

FORMATION OF REPRESENTATIONS OF ALGORITHMS FOR SOFTWARE SYSTEMS BASED ON THE CONCEPT OF A Q -DETERMINANT

V. N. Aleeva, M. P. Sokolov

South Ural State University (National Research University),
454080, Chelyabinsk, Russia

DOI: 10.24412/2073-0667-2026-1-5-23
EDN: EZZXDL

The paper continues the research based on the concept of the Q -determinant, which is the author's approach to parallelization of numerical algorithms. There are described the following concepts of the Q -determinant concept used in research.

An algorithmic problem has the form $\bar{y} = F(N, B)$, where $N = \{n_1, \dots, n_k\}$ is the set of parameters of the algorithmic problem dimension or N is the empty set, B is the set of input data, $\bar{y} = \{y_1, \dots, y_m\}$ is the set of output data. Here n_i ($i \in \{1, \dots, k\}$) is any positive integer. If $N = \{n_1, \dots, n_k\}$, then by $\bar{N} = \{\bar{n}_1, \dots, \bar{n}_k\}$ we denote a set of k positive integers, where $\bar{n}_i$ is some given value of the parameter n_i for each $i \in \{1, \dots, k\}$. We denote the set of all possible k -tuples $\bar{N}$ by $\{\bar{N}\}$. An algorithm for solving an algorithmic problem is denoted by $\mathcal{A}$, and the set of operations used by algorithm $\mathcal{A}$ is denoted by Q .

An expression over B and Q is a term in the standard sense of mathematical logic. A chain of length n is the result of applying some associative operation from Q to n expressions. If $N = \emptyset$, then any expression w over B and Q is an unconditional Q -term. If $N \neq \emptyset$ and V is the set of all expressions over B and Q , then any mapping $w : \{\bar{N}\} \rightarrow V \cup \emptyset$ is also called an unconditional Q -term. $w(\bar{N}) = \emptyset$ means that the value of $w(\bar{N})$ is undefined. If $N = \emptyset$ and the expression w over B and Q has a value of logical type under any interpretation of the variables B , then we call the unconditional Q -term w an unconditional logical Q -term. Let $N \neq \emptyset$ and w be an unconditional Q -term. Suppose also that the expression $w(\bar{N})$ for each $\bar{N} \in \{\bar{N}\}$ has a value of logical type under any interpretation of the variables B . In this case, we call the unconditional Q -term w an unconditional logical Q -term. Suppose that $u_1, \dots, u_l$ are unconditional logical Q -terms, $w_1, \dots, w_l$ are unconditional Q -terms. Then the set l of pairs $(\hat{u}, \hat{w}) = \{(u_i, w_i)\}_{i \in \{1, \dots, l\}}$ is a conditional Q -term of length l . Let $(\hat{u}, \hat{w}) = \{(u_i, w_i)\}_{i=1, 2, \dots}$ be a countable set of pairs of unconditional Q -terms. Suppose that $\{(u_i, w_i)\}_{i \in \{1, \dots, l\}}$ is a conditional Q -term of length l for any $l < \infty$. Then we call $(\hat{u}, \hat{w})$ a conditional infinite Q -term. Q -terms can be calculated.

Suppose that algorithm $\mathcal{A}$ consists in finding for each $i \in \{1, \dots, m\}$ the value y_i by computing the value of the Q -term f_i . Then the set of Q -terms $\{f_i \mid i \in \{1, \dots, m\}\}$ we call the Q -determinant of the algorithm $\mathcal{A}$. Moreover, we call the system of equations $\{y_i = f_i \mid i \in \{1, \dots, m\}\}$ a representation of the algorithm $\mathcal{A}$ in the form of a Q -determinant.

Previously, the concept of the Q -determinant was applied to improve the efficiency of parallel computing by exploring and exploiting the parallelism resources of numerical algorithms. To this end, a software Q -system for studying parallelism resources of algorithms and a program design method for the effective implementation of numerical algorithms have been developed. The Q -system and method can be used for any numerical algorithms, including those solving practically significant problems.

The method was applied to algorithms for matrix multiplication, solving systems of linear algebraic equations, finding the maximum element in an array of numbers, sorting, training neural networks, Dijkstra's algorithm, and other algorithms.

The method of designing programs for the efficient implementation of numerical algorithms is very useful, but its application requires the development of programs by programmers. There is a question about possibility of creating software system that develops and executes effective programs using a generally accepted description of algorithms without the use of programmers. To answer this question, several published papers have demonstrated the possibility of creating a unified software system for the automated design and execution of programs for the efficient implementation of numerical algorithms. These papers describe different approaches to designing such systems and outline general principles for their organization. The application of the developed software systems was demonstrated on problems with small values of the parameters of the algorithmic problem dimension. It is now important to explore their application to practically significant problems.

Software systems for the automated design and execution of programs for the efficient implementation of numerical algorithms use representations of algorithms in the form of Q -determinants to describe algorithms. These representations are formed using a subsystem of the Q -system for analyzing the parallelism resources of algorithms. The same representations of algorithms in the form of Q -determinants are also used by the subsystem for analyzing the characteristics of the parallelism resources of the Q -system. In this regard, the formation of representations of algorithms in the form of Q -determinants is an important task for the use of software systems based on the concept of Q -determinant.

The parameters of the algorithmic problems dimension can reach large values when solving practical problems. The number of iterations when implementing iterative algorithms can also be large. As a result, the Q -terms that make up the Q -determinants of algorithms can be very long, and the files containing the algorithm representations in the form of Q -determinants can be large. Therefore, problems can arise when forming and using algorithm representations in the form of Q -determinants. In this case, we will refer to Q -determinants as large, and the resulting problems as of large Q -determinant problems. Solving of large Q -determinant problems is extremely important for the practical application of software systems based on the Q -determinant concept.

In this paper, the problems of large Q -determinants are described and solved. More precisely, the problems that arise during the formation and use of representations of algorithms in the form of Q -determinants are formulated and solved. Furthermore, the results of experimental studies confirming the validity of the proposed solutions are presented. The solutions proposed in the paper enable the generation of representations of algorithms in the form of Q -determinants for large values of the algorithmic problem dimension parameters and a large number of iterations of iterative algorithms. Furthermore, these solutions reduce the size of files containing representations of algorithms in the form of Q -determinants.

The paper consists of three sections and a conclusion. Section 1 introduces the concepts of the Q -determinant concept used in this article. Section 2 describes the generation of representations of algorithms in the form of Q -determinants using algorithm flowcharts. Section 3 presents the problems of large Q -determinants and proposes solutions to them. The conclusion summarizes the results of the study and formulates the direction of further research.

Key words: Q -determinant of algorithm, representation of algorithm in form of Q -determinant, Q -effective implementation of algorithm, parallelism resource of algorithm, Q -effective program, computer-aided design of effective programs.

References

1. Aleeva V. N. Analiz parallel'ny'x chislenny'x algoritmov. Preprint N 590. Novosibirsk: VC SO AN SSSR, 1985. 23 s. (in Russian).
2. Valentina Aleeva, Rifkhat Aleev. Investigation and Implementation of Parallelism Resources of Numerical Algorithms // ACM Transactions on Parallel Computing. 2023. Vol. 10. N 2, Article number 8. P. 1–64. DOI: 10.1145/3583755.
3. Aleeva V. N. Podxod k e'ffektivnoj realizacii chislenny'x algoritmov // Problemy' informatiki. 2025. № 1. S. 29–44. DOI: 10.24412/2073-0667-2025-1-29-44 (in Russian).
4. Aleeva V. N., Kuzneczov E. K. E'ffektivnaya realizaciya algoritmov sortirovki s pomoshh'yu koncepcii Q -determinanta // Vestnik NGU. Seriya: Informacionny'e texnologii. 2025. T. 23, № 2. S. 5–17. DOI: 10.25205/1818-7900-2025-23-2-5-17 (in Russian).
5. Aleeva V. N., Sapozhnikov A. S. E'ffektivnaya realizaciya algoritmov obucheniya nejronny'x setej s pomoshh'yu koncepcii Q -determinanta // Problemy' informatiki. 2025. № 3. S. 5–16. DOI: 10.24412/2073-0667-2025-3-5-16 (in Russian).
6. Aleeva V. N. Avtomatizirovannoe proektirovanie i ispolnenie e'ffektivny'x programm dlya chislenny'x algoritmov // Vestnik YuUrGU. Seriya: Vy'chislitel'naya matematika i informatika. 2023. T. 12, № 3. S. 31–49. DOI: 10.14529/cmse230303 (in Russian).
7. Manatin P., Aleeva V. Efficient Implementation of Numerical Algorithms Based on a Lexical Analyzer // Parallel Computational Technologies. PCT 2024. Communications in Computer and Information Science. 2024. Vol. 2241. P. 107–121. DOI: 10.1007/978-3-031-73372-7_8.
8. Aleeva V. N. Razrabotka programmy'x sistem avtomatizirovannogo proektirovaniya i ispolneniya programm dlya e'ffektivnoj realizacii chislenny'x algoritmov na osnove koncepcii Q -determinanta // Vestnik NGU. Seriya: Informacionny'e texnologii. 2025. T. 23, № 1. S. 5–18. DOI: 10.25205/1818-7900-2025-23-1-5-18 (in Russian).
9. Ershov Yu. L., Palyutin E. A. Matematicheskaya logika. M.: Nauka, 1987. 336 s (in Russian).
10. Aleeva V. N., Zotova P. S., Skleznev D. S. Rasshirenie vozmozhnostej issledovaniya resursa parallelizma chislenny'x algoritmov s pomoshh'yu programmnoj Q -sistemy' // Vestnik YuUrGU. Seriya: Vy'chislitel'naya matematika i informatika. 2021. T. 10, № 2. S. 66–81. DOI: 10.14529/cmse210205 (in Russian).
11. Aleeva V. N. Improving Parallel Computing Efficiency // Proceedings — 2020 Global Smart Industry Conference, GloSIC 2020. IEEE. 2020. P. 113–120. Article number 9267828. DOI: 10.1109/GloSIC50886.2020.9267828.
12. Aleeva V., Bogatyreva E., Skleznev A., et al. Software Q -system for the Research of the Resource of Numerical Algorithms Parallelism // Supercomputing. RuSCDays 2019. Communications in Computer and Information Science. 2019. Vol. 1129. P. 641–652. DOI: 10.1007/978-3-030-36592-9_52.
13. Sokolov M. P. Razrabotka programmnogo obespecheniya dlya formirovaniya Q -determinantov chislenny'x algoritmov s Q -termami bol'shoj dliny' // Vy'p. kvalif. rabota bakalavra po napravleniyu "Fundamental'naya informatika i informacionny'e texnologii": 02.03.02 / Yuzhno-Ural'skij gosudarstvenny'j universitet. Chelyabinsk. 2020. 32 l. [Electron. Res.]: http://omega.sp.susu.ac.ru/publications/bachelorthesis/2020_402_sokolovmp.pdf. (Data obrashheniya: 15 dekabrya 2025 goda) (in Russian).
14. Superkomp'yuter "Tornado YuUrGU". [Electron. Res.]: <http://supercomputer.susu.ru/computers/tornado/> (Data obrashheniya: 15 dekabrya 2025 goda) (in Russian).
15. Sokolov M. P. Formirovanie predstavlenij algoritmov v forme Q -determinantov na klasterny'x vy'chislitel'ny'x sistemax // Vy'p. kvalif. rabota magistra po napravleniyu "Fundamental'naya informatika i informacionny'e texnologii": 02.04.02 / Yuzhno-Ural'skij gosudarstvenny'j universitet.

Chelyabinsk. 2022. 40 l. [Electron. Res.]: http://omega.sp.susu.ru/publications/masterthesis/2022_220_sokolovmp.pdf (Data obrashheniya: 15 dekabrya 2025 goda) (in Russian).

16. Oseledets I., Tyrtyshnikov E. TT-cross approximation for multidimensional arrays // Linear Algebra and its Applications, 2010. Vol. 432. N 1, P. 70–88. DOI: 10.1016/j.laa.2009.07.024.

17. Oseledets I. V. Tensor-Train Decomposition // SIAM Journal on Scientific Computing (SISC), 2011. Vol. 33. N 5, P. 2295–2317. DOI: 10.1137/090752286.

ФОРМИРОВАНИЕ ПРЕДСТАВЛЕНИЙ АЛГОРИТМОВ ДЛЯ ПРОГРАММНЫХ СИСТЕМ НА ОСНОВЕ КОНЦЕПЦИИ Q -ДЕТЕРМИНАНТА

В. Н. Алеева, М. П. Соколов

Южно-Уральский государственный университет (НИУ),
454080, Челябинск, Россия

УДК 004.021, 004.032.24, 004.051, 004.272

DOI: 10.24412/2073-0667-2026-1-5-23

EDN: EZZXDL

Концепция Q -детерминанта является одним из подходов к распараллеливанию численных алгоритмов. На ее основе можно повысить эффективность параллельных вычислений с помощью выявления ресурсов параллелизма алгоритмов с применением программной системы, а затем и их использования. Для этого была разработана технология параллельного программирования. Исследования с применением концепции Q -детерминанта доказали возможность создания программных систем автоматизированного проектирования и исполнения программ для эффективной реализации численных алгоритмов. Для описания алгоритмов программные системы должны использовать представления алгоритмов в форме Q -детерминантов. При решении практических задач файлы, содержащие представления алгоритмов в форме Q -детерминантов, в настоящее время могут иметь большие размеры. В связи с этим при формировании и использовании представлений алгоритмов в форме Q -детерминантов могут возникать проблемы. В этом случае Q -детерминанты называются большими, а возникающие проблемы — проблемами больших Q -детерминантов. В статье описаны проблемы больших Q -детерминантов и предложены их решения.

Ключевые слова: Q -детерминант алгоритма, представление алгоритма в форме Q -детерминанта, Q -эффективная реализация алгоритма, ресурс параллелизма алгоритма, Q -эффективная программа, автоматизированное проектирование эффективных программ.

Введение. Статья продолжает исследования на основе концепции Q -детерминанта — авторского подхода к распараллеливанию численных алгоритмов [1]. Ранее с его помощью было предложено повышение эффективности параллельных вычислений за счет исследования и использования ресурсов параллелизма численных алгоритмов [2]. Для этого, в частности, разработан метод проектирования программ для эффективной реализации численных алгоритмов, называемых Q -эффективными. Метод может быть использован для любых численных алгоритмов, в том числе для решения практически значимых задач. Обзор по применению данного метода для различных алгоритмов приведен в работе [3]. Его дополняет работа [4] по использованию метода для алгоритмов сортировки и работа [5] по использованию метода для алгоритмов обучения нейронных сетей.

Метод проектирования программ для эффективной реализации численных алгоритмов очень полезен, однако его применение предполагает разработку программ специалистами. А можно ли создать программную систему, которая без использования специалистов

разрабатывает и исполняет эффективные программы, используя общепринятое описание алгоритмов? Чтобы ответить на этот вопрос, была показана возможность создания единой для численных алгоритмов программной системы автоматизированного проектирования и исполнения программ для эффективной реализации численных алгоритмов [6–8]. В работах [6–7] описаны разные подходы к проектированию таких систем, а в [8] изложены общие принципы их организации. Применение разработанных систем было продемонстрировано на задачах с небольшими значениями параметров размерности. Теперь важно исследовать их работоспособность на практически значимых задачах.

Программные системы автоматизированного проектирования и исполнения программ для эффективной реализации численных алгоритмов используют для описания алгоритмов представления алгоритмов в форме Q -детерминантов, формируемые с помощью подсистемы программной Q -системы для исследования ресурсов параллелизма алгоритмов [2]. Такие же представления алгоритмов в форме Q -детерминантов использует и подсистема для анализа характеристик ресурсов параллелизма алгоритмов программной Q -системы. В связи с этим подготовка представлений алгоритмов в форме Q -детерминантов является важной задачей для использования программных систем на основе концепции Q -детерминанта.

При решении практических задач параметры размерности алгоритмической проблемы могут достигать больших значений, количество итераций при реализации итерационных алгоритмов также может быть большим. В результате Q -термы, составляющие Q -детерминанты алгоритмов, могут иметь большую длину, а файлы, содержащие представления алгоритмов в форме Q -детерминантов, большие размеры. В связи с этим при формировании и использовании представлений алгоритмов в форме Q -детерминантов могут возникать проблемы. В этом случае Q -детерминанты будем называть большими, а возникающие проблемы — проблемами больших Q -детерминантов. Решение этих проблем чрезвычайно важно для практического применения программных систем на основе концепции Q -детерминанта.

Цель данного исследования — решение проблем больших Q -детерминантов. Для достижения цели необходимо решить следующие задачи.

- 1) Описание и решение проблем, возникающих при формировании и использовании представлений алгоритмов в форме Q -детерминантов.
- 2) Описание экспериментальных исследований, подтверждающих правильность предложенных решений.

Статья организована следующим образом. Раздел 1 содержит некоторые основные понятия концепции Q -детерминанта, используемые в данной работе. В разделе 2 описано формирование представлений алгоритмов в форме Q -детерминантов с помощью их блок-схем. В разделе 3 приведены проблемы больших Q -детерминантов, предложены их решения и описаны проведенные экспериментальные исследования. В заключении подводятся итоги исследования и формулируется направление дальнейших исследований.

1. Некоторые основные понятия концепции Q -детерминанта. Концепция Q -детерминанта в настоящее время наиболее полно представлена в работе [2]. Кратко опишем понятия концепции Q -детерминанта, используемые в данной работе.

Алгоритмическую проблему опишем в виде $\bar{y} = F(N, B)$, где $N = \{n_1, \dots, n_k\}$ — множество параметров размерности проблемы или N — пустое множество, B — множество входных данных, $\bar{y} = \{y_1, \dots, y_m\}$ — множество выходных данных, при этом целое число m является либо константой, либо значением вычислимой функции параметров N при усло-

вии, что $N \neq \emptyset$. Здесь n_i ($i \in \{1, \dots, k\}$) равно любому положительному целому числу. Если $N = \{n_1, \dots, n_k\}$, то через $\bar{N} = \{\bar{n}_1, \dots, \bar{n}_k\}$ обозначим набор из k положительных целых чисел, где $\bar{n}_i$ — некоторое заданное значение параметра n_i для каждого $i \in \{1, \dots, k\}$. Через $\{\bar{N}\}$ обозначим множество всех возможных k -наборов $\bar{N}$. Пусть $\mathcal{A}$ — алгоритм для решения алгоритмической проблемы, Q — набор операций, используемых алгоритмом $\mathcal{A}$.

Определение 1. Под выражением над B и Q будем понимать терм в стандартном смысле математической логики [9].

Определение 2. Мы называем выражение цепочкой длины n , если оно является результатом применения некоторой ассоциативной операции из Q к n выражениям.

Определение 3. Если $N = \emptyset$, то любое выражение w над B и Q мы называем безусловным Q -термом. Пусть $N \neq \emptyset$ и V — множество всех выражений над B и Q . Тогда любое отображение $w : \{\bar{N}\} \rightarrow V \cup \emptyset$ также называется безусловным Q -термом.

Определение 4. Пусть $N = \emptyset$ и w — безусловный Q -терм. Предположим, что выражение w над B и Q имеет значение логического типа при любой интерпретации переменных B . Тогда безусловный Q -терм w называется безусловным логическим Q -термом. Пусть $N \neq \emptyset$ и w — безусловный Q -терм. Если выражение $w(\bar{N})$ для каждого $\bar{N} \in \{\bar{N}\}$ имеет значение логического типа при любой интерпретации переменных B , то безусловный Q -терм w называется безусловным логическим Q -термом.

Определение 5. Пусть $u_1, \dots, u_l$ — безусловные логические Q -термы, $w_1, \dots, w_l$ — безусловные Q -термы. Тогда множество l пар

$$(\hat{u}, \hat{w}) = \{(u_i, w_i)\}_{i \in \{1, \dots, l\}}$$

называется условным Q -термом длины l .

Определение 6. Пусть $(\hat{u}, \hat{w}) = \{(u_i, w_i)\}_{i=1,2,\dots}$ — счетное множество пар безусловных Q -термов. Предположим, что $\{(u_i, w_i)\}_{i \in \{1, \dots, l\}}$ — условный Q -терм для любого $l < \infty$. Тогда мы называем $(\hat{u}, \hat{w})$ условным бесконечным Q -термом.

Опишем нахождение значения безусловного Q -терма w при интерпретации переменных B . Если $N = \emptyset$, то нахождение значения выражения w означает нахождение значения безусловного Q -терма w при любой интерпретации переменных B . Если $N \neq \emptyset$ и $w(\bar{N}) \neq \emptyset$, то $w(\bar{N})$ является выражением над B и Q . Можно найти значение выражения $w(\bar{N})$ при любой интерпретации переменных B . Конечно, мы опускаем значение $w(\bar{N}) = \emptyset$. Следовательно, мы находим значение безусловного Q -терма w при любой интерпретации переменных B . Теперь опишем нахождение значения условного Q -терма $(\hat{u}, \hat{w})$ при интерпретации переменных B . Пусть $N = \emptyset$. Находим значения выражений u_i, w_i для $i \in \{i = 1, \dots, l\}$. При нахождении значений мы можем найти пару u_{i_0}, w_{i_0} такую, что u_{i_0} имеет значение **true**. Следовательно, мы можем найти значение w_{i_0} . Тогда считаем, что $(\hat{u}, \hat{w})$ имеет значение w_{i_0} . В противном случае считаем, что значение $(\hat{u}, \hat{w})$ при интерпретации переменных B не определено. Пусть $N \neq \emptyset$ и $\bar{N} \in \{\bar{N}\}$. Находим выражения $u_i(\bar{N}), w_i(\bar{N})$ для $i \in \{i = 1, \dots, l\}$. При нахождении значений мы можем найти пару $u_{i_0}(\bar{N}), w_{i_0}(\bar{N})$ такую, что $u_{i_0}(\bar{N})$ имеет значение **true**. Следовательно, мы можем найти значение $w_{i_0}(\bar{N})$. Тогда мы считаем, что $(\hat{u}, \hat{w})$ имеет значение $w_{i_0}(\bar{N})$. В противном случае считаем, что значение $(\hat{u}, \hat{w})$ для $\bar{N}$ и при такой интерпретации переменных B не определено. Аналогично можно определить значение условного бесконечного Q -терма.

Определение 7. Пусть $M = \{1, \dots, m\}$. Предположим, что алгоритм $\mathcal{A}$ состоит в нахождении для каждого $i \in M$ значения y_i путем вычисления значения Q -терма f_i . Тогда

набор Q -термов $\{f_i \mid i \in M\}$ называется Q -детерминантом алгоритма $\mathcal{A}$. Система уравнений $\{y_i = f_i \mid i \in M\}$ называется представлением алгоритма $\mathcal{A}$ в форме Q -детерминанта.

Определение 8. Процесс вычисления Q -термов $\{f_i \mid i \in M\}$ алгоритма $\mathcal{A}$ называется реализацией алгоритма $\mathcal{A}$. Реализация алгоритма $\mathcal{A}$ называется параллельной, если существуют операции, которые выполняются одновременно.

Для дальнейшего изложения будем использовать некоторое разбиение множества M . Предположим, что U , C и I образуют разбиение множества $M = \{1, \dots, m\}$ с пустыми членами, то есть: $U \cup C \cup I = M$; $U \cap C = U \cap I = C \cap I = \emptyset$; кроме того, одно или два подмножества U , C и I могут быть пустыми. Предположим также, что подмножества U , C и I можно связать с подмножествами множества Q -термов $\{f_i \mid i \in M\}$ таким образом: 1) для каждого $i \in U$ существует Q -терм f_i , который является безусловным, и $f_i = w^i$; 2) для каждого $i \in C$ существует Q -терм f_i , который является условным, и $f_i = \{(u_j^i, w_j^i)\}_{j \in \{1, \dots, l(i)\}}$, где $l(i)$ — либо константа, либо значение вычислимой функции от N , если $N \neq \emptyset$; 3) для каждого $i \in I$ существует Q -терм f_i , который является условным бесконечным, и $f_i = \{(u_j^i, w_j^i)\}_{j \in \{1, 2, \dots\}}$.

Определение 9. Опишем реализацию алгоритма $\mathcal{A}$, называемую Q -эффективной. Пусть $N = \emptyset$. Зададим интерпретацию переменных B . Будем вычислять выражения

$$W = \{w^i(i \in U); u_j^i, w_j^i(i \in C, j \in \{1, \dots, l(i)\}); u_j^i, w_j^i(i \in I, j \in \{1, 2, \dots\})\}$$

одновременно, параллельно. Мы говорим, что операция готова к выполнению, если вычислены значения всех ее операндов. При вычислении каждого из выражений W мы выполняем операции, как только они готовы к выполнению. Если несколько операций цепочки готовы к выполнению, то они выполняются по схеме сдваивания. Например, вычисление цепочки $a_1 + a_2 + a_3 + a_4$ по схеме сдваивания выполняется так: сначала вычисляем $b_1 = a_1 + a_2$ и $b_2 = a_3 + a_4$ одновременно, затем $c = b_1 + b_2$. Если для $i \in C \cup I$ и $j \in \{1, 2, \dots\}$ u_j^i имеет значение **false**, то вычисление соответствующего w_j^i прекращается. Если для $i \in C \cup I$ и $j \in \{1, 2, \dots\}$ вычисление пары (u_j^i, w_j^i) приводит к тому, что значение одного выражения не определено, то вычисление второго выражения прекращается. Если для $i \in C \cup I$ вычисление некоторой пары $(u_{j_0}^i, w_{j_0}^i)$ приводит к определению их значений и значение $u_{j_0}^i$ **true**, то вычисление выражений u_j^i, w_j^i прекращается для любого $j \neq j_0$.

Теперь пусть $N \neq \emptyset$. Зададим интерпретацию переменных B и $\bar{N} \in \{\bar{N}\}$. Получаем множество выражений

$$W(\bar{N}) = \{w^i(\bar{N})(i \in U); u_j^i(\bar{N}), w_j^i(\bar{N})(i \in C, j \in \{1, \dots, l(i)\}); u_j^i(\bar{N}), w_j^i(\bar{N})(i \in I, j \in \{1, 2, \dots\})\}.$$

Выражения $W(\bar{N})$ вычисляются по аналогии с выражениями W .

Замечание. Определение Q -эффективной реализации показывает, что она полностью использует ресурс параллелизма алгоритма.

Ресурс параллелизма алгоритма характеризуют его высота и ширина. Эти понятия рассматриваются в работах [2, 10–12]. В [2, 10, 12] описана программная Q -система, разработанная для автоматизированного исследования ресурса параллелизма алгоритмов, а также сравнения ресурсов параллелизма алгоритмов, решающих одну и ту же алгоритмическую проблему.

Определение 10. Реализация алгоритма $\mathcal{A}$ называется выполнимой, если одновременно должно выполняться конечное (непустое) множество операций.

Замечание. Существуют алгоритмы, Q -эффективная реализация которых невыполнима. Пример такого алгоритма приведен в работе [10].

2. Использование блок-схем алгоритмов для формирования их представлений в форме Q -детерминантов. Опишем метод формирования представлений алгоритмов в форме Q -детерминантов, реализованный в подсистеме для генерации Q -детерминантов численных алгоритмов программной Q -системы [12].

Представления алгоритмов в форме Q -детерминантов можно формировать, используя их блок-схемы. Блок-схемы алгоритмов хорошо известны и часто используются. Опишем ограничения и дадим пояснения для блок-схем, применяемых в исследованиях на основе концепции Q -детерминанта. Блок-схема имеет два терминальных блока, один из которых «Начало» означает начало алгоритма, а другой «Конец» — конец алгоритма. Блок «Начало» имеет одну исходящую линию потока и не имеет входящей линии потока. Блок «Конец» имеет одну входящую линию потока и не имеет исходящей линии потока. Блок «Решение» имеет две исходящие потоковые линии, одна из которых соответствует передаче управления, если условие блока имеет значение true, а другая, если его значение false. Условие блока использует одну операцию сравнения. Его операнды не содержат операций. Блок «Процесс» содержит одно присвоение значения переменной. Справа от знака присваивания находится не более одной операции. Для групп операций используются цепочки блоков. Блоки «Данные» делятся на блоки «Входные данные» и «Выходные данные».

Для формирования представления алгоритма в форме Q -детерминанта блок-схема алгоритма анализируется при фиксированных значениях параметров размерности алгоритмической проблемы $\bar{N} = \{\bar{n}_1, \dots, \bar{n}_k\}$, если $N \neq \emptyset$. При этом исследуются особенности алгоритмов, Q -детерминанты которых содержат различные типы Q -термов.

Если Q -детерминант алгоритма имеет безусловные Q -термы, то проход по блок-схеме должен осуществляться последовательно в соответствии с выполнением алгоритма. Для каждого $i \in U$ выражения w^i , если $N = \emptyset$, и $w^i(\bar{N})$, если $N \neq \emptyset$, получаются, как значения безусловного Q -терма f_i . Они формируются с использованием содержимого блоков «Процесс», участвующих в вычислении f_i для каждого $i \in U$.

Предположим, что Q -детерминант содержит условные Q -термы. Проход по блоку «Решение» с условием, содержащим входные данные, порождает ветвление. Затем каждая ветвь обрабатывается отдельно. В результате обработки одной ветви для каждого $i \in C$ и соответствующего ветви $j \in \{1, \dots, l(i)\}$ получаем пары выражений (u_j^i, w_j^i) , если $N = \emptyset$ и $(u_j^i(\bar{N}), w_j^i(\bar{N}))$, если $N \neq \emptyset$. Выражения u_j^i , если $N = \emptyset$ и $u_j^i(\bar{N})$, если $N \neq \emptyset$ формируются из условий блоков «Решение», содержащих входные данные. Выражения w_j^i , если $N = \emptyset$ и $w_j^i(\bar{N})$, если $N \neq \emptyset$ формируются с использованием содержимого блоков «Процесс». Как только первая ветвь обработана, обработчик блок-схемы возвращается к ближайшему блоку, где произошло ветвление, и продолжает обработку с противоположным условием. Следующая ветвь будет получена путем возврата к ближайшему блоку, где ветвление выполнялось один раз. Если таких блоков не осталось, то перебор ветвей заканчивается. После обработки всех ветвей для всех $i \in C$ и $j \in \{1, \dots, l(i)\}$ получаем пары выражений (u_j^i, w_j^i) , если $N = \emptyset$ и $(u_j^i(\bar{N}), w_j^i(\bar{N}))$, если $N \neq \emptyset$ для всех условных Q -термов f_i .

Условные бесконечные Q -термы содержатся в Q -детерминантах итерационных численных алгоритмов. Ограничивая количество итераций, можно свести случай Q -детерминанта с условными бесконечными Q -термами к случаю Q -детерминанта с условными Q -термами. Будем использовать параметр L для количества итераций. Этот параметр

может принимать любые целые положительные значения. Обозначим значение параметра L через $\bar{L}$. Значение $\bar{L}$ определяет длину Q -термов f_i для каждого $i \in I$.

Опишем формат файлов, содержащих представления алгоритмов в форме Q -детерминантов, формируемых с помощью подсистемы для генерации Q -детерминантов численных алгоритмов программной Q -системы. Представления алгоритмов в форме Q -детерминантов имеют фиксированные значения параметров размерности N и ограниченное число итераций L .

Для каждого $i \in C$ условный Q -терм f_i определяет $l(i)$ строк файла, где $l(i)$ — длина Q -терма. Каждая из этих строк содержит идентификатор выходной переменной, вычисляемой с использованием данного Q -терма, знак равенства и одну пару выражений (u_j^i, w_j^i) , если $N = \emptyset$ или $(u_j^i(\bar{N}), w_j^i(\bar{N}))$, если $N \neq \emptyset$ для некоторого $j \in \{1, \dots, l(i)\}$. Здесь для каждого $j \in \{1, \dots, l(i)\}$ Q -термы таковы, что u_j^i является безусловным логическим Q -термом, а w_j^i безусловным Q -термом. Выражения (u_j^i, w_j^i) , если $N = \emptyset$, или $(u_j^i(\bar{N}), w_j^i(\bar{N}))$, если $N \neq \emptyset$, описываются в формате JSON и разделяются точкой с запятой. Описание условного бесконечного Q -терма аналогично описанию условного Q -терма, поскольку длина условного бесконечного Q -терма ограничена значением параметра L .

Для каждого $i \in U$ безусловный Q -терм f_i определяет одну строку файла. Безусловные Q -термы также описываются в формате JSON. В этом случае нет логического Q -терма, поэтому вместо него используется пробел.

Для описания выражений в формате JSON применяются следующие обозначения: op — унарная или бинарная операция, od — операнд унарной операции, fO — первый операнд бинарной операции, sO — второй операнд бинарной операции, скобки $\{$ и $\}$ определяют порядок выполнения операций.

Пример. Пусть матрица A размера 2×2 имеет ненулевой определитель. Предположим, что $\vec{X} = (X(1), X(2))^T$ и $\vec{B} = (\bar{A}(1,3), \bar{A}(2,3))^T$ — векторы-столбцы, где $\bar{A} = [A(i,j)]_{2 \times 3}$ — расширенная матрица системы $A\vec{X} = \vec{B}$. На рис. 1 показано содержимое файла с представлением в форме Q -детерминанта алгоритма решения системы линейных уравнений второго порядка методом Гаусса–Жордана.

3. Проблемы больших Q -детерминантов и их решения.

3.1 Формирование представлений алгоритмов в форме Q -детерминантов с использованием персонального компьютера. Подсистема для генерации Q -детерминантов численных алгоритмов сначала использовалась на персональном компьютере. В результате было сформировано множество представлений алгоритмов в форме Q -детерминантов для алгоритмов с различными структурами Q -детерминантов, с различными значениями параметров размерности, а для итерационных алгоритмов с различными значениями количества итераций. Это множество представлений алгоритмов в форме Q -детерминантов было использовано для разработки и тестирования подсистемы Q -системы для вычисления ресурсов параллелизма численных алгоритмов, а также для разработки и тестирования программных систем автоматизированного проектирования и исполнения программ для эффективной реализации численных алгоритмов.

Эксплуатация подсистемы для генерации Q -детерминантов численных алгоритмов выявила, что при формировании Q -термов некоторых алгоритмов возникает исключение `OutOfMemoryException`, что делает невозможным в этом случае формирование представлений алгоритмов в форме Q -детерминантов. Например, исключение возникало при формировании представления в форме Q -детерминанта алгоритма для решения системы линейных уравнений методом Гаусса–Зейделя при задании количества линейных уравнений

```

X(1)={"op": "!=", "f0": "A(1,1)", "s0": "0"};
  {"op": "-", "f0": {"op": "/", "f0": "A(1,3)",
    "s0": "A(1,1)"},
    "s0": {"op": "*", "f0": {"op": "/", "f0": {"op": "-", "f0": "A(2,3)",
      "s0": {"op": "*", "f0": {"op": "/", "f0": "A(1,3)",
        "s0": "A(1,1)"},
        "s0": "A(2,1)"},
        "s0": {"op": "-", "f0": "A(2,2)",
          "s0": {"op": "*", "f0": {"op": "/", "f0": "A(1,2)",
            "s0": "A(1,1)"},
            "s0": "A(2,1)"},
            "s0": {"op": "/", "f0": "A(1,2)",
              "s0": "A(1,1)"}}}
          "s0": "A(1,1)"}}}
X(2)={"op": "!=", "f0": "A(1,1)", "s0": "0"};
  {"op": "/", "f0": {"op": "-", "f0": "A(2,3)",
    "s0": {"op": "*", "f0": {"op": "/", "f0": "A(1,3)",
      "s0": "A(1,1)"},
      "s0": "A(2,1)"},
      "s0": {"op": "-", "f0": "A(2,2)",
        "s0": {"op": "*", "f0": {"op": "/", "f0": "A(1,2)",
          "s0": "A(1,1)"},
          "s0": "A(2,1)"}}}
        "s0": "A(1,1)"}}}
X(1)={"op": "=", "f0": "A(1,1)", "s0": "0"};
  {"op": "/", "f0": {"op": "-", "f0": "A(2,3)",
    "s0": {"op": "*", "f0": {"op": "/", "f0": "A(1,3)",
      "s0": "A(1,2)"},
      "s0": {"op": "-", "f0": "A(2,1)",
        "s0": {"op": "*", "f0": {"op": "/", "f0": "A(1,1)",
          "s0": "A(1,2)"},
          "s0": "A(2,2)"}}}
        "s0": "A(1,2)"}}}
X(2)={"op": "=", "f0": "A(1,1)", "s0": "0"};
  {"op": "-", "f0": {"op": "/", "f0": "A(1,3)",
    "s0": "A(1,2)"},
    "s0": {"op": "*", "f0": {"op": "/", "f0": {"op": "-", "f0": "A(2,3)",
      "s0": {"op": "*", "f0": {"op": "/", "f0": "A(1,3)",
        "s0": "A(1,2)"},
        "s0": {"op": "-", "f0": "A(2,1)",
          "s0": {"op": "*", "f0": {"op": "/", "f0": "A(1,1)",
            "s0": "A(1,2)"},
            "s0": "A(2,2)"}}}
          "s0": "A(1,2)"},
          "s0": {"op": "/", "f0": "A(1,1)",
            "s0": "A(2,2)"}}}
            "s0": {"op": "/", "f0": "A(1,1)",
              "s0": "A(1,2)"}}}

```

Рис. 1. Представление в форме Q -детерминанта алгоритма решения СЛАУ второго порядка методом Гаусса-Жордана

10 и количества итераций 2, а при задании количества линейных уравнений 20 исключение возникало для одной итерации. Причина возникновения исключения состоит в нехватке оперативной памяти для хранения и обработки Q -термов большой длины. В работе [13] было предложено решение данной проблемы, заключающееся в том, что в процессе формирования Q -термы хранятся и обрабатываются не в оперативной памяти, а в постоянной памяти. Проблема была решена, но время формирования представлений алгоритмов в форме Q -детерминантов возросло.

Когда была доказана возможность создания на основе концепции Q -детерминанта программных систем автоматизированного проектирования и исполнения программ для эффективной реализации численных алгоритмов, стало понятно, что вычисления представлений алгоритмов в форме Q -детерминантов на персональном компьютере выполнили свою задачу, поэтому было принято решение использовать для этих вычислений суперкомпьютер «Торнадо ЮУрГУ» [14].

3.2. Формирование представлений алгоритмов в форме Q -детерминантов с использованием суперкомпьютера. Для формирования представлений алгоритмов в форме Q -

Таблица 1

Результаты вычислительных экспериментов для алгоритма
умножения двух плотных квадратных матриц

Размер матриц	Размер выходного файла, МБ
150	209
160	254
170	306
180	364
190	429
200	502

детерминантов с помощью кластерной вычислительной системы «Торнадо ЮУрГУ» было разработано новое программное обеспечение [15], использующее метод формирования представлений алгоритмов в форме Q -детерминантов, который применялся в подсистеме для генерации Q -детерминантов численных алгоритмов программной Q -системы (см. раздел 2). Для организации распределенных вычислений на кластерной вычислительной системе разработанное программное обеспечение использует технологию MPI, с помощью которой осуществляется распределение проходов по блок-схеме алгоритма по вычислительным узлам системы. В результате использования технологии MPI существенно сократилось время формирования выходного файла, а также стало возможным получить представления алгоритмов в форме Q -детерминантов, которые было невозможно получить ранее, используя последовательную программу на персональном компьютере.

Приведем некоторые результаты, полученные с помощью вычислительных экспериментов с использованием разработанного программного обеспечения. Результаты вычислительных экспериментов, полученные при формировании представлений в форме Q -детерминанта алгоритма умножения двух плотных квадратных матриц, приведены в табл. 1. В табл. 2 представлены результаты вычислительных экспериментов, показывающие зависимость размера выходного файла от длины массива при формировании представлений алгоритма поиска максимального элемента в массиве чисел в форме Q -детерминанта. Кроме того, были получены результаты вычислительных экспериментов, в которых использовались итерационные алгоритмы. В этом случае формирование представлений алгоритмов в форме Q -детерминантов выполняется при фиксированных значениях параметров размерности и числа итераций. Табл. 3 содержит результаты вычислительных экспериментов для алгоритма, реализующего метод Якоби, а табл. 4 для алгоритма, реализующего метод Гаусса-Зейделя для решения систем линейных уравнений (СЛАУ).

Приведенные результаты показывают, что с ростом параметров размерности и количества итераций файл, содержащий представление алгоритма в форме Q -детерминанта, может достигать довольно большого размера, затрудняющего его формирование, хранение и использование. Следует отметить, что представления алгоритмов в форме Q -детерминантов могут быть заранее сформированы и сохранены для дальнейшего использования, поэтому время их формирования не имеет такого значения, как размер содержащих их файлов. Несмотря на это, быстроедействие программного обеспечения для формирования представлений алгоритмов в форме Q -детерминантов планируется повысить за счет имеющегося у него потенциала распараллеливания, что будет способствовать решению практически значимых задач.

Таблица 2

Результаты вычислительных экспериментов для алгоритма поиска
максимального элемента в массиве чисел

Длина массива	Размер выходного файла, МБ
24	10784
25	22528
26	46972
27	97779
28	203227
29	421789
30	874250

Таблица 3

Результаты вычислительных экспериментов для алгоритма,
реализующего метод Якоби для решения СЛАУ

Размер матрицы	Количество итераций	Размер выходного файла, МБ
3	19	2483
4	13	6093
5	11	17487
6	9	9501
7	8	9674
8	6	801
8	7	5608
9	6	2096
9	7	16772

3.3. *Подход к сокращению размеров файлов с представлениями алгоритмов в форме Q -детерминантов.* Опишем решения, позволяющие сократить размеры файлов, содержащих представления алгоритмов в форме Q -детерминантов, формируемые с помощью подсистемы Q -системы для генерации Q -детерминантов численных алгоритмов. Эти решения основаны на особенностях структуры представлений алгоритмов в форме Q -детерминантов.

В разделе 2 показано, что после обработки всех ветвей блок-схемы алгоритма для всех $i \in C$ и $j \in \{1, \dots, l(i)\}$ будут получены пары выражений (u_j^i, w_j^i) , если $N = \emptyset$ и $(u_j^i(\bar{N}), w_j^i(\bar{N}))$, если $N \neq \emptyset$ для всех условных Q -термов f_i . При этом безусловные логические Q -термы с одинаковыми значениями j совпадают, так как они формируются из условий блоков «Решение», содержащих входные данные, при одном проходе по блок-схеме. Рис. 1 подтверждает это утверждение. Таким образом, в случае условных Q -термов значительную часть файла, содержащего представление алгоритма в форме Q -детерминанта, могут занимать повторяющиеся безусловные логические Q -термы. Целесообразно сократить размер файла за счет однократного использования безусловных логических Q -термов.

Предлагается также формировать, хранить и использовать представления алгоритмов в форме Q -детерминантов, разделив их согласно разработанному стандарту на некото-

Таблица 4

Результаты вычислительных экспериментов для алгоритма, реализующего метод Гаусса-Зейделя для решения СЛАУ

Размер матрицы	Количество итераций	Размер выходного файла, МБ
3	10	3854
4	6	3904
10	2	6554
20	1	25200

рое количество частей, размещаемых в отдельных архивированных файлах. Опишем этот стандарт.

Если представление алгоритма в форме Q -детерминанта содержит безусловные Q -термы, то его строки должны быть последовательно распределены по файлам с именами k_w , где k — порядковый номер файла. Количество файлов выбирается, исходя из структуры Q -детерминанта. В каждой строке после идентификатора выходной переменной и знака равенства сразу следует безусловный Q -терм, то есть исключаются пробел, используемый вместо безусловного логического Q -терма, и точка с запятой, разделяющая безусловный логический и безусловный Q -термы.

Если представление алгоритма в форме Q -детерминанта содержит условные Q -термы, то предлагается три варианта согласно разработанному стандарту: с разделением по логическим Q -термам, с разделением безусловных логических и безусловных Q -термов, с разделением по безусловным Q -термам. Следует выбирать оптимальный вариант в зависимости от структуры Q -детерминанта. Опишем эти варианты.

Первый вариант представления алгоритма в форме Q -детерминанта, содержащего условные Q -термы, с разделением по логическим Q -термам состоит в следующем. Строки представления алгоритма в форме Q -детерминанта должны быть распределены по файлам с именами k_uw , где k — порядковый номер файла. Каждый из этих файлов содержит либо описание одного подмножества пар условных Q -термов, имеющих одинаковые безусловные логические Q -термы, либо нескольких подмножеств. Описание подмножества пар условных Q -термов состоит в том, что первая строка содержит безусловный логический Q -терм, а в каждой из следующих строк находится идентификатор выходной переменной, знак равенства и безусловный Q -терм. Количество строк в файле и количество файлов выбираются, исходя из структуры Q -детерминанта.

Второй вариант представления алгоритма в форме Q -детерминанта, содержащего условные Q -термы, с разделением безусловных логических и безусловных Q -термов заключается в следующем. Пусть Q -детерминант состоит из условных Q -термов длины M . Q -термы должны быть распределены по файлам с именами

$$1_u1, 2_w1, 3_u2, 4_w2, 5_u3, 6_w3, \dots, (K-1)_uM, K_wM,$$

где K — количество файлов. Пары условных Q -термов, имеющие порядковый номер m , описываются следующим образом: безусловный логический Q -терм содержится в файле с именем k_um , где k — порядковый номер файла, а безусловные Q -термы в файле с именем

$(k + 1)_{wm}$, каждая из строк которого содержит идентификатор выходной переменной, знак равенства и один безусловный Q -терм.

Третий вариант представления алгоритма в форме Q -детерминанта, содержащего условные Q -термы, заключающийся в разделении по безусловным Q -термам, отличается от предыдущего построчным последовательным распределением содержимого файлов с именами k_{wm} по файлам с именами

$$k_{wm}, k_{wmp1}, k_{wmp2}, k_{wmp3}, \dots$$

Количество Q -термов, содержащееся в каждом таком файле, и количество файлов выбираются исходя из структуры Q -детерминанта.

По-видимому, можно провести аналогию между описанным подходом к сокращению размеров файлов с представлениями алгоритмов в форме Q -детерминантов и подходом, использующим понятие тензорного поезда (Tensor Train) — формы разложения многомерных массивов (тензоров), которая не страдает от «проклятия размерности» [16–17]. Учитывая это, представление алгоритма в форме Q -детерминанта, полученное в соответствии с предложенным стандартом, можно назвать Q -детерминантным поездом (Q -determinant Train).

Внедрение описанного подхода позволило значительно сократить размер выходных данных подсистемы для генерации Q -детерминантов численных алгоритмов. В зависимости от структуры Q -детерминанта величина сокращения размера разная. Для подтверждения этого опишем проведенные вычислительные эксперименты.

Так как пользователи суперкомпьютера имеют ограничение на использование дискового пространства, было невозможно сформировать представления в форме Q -детерминанта алгоритма поиска максимального элемента в массиве чисел длины более 30. Использование нового формата и архивации по мере формирования файлов, содержащих выходные данные, позволило сформировать такие представления. В данном случае Q -детерминант алгоритма состоит из одного условного Q -терма длины $2^{(l-1)}$, где l — длина массива. Пары Q -терма таковы, что их безусловные логические Q -термы не повторяются, а безусловные Q -термы представляют из себя элементы массива. С учетом структуры Q -детерминанта был выбран первый вариант представления алгоритма в форме Q -детерминанта, содержащего условные Q -термы, с разделением по логическим Q -термам. С использованием нового формата были сформированы как представления в форме Q -детерминанта алгоритма, которые могут быть получены в исходном формате, так и представления, которые в исходном формате получить было невозможно. Результаты экспериментального исследования представлены в табл. 5.

В целях равномерного и универсального распределения пар условного Q -терма по файлам было принято решение размещать в каждом файле число пар, являющееся степенью числа 2. Тогда количество файлов также будет числом, являющимся степенью числа 2. Для организации параллельного формирования файлов было выбрано максимально доступное пользователю суперкомпьютера число MPI-процессов, являющееся степенью числа 2, а именно 256. Также при выборе количества информации в одном файле необходимо учесть, что суммарный размер параллельно формируемых файлов не должен превышать доступное дисковое пространство. Рассмотрим пример, когда массив чисел имеет длину 30. В этом случае число пар условного Q -терма равно $2^{29} = 536870912$. При формировании представления алгоритма в форме Q -детерминанта было получено $2^{12} = 4096$ архивированных файла, каждый из которых содержал $2^{17} = 131072$ пар условного Q -термов. Каж-

Таблица 5

Результаты экспериментального исследования алгоритма
поиска максимального элемента в массиве чисел

Длина массива	Объем выходных данных в исходном формате, МБ	Объем выходных данных в новом формате, МБ
24	10784	102,48
25	22528	213,92
26	46972	445,76
27	97779	927,36
28	203227	1926,40
29	421789	3996,16
30	874250	8279,04
31	—	17131,52
32	—	35409,92
33	—	73113,60
34	—	150814,70
35	—	310804,50

Таблица 6

Результаты экспериментального исследования алгоритма, реализующего метод Якоби для решения пятиточечных разностных уравнений

Количество итераций	Объем выходных данных в исходном формате, МБ	Объем выходных данных в новом формате, МБ
3	687581	4,6
4	2761375	14,9
5	11009442	51,6

дая из пар была описана в виде двух строк: первая содержит безусловный логический Q -терм, вторая идентификатор выходной переменной «max», знак равенства и безусловный Q -терм. В результате использования нового формата и архивации по мере формирования файлов, содержащих выходные данные, объем выходных данных сократился на 99 %.

В табл. 6 представлены результаты экспериментального исследования алгоритма, реализующего метод Якоби для решения пятиточечных разностных уравнений на сетке размера 100×100 , при этом используется представление алгоритма в форме Q -детерминанта, которое содержит только последнюю итерацию. В этом случае безусловные логические Q -термы одинаковы и имеют значительные размеры, а безусловные Q -термы небольшие, поэтому был применен второй вариант представления алгоритма в форме Q -детерминанта, содержащего условные Q -термы, и получено два файла, первый из которых содержит безусловный логический Q -терм, а второй все безусловные Q -термы. По мере формирования файлы были архивированы. В результате объем выходных данных сократился на 99,99 %.

Из-за ограничения на использование дискового пространства представления алгоритмов в форме Q -детерминанта для данного алгоритма, размер которых превышает ограничение, в исходном формате сформировать было невозможно, например, для количества итераций 4 и 5. Поэтому для табл. 6 объемы выходных данных в исходном формате для количества итераций 4 и 5 оценены аналитически. Для этого был проведен анализ струк-

Таблица 7

Результаты вычислительных экспериментов для алгоритма умножения двух плотных квадратных матриц с использованием предлагаемого стандарта

Размер матриц	Размер представления, ГБ	Количество файлов
3000	143	4800
3500	228	7200
4000	340	8400
4500	485	10800
5000	664	12600
5500	886	15600
6000	1149	18000
6500	1463	21150

туры представлений алгоритма в форме Q -детерминанта и выявлена зависимость объема выходных данных в исходном формате от параметров размерности и количества итераций. Неповторяемость безусловного логического Q -терма и архивирование выходных данных по частям позволили сформировать представления алгоритмов в форме Q -детерминанта для количества итераций 4 и 5 в новом формате.

Также были проведены вычислительные эксперименты, результаты которых могут быть использованы для решения практически значимых задач. Эти эксперименты заключаются в формировании представлений в форме Q -детерминанта алгоритма умножения двух плотных квадратных матриц с использованием схемы сдваивания для различных размеров матриц в соответствии с предлагаемым стандартом. Так как в данном случае представление алгоритма в форме Q -детерминанта содержит безусловные Q -термы, то его строки, согласно разработанному стандарту, были последовательно распределены по файлам. Следует отметить, что сформировать полученные представления без использования предлагаемого стандарта было бы невозможно, начиная с размера матриц 3500 включительно, из-за лимита дискового пространства при проведении экспериментов. Некоторые результаты экспериментов приведены в табл. 7.

Заключение. Параметры размерности алгоритмической проблемы при решении задач, применяемых на практике, могут иметь большие значения, кроме того, количество итераций при реализации итерационных алгоритмов может быть значительным. Это приводит к Q -термам большой длины и к представлениям алгоритмов в форме Q -детерминантов больших размеров. В этом случае Q -детерминанты мы называем большими, а возникающие проблемы при формировании и использовании представлений алгоритмов в форме Q -детерминантов проблемами больших Q -детерминантов. В статье описаны проблемы больших Q -детерминантов, предложены их решения и приведены экспериментальные исследования, подтверждающие правильность этих решений.

Использование предложенных решений делает возможной реализацию практических задач с помощью систем автоматизированного проектирования и исполнения программ для эффективной реализации численных алгоритмов. Одно из направлений дальнейших исследований, основанных на концепции Q -детерминанта, будет заключаться в исследовании этой возможности. Также полученные результаты можно использовать для повышения точности вычислений ресурсов параллелизма алгоритмов с помощью программной Q -системы. Кроме того, могут возникать и другие направления исследований, использующие решения, рассмотренные в данной статье.

Список литературы

1. Алеева В. Н. Анализ параллельных численных алгоритмов. Препринт № 590. Новосибирск: ВЦ СО АН СССР, 1985. 23 с.
2. Valentina Aleeva, Rifkhat Aleev. Investigation and Implementation of Parallelism Resources of Numerical Algorithms // ACM Transactions on Parallel Computing. 2023. Vol. 10. N 2, Article number 8. P. 1–64. DOI: 10.1145/3583755.
3. Алеева В. Н. Подход к эффективной реализации численных алгоритмов // Проблемы информатики. 2025. № 1. С. 29–44. DOI: 10.24412/2073-0667-2025-1-29-44.
4. Алеева В. Н., Кузнецов Е. К. Эффективная реализация алгоритмов сортировки с помощью концепции Q -детерминанта // Вестник НГУ. Серия: Информационные технологии. 2025. Т. 23, № 2. С. 5–17. DOI: 10.25205/1818-7900-2025-23-2-5-17.
5. Алеева В. Н., Сапожников А. С. Эффективная реализация алгоритмов обучения нейронных сетей с помощью концепции Q -детерминанта // Проблемы информатики. 2025. № 3. С. 5–16. DOI: 10.24412/2073-0667-2025-3-5-16.
6. Алеева В. Н. Автоматизированное проектирование и исполнение эффективных программ для численных алгоритмов // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2023. Т. 12, № 3. С. 31–49. DOI: 10.14529/cmse230303.
7. Manatin P., Aleeva V. Efficient Implementation of Numerical Algorithms Based on a Lexical Analyzer // Parallel Computational Technologies. PCT 2024. Communications in Computer and Information Science. 2024. Vol. 2241. P. 107–121. DOI: 10.1007/978-3-031-73372-7_8.
8. Алеева В. Н. Разработка программных систем автоматизированного проектирования и исполнения программ для эффективной реализации численных алгоритмов на основе концепции Q -детерминанта // Вестник НГУ. Серия: Информационные технологии. 2025. Т. 23, № 1. С. 5–18. DOI: 10.25205/1818-7900-2025-23-1-5-18.
9. Ершов Ю. Л., Палютин Е. А. Математическая логика. М.: Наука, 1987. 336 с.
10. Алеева В. Н., Зотова П. С., Склезнев Д. С. Расширение возможностей исследования ресурса параллелизма численных алгоритмов с помощью программной Q -системы // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2021. Т. 10, № 2. С. 66–81. DOI: 10.14529/cmse210205.
11. Aleeva V. N. Improving Parallel Computing Efficiency // Proceedings — 2020 Global Smart Industry Conference, GloSIC 2020. IEEE. 2020. P. 113–120. Article number 9267828. DOI: 10.1109/GloSIC50886.2020.9267828.
12. Aleeva V., Bogatyreva E., Skleznev A., et al. Software Q -system for the Research of the Resource of Numerical Algorithms Parallelism // Supercomputing. RuSCDays 2019. Communications in Computer and Information Science. 2019. Vol. 1129. P. 641–652. DOI: 10.1007/978-3-030-36592-9_52.
13. Соколов М. П. Разработка программного обеспечения для формирования Q -детерминантов численных алгоритмов с Q -термами большой длины // Вып. квалиф. работа бакалавра по направлению «Фундаментальная информатика и информационные технологии»: 02.03.02 / Южно-Уральский государственный университет. Челябинск. 2020. 32 л. [Электрон. рес.]: http://omega.sp.susu.ac.ru/publications/bachelorthesis/2020_402_sokolovmp.pdf (Дата обращения: 15 декабря 2025 года).
14. Суперкомпьютер «Торнадо ЮУрГУ». [Электрон. рес.]: <http://supercomputer.susu.ru/computers/tornado/> (Дата обращения: 15 декабря 2025 года).
15. Соколов М. П. Формирование представлений алгоритмов в форме Q -детерминантов на кластерных вычислительных системах // Вып. квалиф. работа магистра по направлению «Фундаментальная информатика и информационные технологии»: 02.04.02 / Южно-Уральский го-

сударственный университет. Челябинск. 2022. 40 л. [Электрон. рес.]: http://omega.sp.susu.ru/publications/masterthesis/2022_220_sokolovmp.pdf (Дата обращения: 15 декабря 2025 года).

16. Oseledets I., Tyrtysnikov E. TT-cross approximation for multidimensional arrays // *Linear Algebra and its Applications*, 2010. Vol. 432. N 1, P. 70–88. DOI: 10.1016/j.laa.2009.07.024.

17. Oseledets I. V. Tensor-Train Decomposition // *SIAM Journal on Scientific Computing (SISC)*, 2011. Vol. 33. N 5, P. 2295–2317. DOI: 10.1137/090752286.



Алеева Валентина Николаевна — e-mail: aleevavn@susu.ru; тел.: +7-351-267-90-89. В 1972 году окончила Новосибирский государственный университет с присвоением квалификации «Математика, прикладная математика».

В 1986 году решением Совета при Вычислительном центре СО АН СССР ей присуждена ученая степень кандидата физико-математических наук. В настоящее время она является доцентом кафедры системного программирования Южно-Уральского государственного университета (Национальный исследовательский университет).

Имеет 2 изобретения по архитектуре параллельных вычислительных систем и более 40 публикаций по распараллеливанию численных алгоритмов. Научные интересы: архитектура вычислительных систем, ресурсы параллелизма алгоритмов, проектирование эффективных программ, эффективность параллельных вычислений, в том числе в автоматизированном режиме.

Valentina Nikolaevna Aleeva — e-mail: aleevavn@susu.ru; tel.: +7-351-267-90-89. Graduated from Novosibirsk State University with the Master of Mathematics and Applied Mathematics degree in 1972 and defended his candidate dissertation in physical and mathematical sciences in Computing Centre of SB RAS in 1986. Currently, she is an Associate Professor of the Department of System Programming at South Ural State University (National Research University).

She has 2 inventions on the architecture of parallel computing systems and more than 40 publications on parallelization of numerical algorithms. The scientific interests include the architecture of computing systems, parallelism resources of algorithms, the design of effective programs, the efficiency of parallel computing, particularly in automated mode.



Соколов Михаил Павлович в 2022 году окончил Южно-Уральский государственный университет (Национальный исследовательский университет) и получил степень магистра. В настоящее время является преподавателем и аспирантом кафедры системного программирования Южно-Уральского государственного университета. E-mail: sokolovmp@susu.ru; тел.: +7-351-267-90-89.

Имеет 4 публикации по распараллеливанию алгоритмов. Научные интересы: параллельные вычисления, проектирование параллельных программ, эффективность параллельных вычислений.

Имеет 4 публикации по распараллеливанию алгоритмов. Научные интересы: параллельные вычисления, проектирование параллельных программ, эффективность параллельных вычислений.

Sokolov Mikhail Pavlovich graduated from South Ural State University (National Research University) with the Master degree in 2022. Currently, he is a teacher and postgraduate of the Department of System Programming at South Ural State University. E-mail: sokolovmp@susu.ru; tel.: +7-351-267-90-89.

He has 4 publications on parallelization of algorithms. The scientific interests include parallel computing, the design of parallel programs, the efficiency of parallel computing.

Дата поступления — 26.01.2026