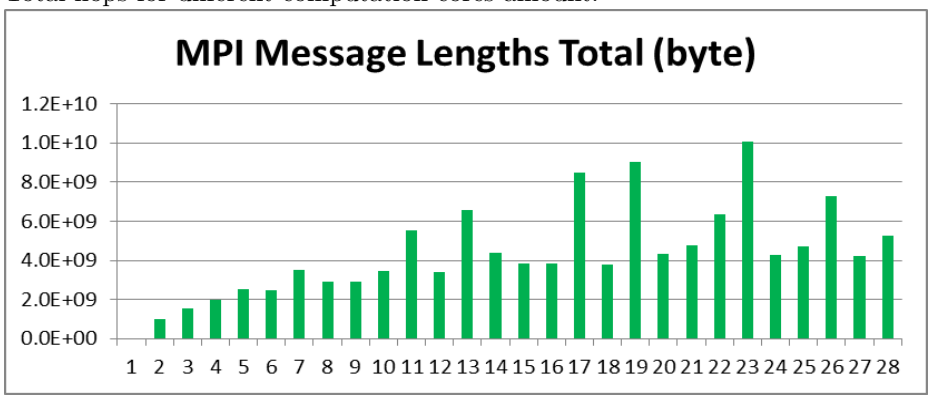
a) Growth of the size of the problem and calculation time



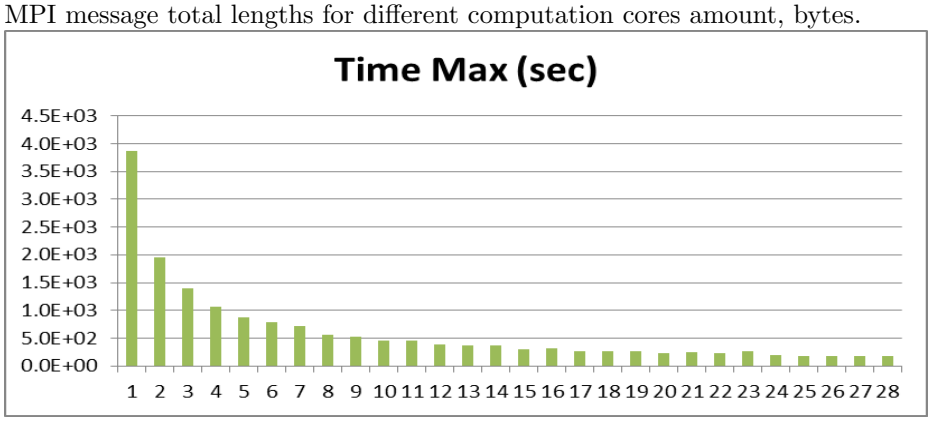
Total memory used for different computation cores amount, bytes.



Total flops for different computation cores amount.

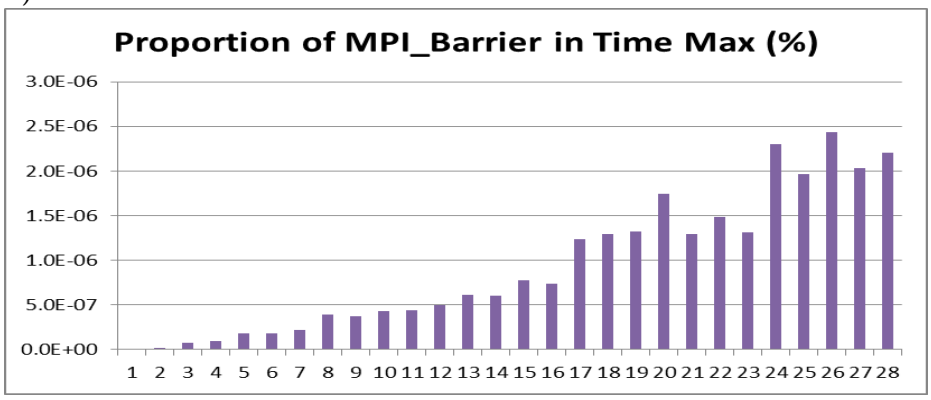


MPI message total lengths for different computation cores amount, bytes.



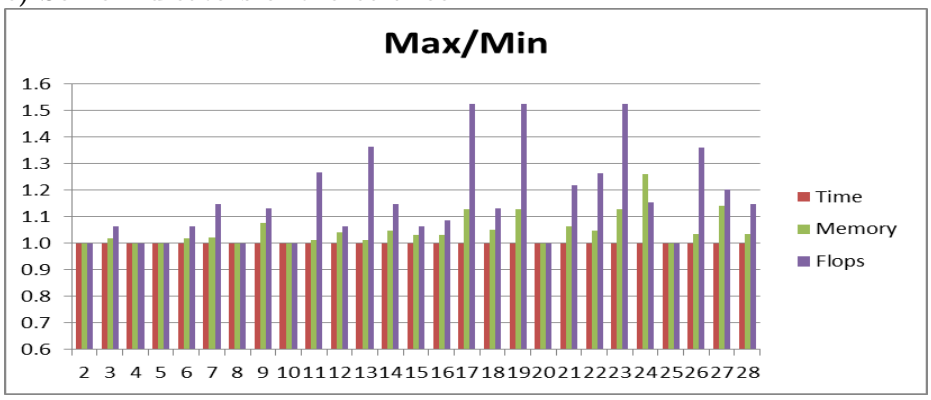
Maximum computational time for a single core in cases of different computation cores amount, sec.

b) MPIBarrier call time



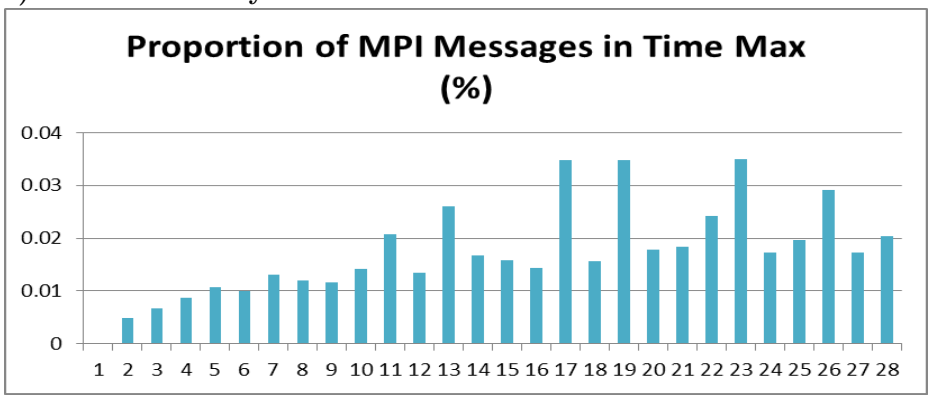
Percentage of MPIBarrier call time in the maximum computation time for a single core in cases of different computation cores amount.

c) Some indicators of the balance



[Red]The ratio of the maximum time for a single core to a minimum. [Green]The ratio of the maximum memory usage to a minimum during the calculation. [Blue]The ratio of the maximum flops indicator to a minimum during the calculation.

c) Network activity



Percentage of MPI messages in maximum computation time for a single core in cases of different computation cores amount.

3 Results

During computing the tasks the cluster Immers showed the best performance 4.45E+09 Flops/sec, 3.68E+09 Flops/sec and 8,03E+08 Flops/sec for the first, second and third experiments, respectively.The experiments results suggest the following conclusions:

3.1 Estimating the size of the problem and the computation time

In the first experiment, the total memory usage, the total number of operations and the total size of messages MPI grow exponentially. These indicators are rising at roughly the same speed. In the second experiment, the total amount of memory used and the total size of MPI messages almost unchanged. The total number of operations is slightly reduced, and this process requires further study. The computation time is also reduced with a logarithmic rate as expected. In the third experiment, there is a large variation in the total amount of memory used - this effect requires further study. Variations in the total number of operations and the total amount of MPI message, apparently associated to variations from memory. Computational time is change expected.

3.2 Balancing

In the first experiment, variation in the memory up to 40%, the number of operations up to 60%. In the second experiment, variation in the memory up to 20%, the number of operations up to 35%. In the third experiment, variation in the memory up to 28%, the number of operations up to 55%. The indicators seem to be unexpectedly large and require further research.

3.3 Overhead

The overhead of synchronization, as expected, increases with increasing amounts of computing nodes and decreasing the size of the grid. Meanwhile, the proportion of execution time MPI Barrier in total computation time is less than 0.001%, which is quite a bit.

4 Conclusions and further work

Current work shows that the solution of the crystal growth problem in the hyperbolic statement allows to simulate the structural transformation of matter at the supercooling. These studies have a big practical importance in the materials science applications. The computational complexity of the hyperbolic PFC equation is forcing scientists to develop special high-performance algorithms. One of the algorithms, which is used PETSC and PETIGA libraries, discussed in the article. This algorithm has already shown effectiveness, but the important issues are the conditions of its applicability and the possibility of it's improvement in terms of parallel programming. An important step in the matter of such issues is to study the efficiency of use of computing resources. To investigate this question, the authors have set specific numerical experiments. To reduce the error in the results due to the peculiarities of the computing cluster setup researchers have used homogeneous autonomous mobile computer. This approach has proved successful in the sense that the used hardware configuration eliminates the influence of unaccounted factors in the work of the computational algorithm.
The calculation results have showed a good balance of parallel computing. Authors have evaluated the overall computing resources. We can say that the amount of consumed memory and CPU time is linearly dependent on the complexity of the problem in terms of the number of computational cells. Also, in all the experiments overheads for synchronization tasks were low and even not up-to-date QDR network system was mostly idle. We believe that these results are a consequence of the high-quality implementation of computational algorithm. Some issues are deviations in performance of computational algorithm in the third experiment. Apparently, they are caused by the peculiarities of the CPU and data exchange on the nodes. It should be noted that these variations are generally not significant effect on the calculations efficiency.
The continuation of this work would be conducting similar experiments on a heterogeneous HPC cluster. Comparison of the results with the results of this paper will be useful for choosing the best HPC configuration with capability of solving of hyperbolic PFC equations. For researchers,who conduct a big series of experiments, these conclusions can be especially interesting.

References

1. Balay, S., Abhyankar, S., Adams, M., Brown, J., Brune, P., Buschelman, K., Eijkhout, V., Gropp, W., Kaushik, D., Knepley, M., et al.: Petsc users manual revision 3.5 (2014)
2. Bueno, J., Starodumov, I., Gomez, H., Galenko, P., Alexandrov, D.: Three dimensional structures predicted by the modified phase field crystal equation. Computational Materials Science 111, 310 – 312 (2016)
3. Collier, N., Dalcin, L., Calo, V.M.: Petiga: high-performance isogeometric analysis. preprint arXiv:1305.4452 (2013)
4. Cottrell, J., Hughes, T., Bazilevs, Y.: Isogeometric Analysis: Toward Integration of CAD and FEA. Wiley (2009)
5. Elder, K.R., Grant, M.: Modeling elastic and plastic deformations in nonequilibrium processing using phase field crystals. Physical Review E 70(5), 051605 (2004)
6. Elder, K.R., Katakowski, M., Haataja, M., Grant, M.: Modeling elasticity in crystal growth. Physical review letters 88(24), 245701 (2002)
7. Galenko, P.K., Gomez, H., Kropotin, N.V., Elder, K.R.: Unconditionally stable method and numerical solution of the hyperbolic phase-field crystal equation. Phys. Rev. E 88, 013310 (Jul 2013)
8. Galenko, P.K., Danilov, D.A., Lebedev, V.G.: Phase-field-crystal and Swift-Hohenberg equations with fast dynamics. Physical Review E 79(5), 051110 (2009)
9. Gomez, H., Calo, V., Bazilevs, Y., Hughes, T.: Isogeometric Analysis of the Cahn-Hilliard phase-field model. Comput. Methods Appl. Mech. and Eng. 197, 43334352 (2008)
10. Gomez, H., Hughes, T., Nogueira, X., Calo, V.: Isogeometric Analysis of the isothermal Navier-Stokes-Korteweg equations. Comput. Methods Appl. Mech. and Eng. 199(25-28), 1828–1840 (2010)
11. Gomez, H., Nogueira, X.: An unconditionally energy-stable method for the phase field crystal equation. Computer Methods in Applied Mechanics and Engineering 249, 52–61 (2012)
12. Tegze, G., Bansel, G., Tóth, G.I., Pusztai, T., Fan, Z., Gránásy, L.: Advanced operator splitting-based semi-implicit spectral method to solve the binary phase-field crystal equations with variable coefficients. Journal of Computational Physics 228(5), 1612–1623 (2009)
13. Tóth, G.I., Tegze, G., Pusztai, T., Tóth, G., Gránásy, L.: Polymorphism, crystal nucleation and growth in the phase-field crystal model in 2d and 3d. Journal of Physics: Condensed Matter 22(36), 364101 (2010)
14. Zhang, J.: A petsc-based parallel implementation of finite element method for elasticity problems. Mathematical Problems in Engineering vol. 2015, 7 (2015)