Using the AlgoWiki Open Encyclopedia of Parallel Algorithmic Features in HPC Education

Alexander Antonov(✉) and Vladimir Voevodin

Lomonosov Moscow State University, Moscow, Russia
{asa,voevodin}@parallel.ru

Abstract. AlgoWiki is an open encyclopedia of algorithm properties and their implementations on various hardware and software platforms. It can be used for various purposes, such as finding the optimal algorithm for addressing a particular problem, analyzing the information structure of an application, or comparing the efficiency of different implementations of the same algorithm. In this article, we consider several options for using AlgoWiki to illustrate various concepts and notions in the Body of Knowledge and Skills for parallel computation and supercomputer technologies. The approbation of these options is conducted as part of the courses "Supercomputers and Parallel Data Processing" and "Supercomputer Simulation and Technologies", taught in the Computational Mathematics and Cybernetics Department at Lomonosov Moscow State University.

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

1 Introduction

The demand for parallel computation is constantly growing. Technologies originally developed for supercomputers are becoming relevant for any computing devices, including mobile ones. This explains the growing need to train specialists in the field of parallel computation and supercomputer technologies. Software development in this field is quite active all around the world [1–5] and in Russia [6,7]. Interesting discussions on these issues are annually held within the framework of international seminars: EduHPC [8], EduPAR [9], Euro-EduPAR [10].

The Supercomputing Consortium of Russian Universities was established in Russia in 2008 [11]. The goal of the Consortium was to develop and implement a set of activities aimed at efficiently using the available higher-education capacities to develop and implement supercomputer technologies in Russian education, science, and industry. Since this moment, the Supercomputer Consortium has coordinated the development of educational programs for high-performance

L. Sokolinsky and M. Zymbler (Eds.): PCT 2019, CCIS 1063, pp. 77–90, 2019.
https://doi.org/10.1007/978-3-030-28163-2_6

computing in Russia. As of the end of 2018, the Supercomputing Consortium of Russian Universities features 48 permanent and 16 associate members.

From 2010 to 2012, the members of the Supercomputer Consortium played an active role in implementing a large-scale project for the President's Committee on Modernization and Technological Development of the Russian Economy called "Building a system to train highly skilled staff on supercomputing technologies and specialized software" ("Supercomputing Education"). The project involved dozens of Russian universities. The project featured a major body of work for developing a holistic supercomputer education system in Russia [12–14].

2 The Body of Knowledge and Skills

Part of the "Supercomputer Education" project was dedicated to building a Summary List (Body) of knowledge and skills (professional competencies) in the field of parallel computation and supercomputing technologies [15]. The Body describes the competencies in supercomputer education that students must possess once they graduate from the respective college or complete the corresponding training course, refresher course, or specialized training as part of special groups. The Body of Knowledge describes the structure and basis of the education process.

The main content of the Body of Knowledge is a description of the subject area of "Supercomputers and parallel computation," which clearly defines what should be taught and how the education process should be organized for each specific group of students. The structure of the Body of Knowledge follows the recommendations of such international professional communities as the Association for Computing Machinery (ACM) and the IEEE Computer Society [16,17]. The Body of Knowledge is a set of five knowledge areas representing different aspects of the chosen specialization. The areas are divided into smaller units called sections which contain individual thematic modules within each respective area. Each section, in its turn, consists of a set of topics which represent the lowest level in the hierarchy for the given specialization.

The Body of Knowledge was the basis for a system for certifying training courses and programs on "Supercomputers and parallel computation" [18] and for the implementation of measures allowing to verify the aggregate content of training courses adopted in Russian universities for training, refresher training, and continuing education aimed at personnel working in the field of supercomputing technologies and parallel computation.

3 AlgoWiki, the Open Encyclopedia of Parallel Algorithmic Features

The AlgoWiki Open Encyclopedia of Parallel Algorithmic Features [19] has been developed since 2014 as an online project based on wiki-technologies. This project allows the entire computing community to participate in the description of algorithm properties. The project started with a universal structure for describing algorithm properties, which, in turn, enabled a common structure for explaining any computational algorithm [20,21]. When describing algorithms according to this structure, special emphasis is placed on their parallelism properties [22,23]. The description is divided into two parts. The first part describes the properties of an actual algorithm which are independent of the hardware and software platforms on which it will be implemented. The second part describes the properties of specific implementations of a given algorithm: both serial and parallel, intended for different hardware and software platforms [24].

Once the number of algorithms described [25] reached a certain limit, we needed to bring some order to this information. Since algorithms have no value in and of themselves but are needed to address specific tasks, the AlgoWiki Encyclopedia was complemented with a level of description related to the tasks being addressed. Each task can be addressed in various ways, so another level of description appeared related to the methods for addressing tasks. At the task and method levels, the descriptions can be prepared in an arbitrary manner; introducing a unified description structure for these levels can be the subject of further study for the project. Each method can be implemented via several algorithms, and each algorithm can have several implementations. Thus, it was natural to represent the subject area in a hierarchical form: a chain from task to method to algorithm to implementation [26].

Since the second part of an algorithm description gathers data, among other things, on execution results of the algorithm on various hardware and software platforms, we decided to use this data to conduct various comparisons. In this case, it is possible to perform comparisons using traditional computing platform ratings, similar to well-known supercomputer ratings (such as Top500, HPCG [27], Graph500 [28], and others), and to compare different implementations, algorithms, methods and tasks from the standpoint of their suitability for various hardware and software platforms [29]. The AlgoWiki Encyclopedia is working to implement "architectural profiles", which will allow subsets of tasks, methods, algorithms, and implementations to be singled out from the multitude of descriptions that best correspond to a specific type of software and hardware platforms.

4 The AlgoWiki Encyclopedia and the Body of Knowledge and Skills

When developing a training course on parallel computation and supercomputing technologies, it is important to take into account how training materials correspond to the Body of Knowledge and Skills. Many points in the Body of Knowledge must be illustrated with examples, and the AlgoWiki Open Encyclopedia of Parallel Algorithmic Features can be a great source of such examples. AlgoWiki furnishes plenty of useful information on all five areas of knowledge described by the Body of Knowledge.

Let us consider some examples of how the AlgoWiki Encyclopedia can be used to illustrate various concepts and notions from the Body of Knowledge and

Skills. However, it should be noted that the practical use of AlgoWiki is much wider and not limited to these few examples.

4.1 Knowledge Area 1. The Mathematical Basics of Parallel Computations

Section 1.1. Graph Models of Programs. AlgoWiki places special emphasis on the study of information graphs since these elements contain all of the information on an algorithm's information structure. An information graph is an oriented acyclic graph in which the vertices correspond to the operations of the algorithm, while the edges correspond to the actual information dependencies between them. Figure 1 shows an example of an information graph for an algorithm that multiplies a dense matrix by a vector, displaying the input and output data.

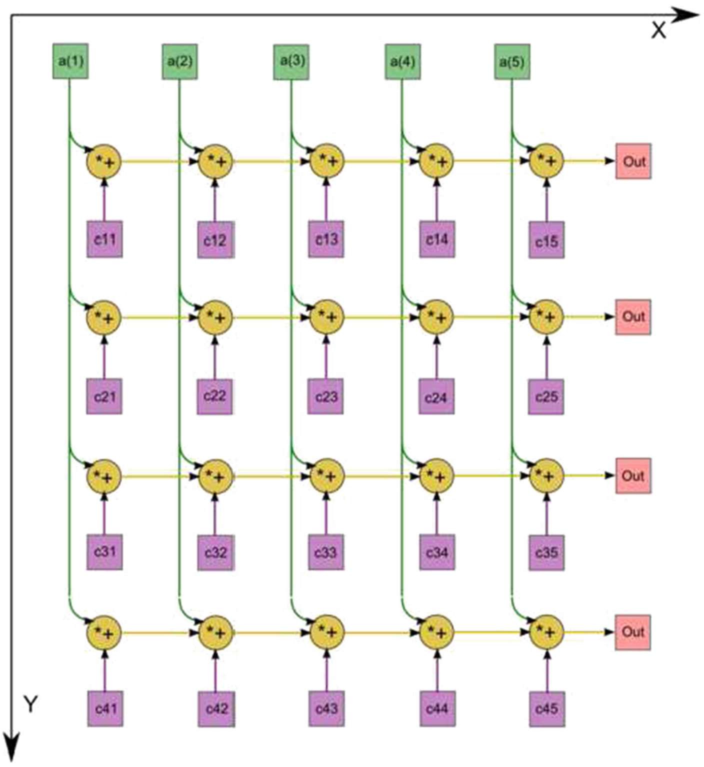


Fig. 1. Information graph for the multiplication of a dense matrix by a vector, displaying the input and output data

To study the parallelism resource of an algorithm, it may be necessary to conduct a study of its *parallel form*. Figure 1 is a representation of an information graph in which all of the vertices are partitioned into numbered subsets of levels, the initial point of each edge is located at a lower-numbered layer than the final point, and there may not be any edges between vertices of the same level. The parallel form can be visualized on an ordinary information graph by marking up the vertices belonging to different levels. The AlgoView interactive 3D visualization system for information graphs [30,31], created under the AlgoWiki project, assists in visualizing the multilevel structure step by step. In this structure, the current step, previous steps, and subsequent steps of an algorithm are displayed in different colors. An example of the obtained image is given in Fig. 2.

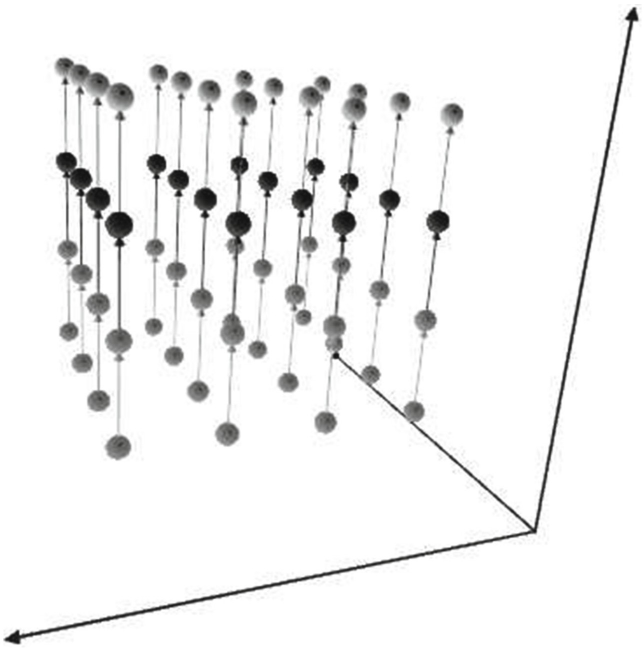


Fig. 2. Parallel form of an information graph for a dense-matrix multiplication algorithm

Section 1.2. Unlimited Parallelism Concept. The concept of unlimited parallelism stipulates that an algorithm is implemented on a parallel computing platform that does not impose any restrictions on its execution. There can be an infinite number of processors, they are all universal, working synchronously, having shared memory, and all information transfers are performed instantly and without conflicts [32].

Algorithm descriptions in AlgoWiki pay particular attention to the notion of parallel complexity, which is understood as the number of steps needed to execute the algorithm given an infinite number of processors (functional devices, computing nodes, cores, etc.), i.e. within the concept of unlimited parallelism. The parallel complexity of an algorithm is understood as the height of the canonical parallel form. Parallel complexity enables comparison of the parallelism resources of various algorithms. For example, the parallel complexity of an algorithm for summing vector elements by pairwise summation equals $\log_2 n$, the parallel complexity of a fast Fourier transform (Cooley–Tukey algorithm) for vectors with lengths equal to a power of two is $2\log_2 n$, and the parallel complexity for dense-matrix multiplication is $2n^3$.

Section 1.3. Fine Information Structure of Programs. The AlgoWiki Encyclopedia presents an extensive array of sample studies of parallel structure for various programs. They include all key *types of parallelism*:

– *finite parallelism*, determined by the information independence of certain fragments in the body of the program;
– *massive parallelism*, determined by the information independence of iterations within the program;
– *coordinate parallelism*, in which information-independent vertices lie in hyperplanes perpendicular to one of the coordinate axes;
– *skewed parallelism*, i.e. parallelism within the iteration space determined by the surfaces of unfolded levels.

Section 1.4. Equivalent Transformations of Programs. Once the parallelism resource of a program fragment under investigation has been determined, the optimum usage option can be achieved by applying *equivalent transformations*, which are transformations of the code that preserve the exact result of the program execution. Since most parallelism resources are usually concentrated in loop structures, the most popular option for equivalent transformation is *elementary loop transformation*. Many examples of this transformation can be found in various algorithm implementations given in the AlgoWiki Encyclopedia. For example, one can study the efficiency of all possible loop interchanges in the main loop nest of the dense-matrix multiplication algorithm.

On the other hand, the available parallelism resources sometimes cannot be used if only equivalent transformations are applied. In certain occasions, *non-equivalent program transformations* can also be used. It should be noted, however, that the calculated results can vary in this case, for example, due to the accumulation of rounding errors in a different order of operation execution. A classic example of non-equivalent transformation for purposes of parallelization is summing arrays by pairwise summation, as per the information graph presented in Fig. 3.

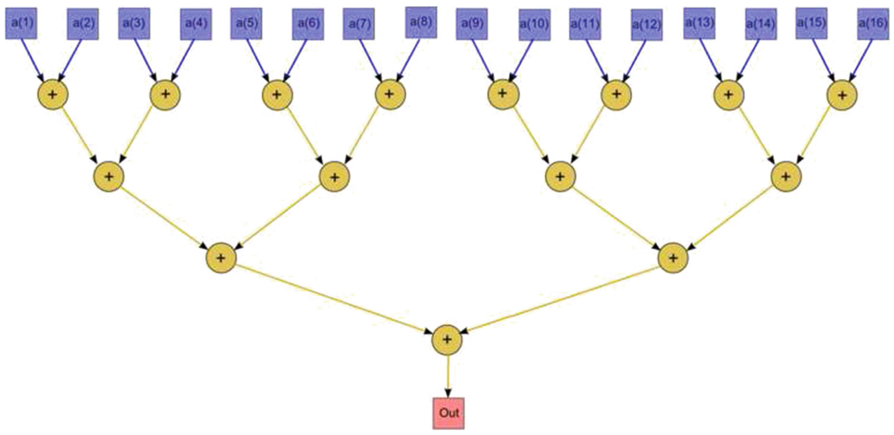


Fig. 3. Information graph for array summation by the pairwise summation algorithm

4.2 Knowledge Area 2. Parallel Computing Systems (Computing Basics)

Section 2.2. The Basics of Building Computing Systems. The AlgoWiki Encyclopedia does not directly include any information on building computer systems. However, for many topics related to using various types of devices (*scalar, vector, and pipeline functional units*, *accelerators*, *graphical processors*, *programmable logic processors (FPGA)*, *specialized processors*), illustrations can be found in AlgoWiki in the form of algorithms and most suitable implementations for using them on a given type of device. The corresponding search can substantially simplify the "architectural profiles" mechanism currently being developed.

Traditional *memory access properties* are related to the computing system properties and are not directly explained in AlgoWiki either. However, a memory access property such as memory locality is determined by the properties of the algorithm and its implementation. This is why significant attention has to be paid to studying the locality of memory calls. Spatial and temporal localities are studied for all algorithms. Spatial locality is understood as the average distance between several consecutive memory calls. Temporal locality shows the average number of calls to the same memory address over the entire program execution time. Studying locality requires building memory access profiles for processing cores and key fragments. Figure 4 compares memory access profiles for six loop order options in the main computational core of the dense-matrix multiplication algorithm. Differences in the profiles determine different memory access properties, which in this case substantially affect the performance of the options under consideration. By analyzing the profiles, it is easy to explain why the fragments with the innermost loop of j show the highest performance in most modern architectures.

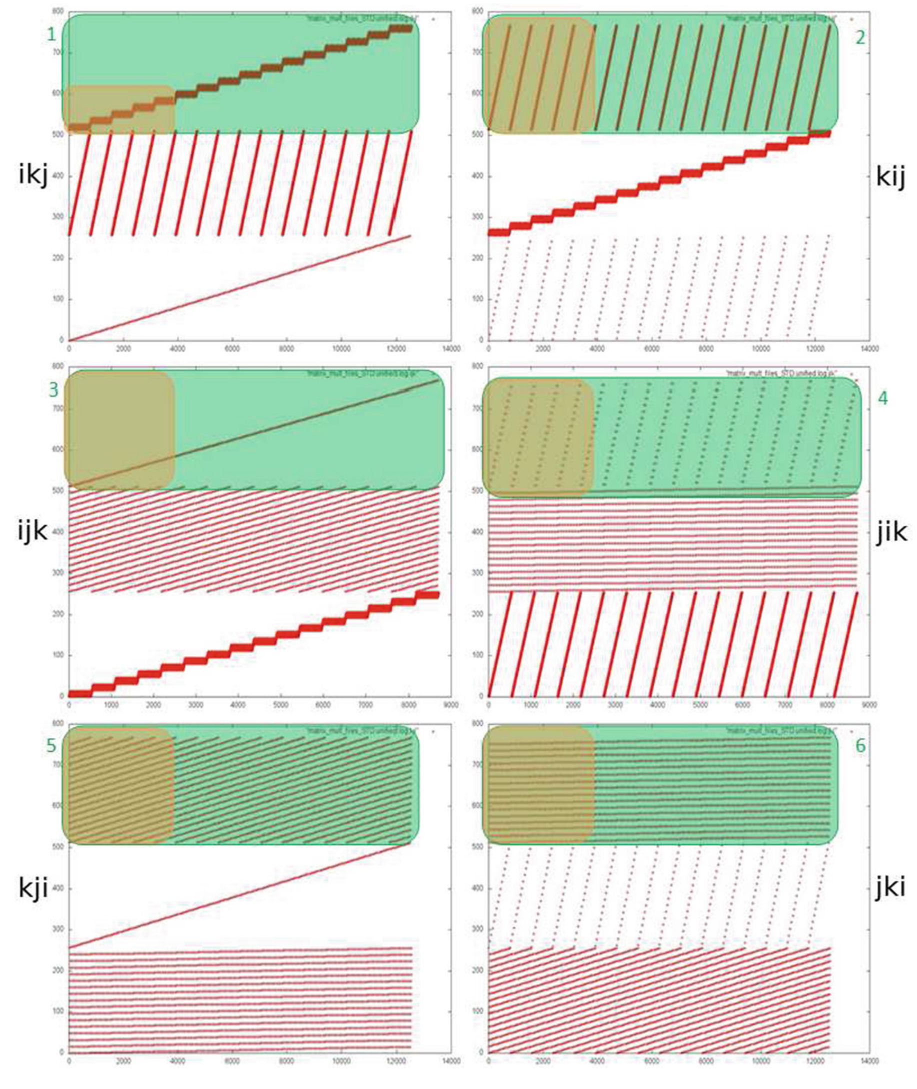
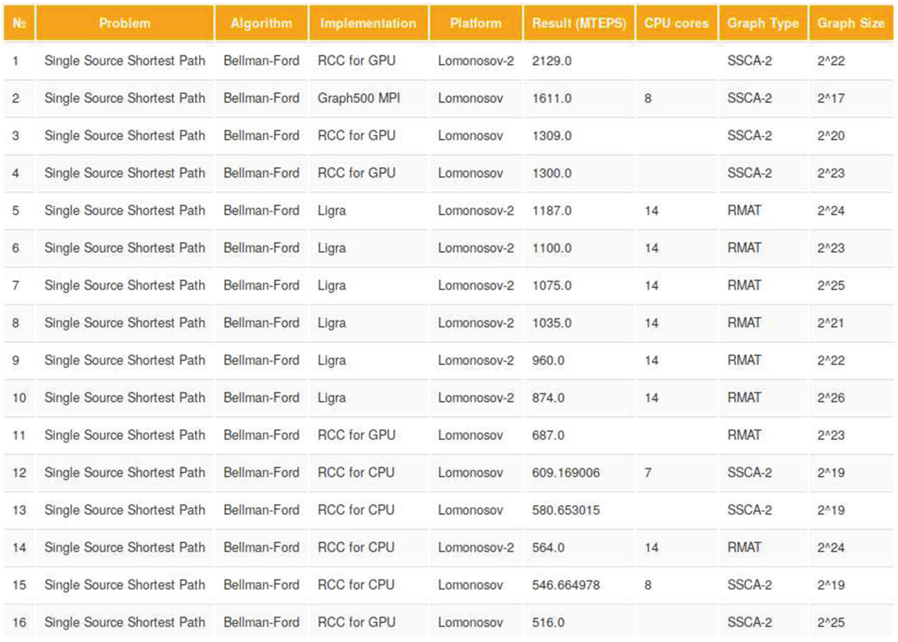


Fig. 4. Memory access profiles for six loop order options in the main computational core of the dense-matrix multiplication algorithm

Sections 2.4 ("Multi-CPU computing systems"), 2.5 ("Multi-CPU computing systems with shared memory"), 2.6 ("Multi-CPU computing systems with distributed memory"), and 2.7 ("Graphics Processing Units") describe different computing system properties, including those concepts that can be illustrated with AlgoWiki materials, such as *efficiency*, *performance*, *speedup*, and *scalability*. Illustrations for sections related to specific supercomputer architecture types can be found using the "architectural profiles" mechanism.

To illustrate the concept of *performance benchmarks*, the AlgoWiki Encyclopedia already describes the opportunity for using as a benchmark any implementation of any algorithm described. After obtaining data on the benchmark performance for a set of computing systems, it is possible to build ratings of these systems, either for a specific implementation or for an algorithm, a method, or even a task. Figure 5 shows an example rating for the Bellman–Ford algorithm [33–35], which computes the shortest possible path in a graph. The rating was built taking into account performance data from several implementations of this algorithm on the MSU Lomonosov and Lomonosov-2 supercomputers [36].



№	Problem	Algorithm	Implementation	Platform	Result (MTEPS)	CPU cores	Graph Type	Graph Size
1	Single Source Shortest Path	Bellman-Ford	RCC for GPU	Lomonosov-2	2129.0		SSCA-2	2^22
2	Single Source Shortest Path	Bellman-Ford	Graph500 MPI	Lomonosov	1611.0	8	SSCA-2	2^17
3	Single Source Shortest Path	Bellman-Ford	RCC for GPU	Lomonosov	1309.0		SSCA-2	2^20
4	Single Source Shortest Path	Bellman-Ford	RCC for GPU	Lomonosov	1300.0		SSCA-2	2^23
5	Single Source Shortest Path	Bellman-Ford	Ligra	Lomonosov-2	1187.0	14	RMAT	2^24
6	Single Source Shortest Path	Bellman-Ford	Ligra	Lomonosov-2	1100.0	14	RMAT	2^23
7	Single Source Shortest Path	Bellman-Ford	Ligra	Lomonosov-2	1075.0	14	RMAT	2^25
8	Single Source Shortest Path	Bellman-Ford	Ligra	Lomonosov-2	1035.0	14	RMAT	2^21
9	Single Source Shortest Path	Bellman-Ford	Ligra	Lomonosov-2	960.0	14	RMAT	2^22
10	Single Source Shortest Path	Bellman-Ford	Ligra	Lomonosov-2	874.0	14	RMAT	2^26
11	Single Source Shortest Path	Bellman-Ford	RCC for GPU	Lomonosov	687.0		RMAT	2^23
12	Single Source Shortest Path	Bellman-Ford	RCC for CPU	Lomonosov	609.169006	7	SSCA-2	2^19
13	Single Source Shortest Path	Bellman-Ford	RCC for CPU	Lomonosov	580.653015		SSCA-2	2^19
14	Single Source Shortest Path	Bellman-Ford	RCC for CPU	Lomonosov-2	564.0	14	RMAT	2^24
15	Single Source Shortest Path	Bellman-Ford	RCC for CPU	Lomonosov	546.664978	8	SSCA-2	2^19
16	Single Source Shortest Path	Bellman-Ford	RCC for GPU	Lomonosov	516.0		SSCA-2	2^25

Fig. 5. Ratings built for the Bellman–Ford algorithm

The *Linpack benchmark* algorithm [37] is also described in AlgoWiki, and this description can provide extensive material for studying the algorithm in detail if necessary.

Section 2.8. Computing Systems of Trans-PetaFLOPS and ExaFLOPS Performance. The *Top500* and *Top50* ratings of top-performance systems are based on Linpack benchmark results. However, the topic can be substantially complemented using the AlgoWiki Encyclopedia, not just by employing algorithm descriptions used by other known ratings (e.g., HPCG [38]), but also by applying the Top500 rating methodology to any other algorithm.

4.3 Knowledge Area 3. Parallel Programming Technologies (Basics of Software Engineering)

Section 3.1. General Parallel Application Development Principles. The *parallelization of serial programs* is one of the key objectives in the field of parallel computation. Parallelization must be based on a study of the parallelism resource and other properties of the algorithm and a consideration of the peculiarities of the target software and hardware platforms. In addition to a study of algorithm properties, the AlgoWiki Encyclopedia provides options for their serial and parallel implementations.

Section 3.3. Methods and Technologies for Parallel Application Development. When studying various *parallel programming technologies*, such as *MPI*, *OpenMP*, *CUDA*, and others, one can use as examples algorithms presented in the AlgoWiki Encyclopedia having implementations that resort to these technologies.

Section 3.4. Parallel Problem-Oriented Libraries and Program Sets. Many algorithms presented in *parallel problem-oriented libraries*, such as *BLAS*, *Lapack*, *Scalapack*, *FFTW*, *PETSc*, *MKL*, and others, are described in the AlgoWiki Encyclopedia. Studying these algorithms can help in selecting the right library functions.

4.4 Knowledge Area 4. Parallel Algorithms for Addressing Tasks

Section 4.1. General Parallel Algorithm Development Principles. The traditional *stages for addressing tasks in parallel computing systems* correspond well with the hierarchical representation of the subject area as a hierarchical form: a chain from task to method to algorithm to implementation, as in AlgoWiki. In this case, the actual "Algorithm Classification" page can act as a good illustration of the concept being studied.

When studying *quality parameters of parallel programs*, such as *speedup*, *efficiency*, and *scalability*, one can use the data provided in the second part of the algorithm description in the AlgoWiki Encyclopedia. For example, the charts provided offer a good illustration of the scalability of algorithm implementations on specific computing systems. By way of illustration, Fig. 6 shows changes in the performance of the Linpack benchmark (HPL implementation) depending on the number of CPUs and the matrix size, obtained from running the test on the Lomonosov supercomputer.

For every algorithm described in AlgoWiki, an assessment of the *computational complexity of serial and parallel algorithms* is provided. The authors show *asymptotic evaluations* which help to understand the algorithm properties when working with large amounts of data.

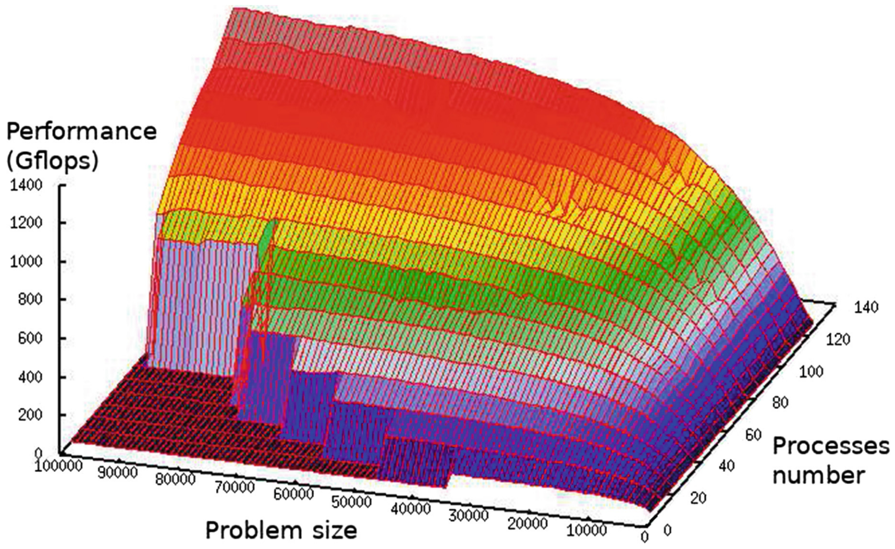


Fig. 6. Performance of the Linpack benchmark (HPL implementation) depending on the number of CPUs and matrix size

Section 4.2. Educational Algorithms for Parallel Programming. Almost all the algorithms in this section are already described in the AlgoWiki Encyclopedia. This can provide exhaustive data for studying the respective algorithms.

Algorithms from sections 4.3 ("Parallel algorithms for matrix computations"), 4.4 ("Parallel algorithms for data search and sorting"), 4.5 ("Parallel algorithms for graph computations"), 4.6 ("Parallel algorithms for solving partial differential equations"), 4.7 ("Parallel algorithms for solving optimization tasks"), 4.8 ("Monte Carlo parallel algorithms"), and 4.9 ("Parallel algorithms for other classes of computationally intensive tasks") are partially represented in the AlgoWiki Encyclopedia. Descriptions of algorithms not currently represented in AlgoWiki can easily be added by any member of the computing community.

4.5 Knowledge Area 5. Parallel Computations, Big Data Tasks, and Specific Subject Areas

The tasks listed in sections 5.1 ("Parallel methods for solving computationally complex tasks in Earth sciences"), 5.2 ("Parallel methods for solving computationally complex tasks in life sciences"), 5.3 ("Parallel methods for solving computationally complex tasks in engineering calculations"), 5.4 ("Parallel methods for solving computationally complex tasks in quantum chemistry"), and 5.5 ("Parallel methods for solving atomistic modeling tasks") can provide extensive material to add to the AlgoWiki Encyclopedia in the future.

5 Conclusions

The AlgoWiki Open Encyclopedia of Parallel Algorithmic Features is a project carried out at Lomonosov Moscow State University. In addition to classic use cases, the AlgoWiki Encyclopedia finds increasingly common use in the education process. Descriptions of methods, algorithms, and implementations from AlgoWiki are used with illustrative purpose in the courses "Supercomputers and Parallel Data Processing" and "Supercomputer Simulation and Technologies", which are taught in the Computational Mathematics and Cybernetics Department at Lomonosov Moscow State University.

In this article, we used AlgoWiki materials to illustrate various concepts of the Body of Knowledge and Skills in the field of parallel computation and supercomputer technologies, a building block in various educational courses.

The benefits of the AlgoWiki Encyclopedia are not limited to illustrative materials for various sections of numerous training courses. Throughout several years, fourth-year students taking the course "Supercomputers and Parallel Data Processing" in the Computational Mathematics and Cybernetics Faculty at Lomonosov Moscow State University have been given a practical assignment that includes algorithm descriptions in the AlgoWiki Encyclopedia. From 2016 to 2018, large-scale practical assignments were organized at the same faculty for second-year Master's degree students as part of the course "Supercomputer Modelling and Technologies" [39]. The assignment was modified slightly each year but always involved studying and describing a series of algorithm properties and implementations. Each practical assignment was given to about 200–250 Master's degree students, and the results were a valuable contribution towards the development of the AlgoWiki Encyclopedia.

The development of the AlgoWiki Open Encyclopedia of Parallel Algorithmic Features continues. In addition to expanding the description database, the authors are working on improving the functions implemented. Future plans include building a rating system for implementations, algorithms, methods, and tasks, as well as introducing "architectural profiles" of the Encyclopedia.

Acknowledgements. The results were obtained with the financial support of the Russian Foundation for Basic Research (grant N. 19-07-01030). The research was carried out on the HPC equipment of the shared research facilities at Lomonosov Moscow State University and was supported through the project RFMEFI62117X0011.

References

1. SIAM: Graduate Education for Computational Science and Engineering. SIAM Working Group on CSE Education (2014). http://www.siam.org/students/resources/report.php
2. 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

3. Prasad, S.K., et al.: NSF/IEEE-TCPP Curriculum Initiative on Parallel and Distributed Computing – Core Topics for Undergraduates, Version I. 55 p. (2012). http://www.cs.gsu.edu/~tcpp/curriculum
4. Qasem, A.: Modules for teaching parallel performance concepts. In: Prasad, S.K., Gupta, A., Rosenberg, A., Sussman, A., Weems, C. (eds.) Topics in Parallel and Distributed Computing, pp. 59–77. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-93109-8_3
5. Ghafoor, S., Brown, D.W., Rogers, M.: Integrating parallel computing in introductory programming classes: an experience and lessons learned. In: Heras, D.B., Bougé, L. (eds.) Euro-Par 2017. LNCS, vol. 10659, pp. 216–226. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-75178-8_18
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. Supercomputing Education in Russia. Supercomputing Consortium of the Russian Universities. Technical report (2012). http://hpc.msu.ru/files/HPC-Education-in-Russia.pdf
8. EduHPC-18: Workshop on Education for High-Performance Computing. https://grid.cs.gsu.edu/~tcpp/curriculum/?q=eduhpc18
9. 8th NSF/TCPP Workshop on Parallel and Distributed Computing Education (EduPar-18). https://grid.cs.gsu.edu/~tcpp/curriculum/?q=edupar18
10. 4th European Workshop on Parallel and Distributed Computing Education for Undergraduate Students (Euro-EDUPAR). http://www.euroedupar.tu-darmstadt.de/
11. HPC Consortium of Russian Universities. http://hpc-russia.ru
12. Voevodin, V., Gergel, V., Sokolinsky, L., Dyomkin, V., Popova, N., Bukhanovskiy, A.: Supercomputing education in Russia: results and perspectives. Vestnik Lobachevsky Univ. Nizhni Novgorod **1**(4), 203–209 (2012)
13. Antonov, A.S., et al.: Supercomputing education project: year 2012. Vestnik Lobachevsky Univ. Nizhni Novgorod **1**(1), 12–16 (2013)
14. Antonov, A.S., Voevodin, Vl.V., Gergel': A systematic approach to supercomputing education. Bull. South Ural State Univ. **2**(2), 5–17 (2013). Series "Computational Mathematics and Informatics"
15. The Body of Knowledge and Skills in Supercomputing Technologies. http://hpc-education.ru/?q=node/15
16. Computer Science Curricula (2013). http://ai.stanford.edu/users/sahami/CS2013
17. Prasad, S.K., Weems, C.C., Dougherty, J.P., Deb, D.: NSF/IEEE-TCPP curriculum initiative on parallel and distributed computing: status report. In: Proceedings of the 49th ACM Technical Symposium on Computer Science Education (SIGCSE 2018), pp. 134–135. ACM, New York (2018). https://doi.org/10.1145/3159450.3159632
18. Antonov, A.S., Voevodin, Vl.V., Odintsov, I.O.: Methodology for certification of training courses and programs in the area of studies "supercomputers and parallel computing". Vestnik Lobachevsky Univ. Nizhni Novgorod **2**(1), 13–18 (2014)
19. Open Encyclopedia of Parallel Algorithmic Features. http://algowiki-project.org
20. Antonov, A.S., Voevodin, Vad.V., Voevodin, Vl.V., Teplov, A.M., Frolov, A.V.: First version of an open encyclopedia of algorithmic features. Vestnik UGATU **19**(2(68)), 150–159 (2015). Series "Management, Computer Engineering and Informatics"
21. Voevodin, V., Antonov, A., Dongarra, J.: AlgoWiki: an open encyclopedia of parallel algorithmic features. Supercomput. Front. Innov. **2**(1), 4–18 (2015)

22. Voevodin, V., Antonov, A., Dongarra, J.: Why is it hard to describe properties of algorithms? Proc. Comput. Sci. **101**, 4–7 (2016). https://doi.org/10.1016/j.procs.2016.11.002
23. Voevodin, Vl., Antonov, A., Voevodin, Vad.: What do we need to know about parallel algorithms and their efficient implementation? In: Prasad, S.K., Gupta, A., Rosenberg, A., Sussman, A., Weems, C. (eds.) Topics in Parallel and Distributed Computing, pp. 23–58. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-93109-8_2
24. 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, 17–19 February, pp. 359–363 (2016)
25. Antonov, A., et al.: Parallel processing model for Cholesky decomposition algorithm in AlgoWiki project. Supercomput. Front. Innov. **3**(3), 61–70 (2016). https://doi.org/10.14529/jsfi160307
26. Antonov, A., Frolov, A., Konshin, I., Voevodin, Vl.: Hierarchical domain representation in the AlgoWiki encyclopedia: from problems to implementations. In: Sokolinsky, L., Zymbler, M. (eds.) PCT 2018. CCIS, vol. 910, pp. 3–15. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-99673-8_1
27. Top500 Supercomputer Sites. https://www.top500.org
28. Graph 500. https://graph500.org
29. Antonov, A., Dongarra, J., Voevodin, Vl.: AlgoWiki project as an extension of the Top500 methodology. Supercomput. Front. Innov. **5**(1), 4–10 (2018). https://doi.org/10.14529/jsfi180101
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: Voevodin, V., Sobolev, S. (eds.) RuSCDays 2018. CCIS, vol. 965, pp. 587–598. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-05807-4_50
32. Voevodin, V., Voevodin, Vl.: Parallel Computing, p. 608. BHV-Petersburg, St. Petersburg (2002)
33. Bellman, R.: On a routing problem. Q. Appl. Math. **16**, 87–90 (1958)
34. Ford, L.R.: Network Flow Theory. Rand.org, RAND Corporation (1958)
35. Moore, E.F.: The shortest path through a maze. In: International Symposium on the Theory of Switching, pp. 285–292 (1959)
36. Sadovnichy, V., Tikhonravov, A., Voevodin, V., Opanasenko, V.: Lomonosov: supercomputing at Moscow state university. In: Contemporary High Performance Computing: From Petascale Toward Exascale. Chapman & Hall/CRC Computational Science, Boca Raton, pp. 283–307 (2013)
37. Dongarra, J.J., Luszczek, P., Petitet, A.: The LINPACK benchmark: past, present and future. Concurr. Comput.: Pract. Exp. **15**(9), 803–820 (2003)
38. Dongarra, J., Heroux, M.A., Luszczek, P.: HPCG benchmark: a new metric for ranking high performance computing systems. Technical report. Electrical Engineering and Computer Science Department, Knoxville, Tennessee, UT-EECS-15-736 (2015)
39. Antonov, A., Popova, N., Voevodin, Vl.: Computational science and HPC education for graduate students: paving the way to exascale. J. Parallel Distrib. Comput. **118P1**, 157–165 (2018). https://doi.org/10.1016/j.jpdc.2018.02.023