Известия Саратовского университета. Новая серия. Серия: Математика. Механика. Информатика. 2025. Т. 25, вып. 2. С. 295–302
Izvestiya of Saratov University. Mathematics. Mechanics. Informatics, 2025, vol. 25, iss. 2, pp. 295–302
https://mmi.sgu.ru https://doi.org/10.18500/1816-9791-2025-25-2-295-302, EDN: ZTYLML

Article

Heuristic optimization methods for linear ordering of automata

R. A. Farakhutdinov

Saratov State University, 83 Astrakhanskaya St., Saratov 410012, Russia

Renat A. Farakhutdinov, renatfara@mail.ru, https://orcid.org/0000-0002-2877-8557, SPIN: 7667-3987, AuthorID: 1078801

Abstract. The rapid development of society is associated with two key areas of science and technology: methods of working with Big Data and Artificial Intelligence. There is a common belief that up to 80% of the data analysis process is the time spent on data preparation. One aspect of preparing data for analysis is structuring and organizing data sets (also known as data tidying). Order relations are ubiquitous, we meet them when we consider numbers, Boolean algebras, partitions, multisets, graphs, logical formulas, and many other mathematical entities. On the one hand, order relations are used for representing data and knowledge, on the other hand, they serve as important tools for describing models and methods of data analysis, such as decision trees, random forests, version spaces, association rules, and so on. Since a serious limitation of many methods of pattern mining is computational complexity, it is important to have an efficient algorithm for ordering data. In this paper, we consider deterministic automata without output signals and investigate the problem of linear ordering of such automata, which consists of building a linear order on the set of states of an automaton, that will be consistent with the action of each input signal of the automaton. To solve this problem, we consider heuristic methods of global optimization: simulated annealing method and artificial bee colony algorithm. For both methods, we made a software implementation and performed testing on a special kind of automata.

Keywords: data science, optimization, automata, linear order, simulated annealing, bee colony

Acknowledgements: The author expresses gratitude to his scientific supervisor V. A. Molchanov for the assigned task and comprehensive assistance.

For citation: Farakhutdinov R. A. Heuristic optimization methods for linear ordering of automata. *Izvestiya of Saratov University. Mathematics. Mechanics. Informatics*, 2025, vol. 25, iss. 2, pp. 295–302. https://doi.org/10.18500/1816-9791-2025-25-2-295-302, EDN: ZTYLML

Научная статья
УДК 519.688

Эвристические методы оптимизации для линейного упорядочивания автоматов

Р. А. Фарахутдинов

Саратовский национальный исследовательский государственный университет имени Н. Г. Чернышевского, Россия, 410012, г. Саратов, ул. Астраханская, д. 83

Фарахутдинов Ренат Абуханович, аспирант кафедры теоретических основ компьютерной безопасности и криптографии, renatfara@mail.ru, https://orcid.org/0000-0002-2877-8557, SPIN: 7667-3987, AuthorID: 1078801

Аннотация. Стремительное развитие общества связано с двумя ключевыми направлениями науки и технологий: методами работы с большими данными (Big Data) и искусственным интеллектом (Artificial Intelligence). Есть распространенное мнение, что до 80% процесса анализа данных — это время, потраченное на их подготовку. Одним из аспектов подготовки данных к анализу является структурирование и приведение в порядок наборов данных, так называемое data tidying. Отношения порядка встречаются повсеместно: мы встречаем их, когда рассматриваем числа, булевы алгебры, разбиения, мультимножества, графы, логические формулы и многие другие математические объекты. С одной стороны, отношения порядка используются для представления данных и знаний, с другой стороны, они служат важными инструментами для описания моделей и методов анализа данных, таких как деревья решений, случайные леса, пространства версий, правила ассоциации и т. д. Поскольку серьезным ограничением многих методов анализа шаблонов является вычислительная сложность, важно иметь эффективный алгоритм упорядочивания данных. В данной работе рассматриваются детерминированные автоматы без выходных сигналов и исследуется задача линейного упорядочения таких автоматов, заключающаяся в построении на множестве состояний автомата данного линейного порядка, который будет согласован с действием каждого входного сигнала автомата. Для решения этой задачи мы рассматриваем эвристические методы глобальной оптимизации: метод имитации отжига и алгоритм пчелиной колонии. Для обоих методов написана программная реализация и проведено тестирование на автоматах специального вида.

Ключевые слова: наука о данных, оптимизация, автомат, линейный порядок, имитация отжига, пчелиная колония

Благодарности: Автор выражает благодарность своему научному руководителю В. А. Молчанову за поставленную задачу и всестороннюю помощь.

Для цитирования: *Farakhutdinov R. A.* Heuristic optimization methods for linear ordering of automata [*Фарахутдинов Р. А.* Эвристические методы оптимизации для линейного упорядочивания автоматов] // Известия Саратовского университета. Новая серия. Серия: Математика. Механика. Информатика. 2025. Т. 25, вып. 2. С. 295–302. https://doi.org/10.18500/1816-9791-2025-25-2-295-302, EDN: ZTYLML

Introduction

Big data are large volume structured or unstructured data sets. They are processed using special dedicated tools to be used for statistics, analysis, forecasts, and decision-making. Simply stated, work with big data occurs according to the following schema:

1) information is collected from various sources;
2) the data is stored in databases and repositories;
3) the data is processed and analyzed;
4) the processed data is output using visualization tools or used for machine learning.

When analyzing big data, it is important to be able to organize data for more efficient and convenient use. The process of data sorting involves arranging the data into some meaningful order to make it easier to understand, analyze, or visualize. When working with research data, sorting is a common method used for visualizing data in a form that makes it easier to comprehend the story the data is telling. In this case, the ordering of data must be consistent with the action of their special transformations, such as reducing information to canonical form, encoding, encryption, and so on. In this regard, data can be considered as the set of states of the automaton and various special transformations can be considered as input signals of the automaton [1]. That is why for the ordering of data we consider the problem of linear ordering of deterministic automata without output signals. This problem lies in the fact that it is necessary to build such a linear order on the set of states of an automaton, which will be consistent with all input signals of this automaton. By this means, solving the problem of linear ordering of automata will make it possible to develop more efficient algorithms for studying various theoretical and applied problems, related to the development of effective algorithms and methods for analyzing structured and unstructured data of large volumes and diversity.

In [2], to solve the problem of linear ordering of automata, the authors considered a search algorithm with backtracking and cutting. Test results showed the effectiveness of this algorithm compared to the brute force algorithm. However, the complexity of this algorithm is still factorial. In this regard, interest naturally arises in heuristic algorithms for global optimization. In this paper, we consider two such algorithms: The simulated annealing method and the artificial bee colony algorithm.

1. Simulated annealing method

A simulated annealing algorithm is a method for solving various optimization problems. The exotic name of this algorithm is associated with simulation methods in statistical physics [3] based on the Monte Carlo technique. In 1953, N. Metropolis developed an algorithm for simulating the establishment of equilibrium in a system with many degrees of freedom at a given temperature [4]. The study of the crystal lattice and the behavior of atoms during the slow cooling of a body led to the emergence of probabilistic algorithms that turned out to be extremely effective in combinatorial optimization. This was first noticed in 1983 by S. Kirkpatrick [5]. Today, this algorithm is popular both among practitioners due to its simplicity, flexibility, and efficiency, and among theorists, since for this algorithm, it is possible to analytically study its properties and prove asymptotic convergence (see, for example, [6, 7]).

One of the main advantages of the annealing method is its ability to avoid getting stuck in a local minimum while continuing to search for a global minimum. This is achieved by accepting not only changes in parameters that lead to a decrease in the value of the function but also some changes that increase its value depending on the so-called temperature.

The simulated annealing algorithm belongs to the class of threshold algorithms and is used to search for the global minimum of some function $f(x)$ defined for values x from some space S. The elements of the set S represent the states of a conditional physical system (its energy levels), and the value of the function f at these points is interpreted as the energy of the system $E = f(x)$. At each moment of time, the system has a temperature T, which decreases. After generating a new state x', the system moves to the next step to state x' with probability $h(\Delta E, T)$. By ΔE we mean the increment of the energy function $f(x')$-$f(x)$, and the value $h(\Delta E, T)$ is called the probability of accepting a new state [8].

In general, the simulated annealing algorithm looks like this.

1. Randomly select a starting point x_0, $x_0 \in S$.
2. Set $x := x_0$, calculate $E := f(x)$.
3. At the ith iteration of the main loop, the following steps are performed:
   a) generate a new point $x_i := G(x, T(i))$, where $G(x, T(i))$ is a random element from S, which is selected according to the given generating family of probability distributions $\zeta(x, T)$. Calculate $E_i := f(x_i)$;
   b) compare the energy of the system E_i in state x_i with the currently found global minimum E. If $E_i < E$, then set the global minimum equal to $E := E_i$ and set the state $x := x_i$. Go to the next iteration of the loop;
   c) otherwise, generate a random number $\alpha \in [0; 1]$;
   d) if $\alpha < h(E' - E, T(i))$, then put $x := x_i$, $E := E_i$ and go to the next iteration of the loop;
   e) exit the loop if the system has reached the global minimum.

There are various schemes for choosing the parameters of the annealing method: Boltzmann annealing, Cauchy annealing (fast annealing), ultrafast annealing, Xin Yao algorithm, and others. A detailed description of each scheme, as well as its advantages and disadvantages, can be found in [8].

2. Linear ordering of automata using simulated annealing method

Let X be a non-empty set. A binary relation $\omega \in X \times X$ is called an order on the set X if it is reflexive, antisymmetric, and transitive. An order ω is called linear if for any $x, y \in X$ either $(x, y) \in \omega$ or $(y, x) \in \omega$ [9].

An ordered set is an algebraic system $(X, \leqslant)$, where X is some non-empty set, $\leqslant$ is an order on it. An ordered set with a linear order is called a linearly ordered set.

By automaton we mean an algebraic system $A = (X, S, \delta)$, where X is a finite non-empty set of states, S is a finite non-empty set of input signals, $\delta : X \times S \to X$ is a transition function.

The problem of linear ordering of a finite automaton $A = (X, S, \delta)$ is as follows: it is necessary to construct on the set of states X a linear order $\leqslant_X$, for which for any $s \in S$ from the condition $x_1 \leqslant_X x_2$ follows $\delta(x_1, s) \leqslant_X \delta(x_2, s)$ $(x_1, x_2 \in X)$.

Criteria for linear orderability of automata were investigated in [10]. Beyond that, the problem of linear ordering of automata relates to the problem of orderability of different kinds of algebras [11]. Apart from that, the problem arises in the theory of formal languages [12, 13].

Let us apply the simulated annealing method to solve the problem of linear ordering of finite automata. Let the input to the algorithm be a finite automaton $A = (X, S, \delta)$ with a set of states $X = \{x_1, x_2, \ldots, x_n\}$. The algorithm consists of the following steps.

1. Randomly generate a chain $\omega_0 := [x_{0_1} \leqslant x_{0_2} \leqslant \cdots \leqslant x_{0_n}]$ on the set of states X.
2. Calculate the initial energy $E_0 := f(\omega_0)$.
3. If $E_0 = 0$, then return ω_0 as a positive result (linear order).
4. Set $E := E_0$, $T := T_{max}$, $\omega := \omega_0$.
5. While $T > T_{min}$ and $E > 0$, do the following steps:
   a) calculate $\omega' := reverse(\omega, i_1, i_2)$, where $reverse$ is a function that reverses elements in the chain ω between two randomly selected indices i_1 and i_2;
   b) calculate $E' := f(\omega')$;
   c) if $E' < E$, then set $\omega := \omega'$ and $E := E'$, otherwise:
      – generate random number $\alpha \in [0; 1]$;
      – if $\alpha < h(\Delta E, T)$, then set $\omega := \omega'$ and $E := E'$;
   d) decrease temperature: $T := t(T)$.
6. If $E = 0$, then return ω as a positive result (global minimum, i.e. linear order), otherwise return ω as a negative result (local minimum).

The value of the target function $f(\omega)$ for a chain $\omega = [x_{i_1} \leqslant x_{i_2} \leqslant \cdots \leqslant x_{i_n}]$ in the automaton $A = (X, S, \delta)$ with n states is calculated using the following algorithm:

1) set $res := 0$;
2) in a loop for each input signal $s \in S$:
   a) calculate orbit $O_s(\omega) := (\delta(x_{i_1}, s), \delta(x_{i_2}, s), \ldots, \delta(x_{i_n}, s))$;
   б) in the orbit $O_s(\omega)$ find a number of elements k_s inconsistent with the chain ω;
   в) $res := res + k_s$;
3) return the result res.

It is worth noting that this procedure for calculating the target function allows parallelization across input signals, since for each input signal the calculation of the orbit and the counting of inconsistent elements in it do not depend on other input signals.

The temperature of the system changes according to the law $t(T) = \frac{T}{1+\alpha T}$, where α is a randomly selected coefficient from the interval $[0; 1]$.

3. Artificial bee colony algorithm

Among optimization methods, swarm intelligence algorithms are widely used, which imitate the collective behavior of complex self-organizing living systems. Bioinspired algorithms include the bee colony algorithm, which is based on modeling the behavior of a colony of honey bees when collecting nectar in nature. This algorithm was proposed in 2005 by D. Karaboga [14]. The main activity of a bee swarm is a two-stage search for optimal solutions in a certain space. For

this purpose, there are different types of bees in a bee swarm: scout bees and forager bees. In the first stage, scout bees explore the area surrounding the hive and provide other bees with information about promising places where the largest amount of nectar was found. In the second stage, forager bees fly to the vicinity of the places reported by scout bees and carry out local exploration in order to find a place richer in nectar, and scout bees continue to search for other areas.

The bee colony algorithm can be used to solve discrete (combinatorial) and continuous global optimization problems. Compared to other algorithms, the bee colony algorithm has a simpler structure, fewer control parameters, and more powerful search capabilities. In this regard, the algorithm has found wide application in different optimization problems [15–17].

In general, the bee colony algorithm consists of four phases: the initialization phase, the work phase of scout bees, the information exchange phase, and the work phase of forager bees. The bees can change their roles until the termination condition is reached.

The algorithm uses the following control parameters:

– a number of scout bees T_1, a number of forager bees T_2;
– L is a limit on the number of iterations when a forager bee does not improve the solution;
– M is the maximum number of iterations.

Main steps of the algorithm:

1) generation of a swarm of bees numbering $T = T_1 + T_2$;
2) exploration of space by scout bees from initial positions. Each scout bee generates a random solution for which the value of the target function is calculated;
3) local exploration of the vicinity of the solutions found by scout bees in order to improve them;
4) if the forager bee has improved the solution, then she communicates it to the entire swarm of bees;
5) if the forager bee has not improved the solution in L iterations, then it turns into a scout bee for one iteration in order to find a new solution, after which it returns to its role as a forager;
6) the algorithm terminates if any bee has found an optimal solution or when the maximum number of iterations M has been reached.

It should be noted that the time and complexity of executing the described algorithm directly depends on the number of iterations, the size of the bee colony, and the complexity of calculating the target function.

4. Linear ordering of automata using artificial bee colony algorithm

Let us apply the artificial bee colony algorithm to solve the problem of linear ordering of finite automata. Let the input to the algorithm be a finite automaton $A = (X, S, \delta)$ with a set of states $X = \{x_1, x_2, \ldots, x_n\}$. Let's define algorithm parameters:

$T_1' = k \times n$ $(k \in \mathbb{N})$ — the initial number of scout bees;
T_1'' — the number of scout bees after the initialization phase;
T_2 — the number of forager bees;
M — the maximum number of cycles.

The algorithm consists of the following steps.

1. For each state $x \in X$ of the automaton A, randomly generate k different chains

$$\omega_i := [x_{i_1} \leqslant x_{i_2} \leqslant \cdots \leqslant x_{i_n}]$$

on the set of states X $(1 \leqslant i \leqslant k)$. This step simulates a flight of scout bees from the initial states.
2. Calculate the target function value $E_{\omega_i} := f(\omega_i)$ for each chain ω_i, where $1 \leqslant i \leqslant T_1'$.

3. For each state $x \in X$ of the automaton A, find a chain ω_x with the best (i.e. minimum) value of the target function. Store this information in an associative array B with the key, being state x, and the value, being the best chain ω_x.
4. Define a queue Q of all states of the automaton, which will be used by forager bees.
5. While the solution is not found and the number of iterations is less than M, do:
   5.1. For each forager bee fb_i, $1 \leqslant i \leqslant T_2$:
      a) poll a state from the queue Q. Let it be x;
      b) find the currently best chain ω_x for the state x in the array B;
      c) randomly shuffle the tail of ω_x, where the tail is the last 20% of states in the chain ω_x. Let's denote the result chain as ω'_x;
      d) calculate the target function value $E_{\omega'_x} = f(\omega'_x)$;
      e) if $E_{\omega'_x} < E_{\omega_x}$, then update the array B with new chain ω'_x by the key x;
      f) push the state x back to queue Q.
   5.2. For each scout bee sb_i, $1 \leqslant i \leqslant T''_1$:
      a) randomly select some state $x \in X$ of the automaton;
      b) randomly generate a chain ω'_x on the set of states X;
      c) calculate the target function value $E_{\omega'_x} = f(\omega'_x)$;
      d) compare the target function value $E_{\omega'_x}$ of new chain ω'_x with the target function value E_{ω_x} of the best chain ω_x for the state x in the array B. If $E_{\omega'_x} < E_{\omega_x}$, then update the array B with new chain ω'_x by the key x.
6. If $E_{\omega_x} = 0$ of some state x, then return ω_x as a positive result (global minimum, i.e. linear order), otherwise return a chain with the lowest target function value.

Let us note that the value of the target function $f(\omega)$ for a chain $\omega = [x_{i_1} \leqslant x_{i_2} \leqslant \cdots \leqslant x_{i_n}]$ in the automaton $A = (X, S, \delta)$ with n states is calculated in the same way as it is described in the Section 2.

It is worth noting that the artificial bee colony algorithm allows parallelization across bees since each bee can fly independently, only a safe update of the global array B with the best chains per state is required.

5. Test results

For both considered algorithms, we implemented programs in the Java programming language. Programs were tested on automata $A = (X, S, \delta)$ with a set of states $X = \{1, 2, \ldots, n\}$ and transition functions U_i, $1 \leqslant i \leqslant n-1$, V_k, $2 \leqslant k \leqslant n$, which are determined by the following formulas:

$$U_i(j) = \begin{cases} i+1 \text{ for } j = i, \\ j \text{ otherwise,} \end{cases} \qquad V_k(j) = \begin{cases} k-1 \text{ for } j = k, \\ j \text{ otherwise.} \end{cases}$$

It is known [18] that the transformations U_i, V_k generate the entire semigroup of endomorphisms of a linearly ordered set. Consequently, such automata are automata with the most complex semigroup of input signals, moreover, they definitely have a linear order on the set states (for example, natural order on the set $\{1, 2, \ldots, n\}$).

The program of the linear ordering of automata using the simulated annealing method runs at $T_{max} = 100$, $T_{min} = 0,0000001$. To measure the minimum execution time for each number of states of the automaton, the program was launched simultaneously on all available logical processors.

The program of the linear ordering of automata using the artificial bee colony algorithm run with parameters $k = 50$ (i.e. $T'_1 = 50 \times n$), $T''_1 = 0.3 \times P$, $T_2 = 0.7 \times P$, where P is a number of logical processors, $M = 2^{32}$. For this algorithm, we used parallel implementation, which allows the execution of flights of different kinds of bees independently, based on the number of available logical processors P.

Testing was carried out using an Intel Core i7-6500U 2.50 GHz processor. The test results are presented in the Table.

Table. Test results of different programs of linear ordering of automata

Amount of states	Backtracking algorithm execution time, sec	Average execution time of simulated annealing algorithm, sec	Minimum execution time of simulated annealing algorithm, sec	Execution time of artificial bee colony algorithm, sec
10	0.9	0.918	0.007	0.162
20	44.4	3.173	0.117	2.137
30	387.7	7.319	0.150	4.538
40	1883.9	10.848	0.654	9.855
50	6016.6	25.606	1.693	8.146
60	9654.8	104.745	4.159	13.56
70	32510.4	436.278	9.558	84.951
80	58627.9	561.394	20.526	71.325
90	79478.2	867.496	27.269	544.047
100	90176.4	3857.963	53.977	1297.260
150	—	11452.712	410.033	7859.972
200	—	14859.536	1837.811	11635.203
300	—	18769.451	13817.571	21851.645

The test results show a significant superiority of considered heuristic optimization methods compared to the brute-force algorithm with backtracking and cutting. It is easy to see, that parallel implementation of the simulated annealing method is more efficient than the parallel implementation of the artificial bee colony algorithm. Thus, the heuristic optimization methods considered are effective ways to solve the problem of linear ordering of finite automata.

References

1. Evsyutin O. O., Rossoshek S. K. Use of cellular automatons for problems solving of information transformation. *Doklady TUSUR*, 2010, vol. 21, iss. 1–1, pp. 173–174 (in Russian).
2. Molchanov V. A., Farakhutdinov R. A. Linear ordering of automata. *Matematika. Mekhanika* [Mathematics. Mechanics], 2019, vol. 21, pp. 45–48 (in Russian).
3. Binder K. *Monte Carlo methods in statistical physics*. Berlin, Springer, 1979. 376 p.
4. Metropolis N., Rosenbluth A. W., Rosenbluth M. N., Teller A. H., Teller E. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 1953, vol. 21, pp. 1087–1092. https://doi.org/10.1063/1.1699114
5. Kirkpatrick S., Gelatt C. D. Jr., Vecchi M. P. Optimization by simulated annealing. *Science*, 1983, vol. 220, pp. 671–680.
6. Savin A. N., Timofeeva N. E. The application of optimization algorithm using simulated annealing method for parallel computing systems. *Izvestiya of Saratov University. Mathematics. Mechanics. Informatics*, 2012, vol. 12, iss. 1, pp. 110–116 (in Russian). https://doi.org/10.18500/1816-9791-2012-12-1-110-116, EDN: OUPILL
7. Doerr B., Rajabi A., Witt C. Simulated annealing is a polynomial-time approximation scheme for the minimum spanning tree problem. *Algorithmica*, 2024, vol. 86, pp. 64–89. https://doi.org/10.1007/s00453-023-01135-x
8. Lopatin A. S. Simulated Annealing Method. *Stokhasticheskaya Optimizatsiya v Informatike*, 2005, vol. 1, pp. 133–149. (in Russian). EDN: KYIMEB
9. Bogomolov A. M., Saliy V. N. *Algebraicheskie osnovy teorii diskretnykh sistem* [Algebraic foundations of the theory of discrete systems]. Moscow, Nauka, 1997. 368 p. (in Russian).
10. Kats M. M. Criterion for linear ordering of a partial automaton. *Izvestiya Vysshih Uchebnyh Zavedenij. Matematika*, 1997, iss. 10, pp. 37–43 (in Russian). EDN: HQUVBJ

11. Gabovich E. Ya. Fully ordered semigroups and their applications. *Russian Mathematical Surveys*, 1976, vol. 31, iss. 1, pp. 147–216. https://doi.org/10.1070/RM1976v031n01ABEH001447
12. Klíma O., Polák L. On varieties of ordered automata. In: Martin-Vide C., Okhotin A., Shapira D. (eds.) *Language and automata theory and applications. LATA 2019*. Lecture notes in computer science, vol. 11417. Springer, Cham, 2019, pp. 108–120. https://doi.org/10.1007/978-3-030-13435-8_8
13. Cotumaccio N., D'Agostino G., Policriti A., Prezza N. Co-lexicographically ordering automata and regular languages. Part I. *Journal of the ACM*, 2023, vol. 70, iss. 4, pp. 1–73. https://doi.org/10.1145/3607471
14. Karaboga D. An idea based on honey bee swarm for numerical optimization. *Technical Report, Erciyes University*, 2005. Available at: https://abc.erciyes.edu.tr/pub/tr06_2005.pdf (accessed November 22, 2023).
15. Pham D. T., Castellani M. The bees algorithm: Modelling foraging behaviour to solve continuous optimization problems. *Proceedings of the Institution of Mechanical Engineers. Part C: Journal of Mechanical Engineering Science*, 2009, vol. 223, iss. 12, pp. 2919–2938. https://doi.org/10.1243/09544062JMES1494
16. Cuevas E., Sencion-Echauri F., Zaldivar D., Perez-Cisneros M. Multi-circle detection on images using Artificial Bee Colony (ABC) optimization. *Soft Computing*, 2012, vol. 16, iss. 2, pp. 281–296. https://doi.org/10.1007/s00500-011-0741-0
17. Toktas A. Multi-objective design of multilayer microwave dielectric filters using artificial bee colony algorithm. In: Carbas S., Toktas A., Ustun D. (eds.) *Nature-Inspired Metaheuristic Algorithms for Engineering Optimization Applications*. Springer Tracts in Nature-Inspired Computing. Springer, Singapore, 2021, pp. 357–372. https://doi.org/10.1007/978-981-33-6773-9_16
18. Aizenshtat A. Ya. The defining relations of the endomorphism semigroup of a finite linearly ordered set. *Sibirskii Matematicheskii Zhurnal*, 1962, vol. 3, iss. 2, pp. 161–169 (in Russian).

Received / Поступила в редакцию 22.11.2023
Accepted / Принята к публикации 04.03.2024
Published online / Опубликована онлайн 30.05.2025