The Algorithms Properties and Structure Study as a Mandatory Element of Modern IT Education

Alexander Antonov[1,2(✉)] and Vladimir Voevodin[1,2]

[1] Lomonosov Moscow State University, Moscow, Russia
[2] Moscow Center of Fundamental and Applied Mathematics, Moscow, Russian Federation
{asa,voevodin}@parallel.ru

Abstract. The paper describes the system of mass practical assignments formed in the framework of the course "Supercomputing Simulation and Technologies" at the Faculty of Computational Mathematics and Cybernetics at Lomonosov Moscow State University. The practical assignments were held for four years, from 2016 to 2019, and each year about 200 students of the second year of the magistracy passed through them. These practical assignments are aimed at developing skills in the efficient use of parallel computing systems and knowledge of properties of parallel algorithms. The basics of performing the described assignments is the deep analysis of parallel algorithms and use of the concept of the information structure of algorithms and programs. This concept is extremely important for writing efficient parallel programs, so it should be the cornerstone of modern IT education.

Keywords: High-performance computing education · Structure of parallel algorithms · Information structure · Parallelism resource · Parallel programming · Supercomputers · AlgoWiki

1 Introduction

Understanding the fact that parallelism is an integral part of modern computing devices has long spread throughout the world. Therefore, it is not surprising that the study of parallel technologies has become the basis for the construction of many educational curricula in a variety of educational institutions [1–13].

The study of the algorithms parallel structure and properties has become a key objective of a series of practical assignments that were held as a part of the course "Supercomputing Simulation and Technologies" in 2016–2019. This course is taught to Master's degree students of the Faculty of Computational Mathematics and Cybernetics (CMC) at Lomonosov Moscow State University (MSU) during the final year of education. Annually, about 230 students study this discipline.

V. Voevodin and S. Sobolev (Eds.): RuSCDays 2020, CCIS 1331, pp. 524–535, 2020.
https://doi.org/10.1007/978-3-030-64616-5_45

The course covers the following topics:

- computationally complex tasks in various application fields;
- architectural features of modern processors, new approaches to creating high-performance systems;
- introduction to the analysis of the parallel structure of algorithms;
- scalable algorithms for parallel supercomputer systems: problems and prospects;
- algorithmic approaches for solving problems in various application areas: turbulence, molecular dynamics, climate change, supercomputer drug design etc.;
- additional chapters of MPI, OpenMP and CUDA parallel programming technologies;
- and others.

This course consists of lectures and seminars. Lectures are given on a weekly basis in the fall semester, for four academic hours per week.

The purpose of the practical assignments given to students in the framework of this course is developing skills in the efficient use of parallel computing systems.

2 Formulation of Practical Assignments

The overall goal of the practical assignments was to learn to explore and understand the parallel structure and properties of algorithms. In some cases, students were allowed to work in pairs, which contributes to the development of teamwork skills.

To perform numerical experiments, a wide variety of supercomputer platforms available in the supercomputing center of Moscow State University were provided:

- Lomonosov-2 supercomputer [14],
- Lomonosov supercomputer [15],
- Blue Gene/P supercomputer,
- computing cluster with Angara network,
- systems based on Power8 / Phi (KNL) / GPU (K40, Pascal).

A feature of the verification and acceptance of practical assignments was that in this case it was important not only to give an assessment, but to learn how to analyze and describe the properties of algorithms correctly. This, of course, requires highly qualified teachers and considerable time for multiple iterations in communication with students.

2.1 Practical Assignments on the Study and Description of the Structure and Properties of Algorithms (2016)

Study and describe the structure and properties of the selected algorithm.

Despite the simplicity of the formulation, the assignment is not at all trivial. It is difficult to even explain what it means to describe the structure and properties of algorithms. In our case, the methodological basis of this practical task is an AlgoWiki Open encyclopedia of parallel algorithmic features [16,17].

From the complete universal structure for describing the algorithms of the AlgoWiki project, for the practical assignments it was required to describe the entire first part related to the machine-independent properties of the algorithms. Of the dynamic characteristics of the algorithms described in the second part [18], only the study of the scalability of the algorithm implementation and the compilation of a list of existing implementations were left (Fig. 1).

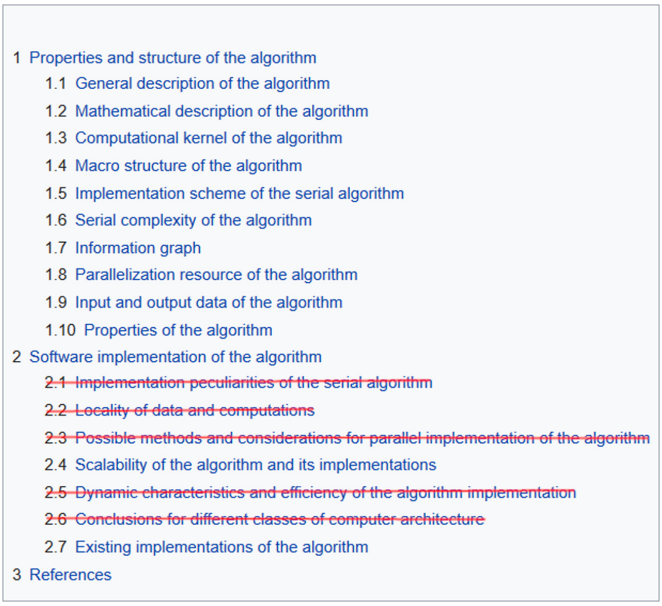

Fig. 1. The algorithm description structure

The students (or pairs) could select one of 30 proposed algorithms for description, particularly:

- Jacobi's method for the symmetric eigenvalue problem,
- Jacobi's method for the singular value decomposition,
- the Lanczos algorithm with full reorthogonalization for the symmetric eigenvalue problem,
- Gram-Schmidt orthogonalization process,
- GMRES (Generalized Minimal RESidual method) as an iterative method for solving systems of linear algebraic equations,
- QR algorithm for solving the algebraic eigenvalue problem,
- Newton's method for solving systems of nonlinear equations,

– fast discrete Fourier transform,
– minimum spanning tree based clustering algorithm,
– k-means clustering algorithm,
– and other algorithms.

Each algorithm is accompanied by links to well-known books that explain it. Thus, students have a reliable source of information. At the same time, both a student and a teacher know exactly which algorithm should be described.

Despite the apparent simplicity, this assignment contains many hidden difficulties. It is far from easy sometimes to answer even the most basic questions that arise, for example:

– What does it mean to "distinguish and describe the full resource of algorithm parallelism"?
– How to show possible ways of parallel execution?
– What does it mean to "define and show the information structure of an algorithm"?
– How to show the macrostructure of the algorithm?
– By what signs should the computing core of the algorithm be distinguished?
– and other.

More details about the features of this practical assignments can be found in the paper [19]. As a result of completing the assignment for each of the 30 proposed algorithms, several descriptions were obtained within the AlgoWiki encyclopedia, using the data of which experienced experts could make full-fledged articles on these algorithms.

Table 1 shows the results of the evaluation of student practical assignments. Most students received positive assessments, a fairly high average mark (4.03) also characterizes the overall high quality of work.

Table 1. Final assessments (2016)

Year	2016
Groups (students)	145 (246)
5 (Excellent)	59 (41%)
4 (Good)	36 (25%)
3 (Satisfactory)	48 (33%)
2 (Unsatisfactory)	2 (1%)
Average mark	4.03

2.2 Practical Assignments on Scalability Study (2017–2019)

Studying the scalability of algorithms and their implementations on various computing platforms when changing the size of the task and the number of processors.

Students needed to conduct a series of computational experiments, collect the necessary data, build graphs with dependencies, interpret, and draw conclusions about scalability. In [20] scalability is defined as a property of a parallel application that describes the dependency of changes in the full range of dynamic characteristics for that program on the full range of its startup parameters. In these assignments, students investigated the dependence of one dynamic characteristic (execution time or performance metric) on two parameters—the number of processes involved and the size of the problem to be solved. For graph problems, the size was determined by the number of vertices, and for matrix ones, by the linear size of the matrix. An example of a graph similar to which it was required to obtain can be seen in Fig. 2.

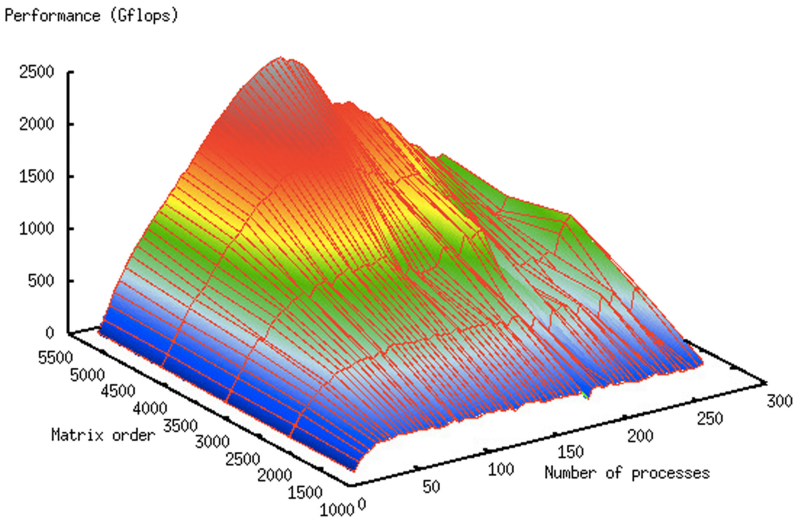

Fig. 2. The scalability of the Cholesky decomposition

In 2017 and 2019, the object of the study were algorithms solving graph problems such as:

- single source shortest path (SSSP),
- breadth-first search (BFS),
- Page Rank,
- construction of the minimum spanning tree (MST),
- finding the strongly connected components (SCC).

For each problem, several algorithms were considered, for example, for the single source shortest path problem, these are the algorithms:

- Bellman-Ford algorithm,
- Dijkstra's algorithm,
- Δ-stepping algorithm.

For each algorithm, up to 5 different ready-made implementations were also provided.

Each student had to experiment with two types of graphs: R-MAT [21] and SSCA2 [22]. Graph generators were provided.

The resulting task for students suggested that for both types of graphs (RMAT and SSCA2) and the selected chain "Problem → Algorithm → Implementation → Computer Platform" [23] was necessary:

- determine the dependence of performance (TEPS) on the number of processors (cores) and graph size;
- find a combination of the number of processors and the size of the graph on which maximum performance is achieved.

Particular attention was required to be paid to large graphs (2^{25} – 2^{28} vertices), since this corresponds to real practical applications, and also in order to avoid the influence of cache memory.

In 2018, the object of the study were linear algebra algorithms:

- SLAU solution,
- eigenvalue calculation,
- eigenvector calculation,
- calculation of singular values,
- LU factorization,
- QR factorization,
- LQ factorization,
- QL factorization,
- RQ factorization,
- and others.

For each algorithm, different ready-made implementations from ScaLAPACK [24] and Magma [25] libraries were provided.

As computing platforms, the SMP nodes of all computer platforms available at the RCC of Moscow State University were considered.

As a result of the practical assignments, students obtained filled tables with the running times of various versions for the chosen algorithm implementations. Some students got the same version of the assignment to complete, which led to an element of competition, which is always useful. Many students began to wonder why their results were worse than others, looked for the reasons for this, and tried to improve something. In the search for the correspondence of the architecture and the implemented algorithm, even ideas of supercomputer co-design [26] were appeared.

This version of the practical assignments requires serious computer resources to get all the necessary performance data for a large number of parallel program runs:

- various sizes of graphs or matrices,
- different numbers of processors,
- various input options, for example, two types of graphs: RMAT and SSCA2,

– for each point on the graph, it is necessary to conduct several experiments in order to remove random fluctuations (artifacts) in performance.

When studying scalability, it is important to draw students' attention to the need to explain all the special points on the graphs: peaks, inflection points, asymptotes, and others. A detailed analysis is complicated, but all the features of the behavior of the algorithm (implementation) must be connected with the properties of the architecture of the computing system.

Table 2 shows the results of the evaluation of student practical assignments. The reason for obtaining very high marks was the need for repeated finishing work in the presence of shortcomings noted by teachers.

Table 2. Final assessments (2017, 2019)

Year	2017	2019
Groups (students)	143 (207)	155
5 (Excellent)	121 (85%)	130 (84%)
4 (Good)	15 (10%)	12 (7.7%)
3 (Satisfactory)	5 (3%)	4 (2.6%)
2 (Unsatisfactory)	2 (1%)	9 (5.8%)
Average mark	4.78	4.7

In 2018, the grading scale for the assignment was more differentiated, including putting down assessments with minuses. The results are shown in Table 3. It can be seen here that a more differentiated approach allows more flexible assessment of the remaining shortcomings of student work.

Table 3. Final assessments (2018)

Students	176
5	25 (14%)
5–	79 (45%)
4	33 (19%)
4–	22 (12.5%)
3	14 (8%)
3–	3 (1.5%)

In general, it is clearly seen that the overall level of assessments for these practical tasks is significantly higher than the level of assessments received in 2016 (see Sect. 2.1). This is due to a noticeably simpler formulation of the task.

The results obtained from these practical assignments are used to populate the database of the project for creating the Algo500 system [27].

2.3 Practical Assignments on the Information Structure of Algorithms and Programs (2019)

Identify and analyze the information structure of the algorithm or a piece of program code.

When preparing the assignment, despite the simple formulation, it was required to find the answer to a number of difficult questions, such as: "How to build the information structure of the algorithm?", "How to depict the information structure of the algorithm?" (the question only at first glance seems simple), "If we know (see) the information structure, how to describe potential parallelism?" etc.

The basis of the analysis of the properties of programs and algorithms is the analysis of the information structure, which is determined by the presence of information dependencies in the analyzed fragment. An informational dependency [28] between two operations exists if the second of them uses what was calculated in the first. Based on the concept of information dependence, an information graph can be constructed using an algorithm or fragment which is an oriented acyclic multigraph whose vertices correspond to the operations of the algorithm, and arcs correspond to information dependencies between them. The information graph is used as a convenient representation of the algorithm in the study of its structure, parallelism resource, as well as other properties.

Our first task was to distribute fragments of the studied program code for more than 200 students. For this, a special parameterized program fragment was constructed.

```
for(i = 1; i <= n; ++i)
   C[i] = C[i+l8] * e;
for(i = 1; i <= n; ++i)
   for(j = 1; j <= n; ++j)
      B[i][j] = B[i+l6][j+l7] + (1-l5)*C[i];
for(i = 1; i <= n; ++i)
   for(j = 1; j <= n; ++j){
      for(k = 1; k <= n; ++k)
         A[i][j][k] = A[i][j][k] + l4*A[i+l1][j+l2][k+l3] +
                                   (1-l4)*A[i+l1][j+l2][n];
      A[i][j][n] = A[i][j][n] + l5*B[i][j];
   }
```

A specific variant for each student was given by a set of parameter values $l1, \ldots, l8$. Some of these parameters took only the values 0 or 1, which corresponds to the presence or absence of an information dependence for the corresponding measurement, other parameters could take other values to specify different types of dependencies. In total, the number of variants exceeded the number of students in the course, so each student received an individual option for research.

In addition to constructing an information graph, students were required to investigate a number of properties:

- The number of vertices in the information graph of a fragment (sequential complexity).
- The length of the critical path in the information graph (parallel complexity).
- The width of the level parallel form (with explanations for which specific LPF the value of the width is given).
- Maximum depth of nesting cycles.
- The number of different types of arcs (the type of arcs is determined by the direction vector and length).
- The presence of long arcs (i.e. arcs whose length depends on external parameters).
- The number of regularity areas in the information graph. A region of regularity is a set of vertices of a graph from which arcs of the same type come from (arcs of different types can come from one region of regularity).

After the study, it was necessary to mark parallel loops of the given program fragment using the OpenMP directive `#pragma omp parallel for` [29].

Students were not provided with a ready-made tool for constructing and analyzing an informational graph of a fragment. Therefore, the reports accepted any images of information graphs—both those built in any graphics editor, as well as hand-drawn on a sheet of paper and scanned. The main thing was to correctly display the information structure of the fragment under consideration and determine its properties. Not all students did this right away; some had to redo their reports several times to achieve an acceptable result. Figure 3 shows the information graph of one of the variants, obtained using the AlgoView [30,31] software tool.

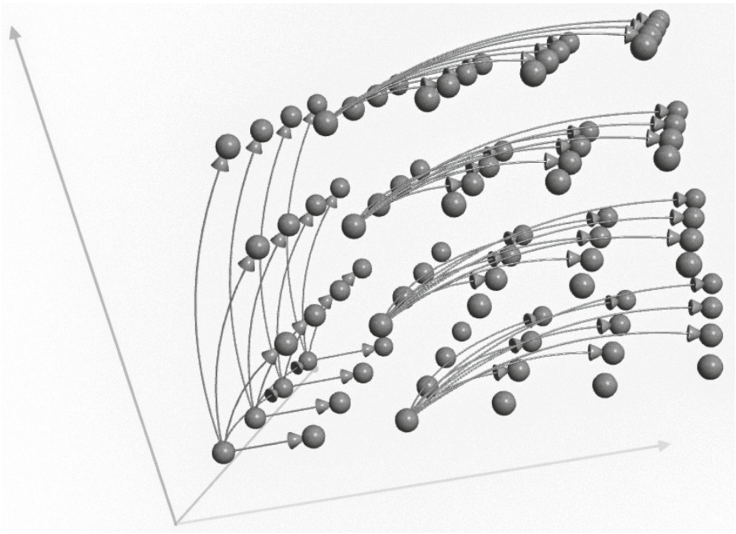

Fig. 3. Information graph of one variant

These assignments were carried out by 183 students, the results are shown in Table 4.

Table 4. Final assessments (2019)

Year	2019
Students	183
5 (Excellent)	168 (91.8%)
4 (Good)	9 (4.9%)
3 (Satisfactory)	3 (1.6%)
2 (Unsatisfactory)	3 (1.6%)
Average mark	4.87

3 Conclusions

Thus, as a part of the course "Supercomputing Simulation and Technologies", the design of a system of assignments aimed at developing skills of efficient use of parallel computing systems is carried out. Different options for practical assignments had varying complexity and were aimed at developing different practical skills. Multiple iterations students-teachers and bringing the report on the practical assignments to a decent state requires students to learn the concepts deeply and thoroughly. The use of modern supercomputer technology prepares students for future work in various areas of the modern science, commerce and industry. The considered system of practical assignments in its current form has proved to be useful, and we are going to develop it further in the framework of this course.

Acknowledgements. The results were obtained in Lomonosov Moscow State University with the financial support of the Russian Science Foundation (agreement N 20-11-20194). The research is carried out using the equipment of the shared research facilities of HPC computing resources at Lomonosov Moscow State University.

References

1. Computer Science Curricula (2013). http://ai.stanford.edu/users/sahami/CS2013
2. NSF/IEEE-TCPP Curriculum Initiative on Parallel and Distributed Computing. http://tcpp.cs.gsu.edu/curriculum/
3. SIAM: Graduate Education for Computational Science and Engineering. SIAM Working Group on CSE Education (2014). http://www.siam.org/students/resources/report.php
4. Future Directions in CSE Education and Research. Report from a Workshop Sponsored by the Society for Industrial and Applied Mathematics (SIAM) and the European Exascale Software Initiative (EESI-2). http://wiki.siam.org/siag-cse/images/siag-cse/f/ff/CSE-report-draft-Mar2015.pdf

5. Prasad, S.K., et al.: NSF/IEEE-TCPP Curriculum Initiative on Parallel and Distributed Computing – Core Topics for Undergraduates, Version I, p. 55 (2012). http://www.cs.gsu.edu/~tcpp/curriculum
6. Voevodin, V., Gergel, V.: Supercomputing education: the third pillar of HPC. Comput. Methods Softw. Dev. New Comput. Technol. **11**(2), 117–122 (2010)
7. Gergel, V., Liniov, A., Meyerov, I., Sysoyev, A.: NSF/IEEE-TCPP curriculum implementation at the state University of Nizhni Novgorod. In: Proceedings of the International Parallel and Distributed Processing Symposium, IPDPS, vol. 6969501, pp. 1079–1084 (2014). https://doi.org/10.1109/IPDPSW.2014.128
8. Voevodin, Vl.V., Gergel, V.P., Popova, N.N.: Challenges of a systematic approach to parallel computing and supercomputing education. In: Lecture Notes in Computer Science, vol. 9523, pp. 90–101 (2015). https://doi.org/10.1007/978-3-319-27308-2_8
9. Rüde, U., Willcox, K., McInnes, L.C., de Sterck, H.: Research and education in computational science and engineering. SIAM Rev. **60**(3), 707–754 (2016). https://doi.org/10.1137/16M1096840
10. Meyerov, I., Bastrakov, S., Barkalov, K., Sysoyev, A., Gergel, V.: Parallel numerical methods course for future scientists and engineers. Commun. Comput. Inf. Sci. **793**, 3–13 (2017). https://doi.org/10.1007/978-3-319-71255-0_1
11. Prasad, S.K., Gupta, A., Rosenberg, A., Sussman, A., Weems, C. (eds.): Topics in Parallel and Distributed Computing. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-93109-8
12. Ghafoor, S., Brown, D.W., Rogers, M.: Integrating parallel computing in introductory programming classes: an experience and lessons learned. In: Heras, D., et al. (eds.) Euro-Par 2017: Parallel Processing Workshops. Euro-Par 2017. Lecture Notes in Computer Science, vol. 10659, pp. 216–226. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-75178-8_18
13. Meyerov, I., Bastrakov, S., Sysoyev, A., Gergel, V.: Comprehensive collection of time-consuming problems for intensive training on high performance computing. Commun. Comput. Inf. Sci. **965**, 523–530 (2019). https://doi.org/10.1007/978-3-030-05807-4_44
14. Voevodin, V., et al.: Supercomputer Lomonosov-2: large scale, deep monitoring and fine analytics for the user community. Supercomput. Front. Innov. **6**(2), 4–11 (2019). https://doi.org/10.14529/jsfi190201
15. Sadovnichy, V., Tikhonravov, A., Voevodin, V., Opanasenko, V.: Lomonosov: supercomputing at Moscow State University. In: Contemporary High Performance Computing: From Petascale toward Exascale, ser, pp. 283–307. Chapman & Hall/CRC Computational Science, Boca Raton (2013)
16. Open Encyclopedia of Parallel Algorithmic Features. http://algowiki-project.org
17. Voevodin, V., Antonov, A., Dongarra, J.: AlgoWiki: an open encyclopedia of parallel algorithmic features. Supercomput. Front. Innov. **1**(2), 4–18 (2015). https://doi.org/10.14529/jsfi150101
18. Antonov, A., Voevodin, Vad., Voevodin, Vl., Teplov, A.: A study of the dynamic characteristics of software implementation as an essential part for a universal description of algorithm properties. In: 24th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing Proceedings, 17th–19th February 2016, pp. 359–363 (2016). https://doi.org/10.1109/PDP.2016.24
19. Antonov, A.S., Voevodin, V.V., Popova, N.N.: Parallel structure of algorithms and training computational technology specialists. J. Phys. Conf. Ser. **1202** (2019). https://doi.org/10.1088/1742-6596/1202/1/012021

20. Antonov, A., Teplov, A.: Generalized approach to scalability analysis of parallel applications. Lect. Notes Comput. Sci. **10049**, 291–304 (2016). https://doi.org/10.1007/978-3-319-49956-7_23
21. Chakrabarti, D., Zhan, Y., Faloutsos, C.: R-MAT: a recursive model for graph mining (2006). http://www.cs.cmu.edu/~christos/PUBLICATIONS/siam04.pdf
22. Bader D.A., Madduri K.: Design and implementation of the HPCS graph analysis benchmark on symmetric multiprocessors. In: Bader, D.A., Parashar, M., Sridhar, V., Prasanna, V.K. (eds) High Performance Computing – HiPC 2005. Lecture Notes in Computer Science, vol. 3769. Springer, Heidelberg (2005). https://doi.org/10.1007/11602569_48
23. Antonov, A., Frolov, A., Konshin, I., Voevodin, V.: Hierarchical domain representation in the AlgoWiki encyclopedia: from problems to implementations. Commun. Comput. Inf. Sci. **910**, 3–15 (2018). https://doi.org/10.1007/978-3-319-99673-8_1
24. ScaLAPACK – Scalable Linear Algebra PACKage. http://www.netlib.org/scalapack/
25. Matrix Algebra on GPU and Multicore Architectures. http://icl.eecs.utk.edu/magma/
26. Dosanjh, S.S., Barrett, R.F., Doerfler, D.W., Hammond, S.D., et al.: Exascale design space exploration and co-design. Fut. Gener. Comput. Syste. **30**, 46–58 (2014). https://doi.org/10.1016/j.future.2013.04.018
27. Antonov, A., Nikitenko, D., Voevodin, V.: Algo500 - a new approach to the joint analysis of algorithms and computers. Lobachevskii J. Math. **8**(41), 1435–1443 (2020). Special issue "Supercomputing Applications, Algorithms and Software Tools"
28. Voevodin, V., Voevodin, Vl.: Parallel Computing, p. 608. BHV-Petersburg, St. Petersburg (2002)
29. The OpenMP API specification for parallel programming. https://www.openmp.org/
30. Antonov, A.S., Volkov, N.I.: An AlgoView web-visualization system for the AlgoWiki project. Commun. Comput. Inf. Sci. **753**, 3–13 (2017). https://doi.org/10.1007/978-3-319-67035-5_1
31. Antonov, A., Volkov, N.: Interactive 3D representation as a method of investigating information graph features. In: Russian Supercomputing Days: Proceedings of the international conference, 24–25 September 2018, pp. 262–273. Moscow State University, Moscow (2018). https://doi.org/10.1007/978-3-030-05807-4_50