# Geometric and game approaches for some discrete optimization problems

**B F Melnikov[1], E A Melnikova[1], S V Pivneva[1], V A Dudnikov[2] and E V Davydova[3]**

[1]Russian State Social University, Wilhelm Pieck str. 4, Moscow, Russia, 129226
[2]Togliatti State University, Belorusskaya str. 16, Togliatti, Russia, 445020
[3]Moscow Aviation Institute (State Technical University), Volokolamskoe shosse 4, Moscow, Russia, 125993

**Abstract.** We consider in this paper the adaptation of heuristics used for programming non-deterministic games to the problems of discrete optimization. In particular, we use some "game" heuristic methods of decision-making in various discrete optimization problems. The object of each of these problems is programming anytime algorithms. Among the problems described in this paper, there are the classical traveling salesman problem and some connected problems of minimization for nondeterministic finite automata. The first of the considered methods is the geometrical approach to some discrete optimization problems. For this approach, we define some special characteristics relating to some initial particular case of considered discrete optimization problem. For instance, one of such statistical characteristics for the traveling salesman problem is a significant development of the so-called "distance functions" up to the geometric variant such problem. And using this distance, we choose the corresponding specific algorithms for solving the problem.

Besides, other considered methods for solving these problems are constructed on the basis of special combination of some heuristics, which belong to some different areas of the theory of artificial intelligence. More precisely, we shall use some modifications of unfinished branch-and-bound method; for the selecting immediate step using some heuristics, we apply dynamic risk functions; simultaneously for the selection of coefficients of the averaging-out, we also use genetic algorithms; and the reductive self-learning by the same genetic methods is also used for the start of unfinished branch-and-bound method again. This combination of heuristics represents a special approach to construction of anytime-algorithms for the discrete optimization problems. This approach can be considered as an alternative to application of methods of linear programming, and to methods of multi-agent optimization, and also to neural networks.

## 1. Introduction. A brief survey of discrete optimization problems

We consider in this paper the adaptation of heuristics used for programming non-deterministic games to the problems of discrete optimization, in particular, some heuristic methods of decision-making in various discrete optimization problems (DOP). The object of each of these problems is programming anytime algorithms, i.e., the algorithms, which can provide a near-to-optimal solution in real time. The basic purposes of the paper are practical questions of construction of algorithms, as well as the creation of the corresponding theory. Let us briefly list the considered problems, more precisely, the classes of considered problems.

*First*, it is the classical traveling salesman problem (TSP; [1] etc.). Certainly, an universal methods for solving TSP simply cannot exist. Some last years, the authors of papers for heuristic methods of TSP-solution consider most often so-called *metric TSP*. For their solving, some methods (of linear programming, of multi-agent optimization, etc.) are used; [1, 2, 3, 4, 5] etc. However, the classical branch-and-bound method (BBM) can also be used not only for the exact (optimal) solution of considered TSP, but also for quasi-optimal heuristic solutions. We shall write below about these things more detailed.

*Second*, these are some related problems of minimization for nondeterministic finite automata (Rabin-Scott automata, NFA). Probably, the main for them is state-minimization, i.e., the problem of constructing NFA, which defines the given regular language and has minimum possible number of states. Since 1970 (i.e., since [6]), there are a few changes in description of the exact algorithms for this problem: all the algorithms are exponential relative to the number of states of considered NFA. The last argument is true because all the algorithms need to construct equivalent automaton of canonical form (or, maybe, some similar graphs or other objects). Let us remark, that from the point of view of the theory of complexity of algorithms, all the algorithms of [6, 7, 8, 9, 10, 11] are equivalent.

Besides, there are other problems for NFA-minimization, the following ones:

- edge-minimization [12];
- and also the star-height-minimization. There exists two solutions of the last problem ([13], and also [14] with the simplification of the proof [15]). However, the authors think that there is impossible *to make a computing algorithm* on basis of these papers.

*Third*, this is the problem of minimization of disjunctive normal forms (DNF). The exact algorithms for this minimization are obtained for ages (and are considered in the classical textbooks for first-year students), however the computer programs making on basis of such solutions cannot work in real time even for the number of variables, which is equal to 20, except, certainly, for a lot of trivial cases. The unified approach of this paper is used in some versions of computer programs.

Let us remark, that, certainly, these groups of problems do not formulate the whole set of problems, which can be heuristically solved by the methods considered in this paper. Let us also mention, for instance, [16]. Some other groups of problems are given in the conclusion, and, probably, each DOP can be solved in such a way.

The methods of solution DOP, considered in this paper, are constructed on the basis of special combination of some heuristics, which belongs to some different areas of the theory of artificial intelligence. Firstly, we shall use some modifications of unfinished branch-and-bound methods. Secondly, for the selecting immediate step using some heuristics, we use dynamic risk functions. Thirdly, simultaneously for the selection of coefficients of the averaging-out, we also use genetic algorithms. Fourthly, the reductive self-learning by the same genetic methods is also used for the start of unfinished branch-and-bound method. Let us consider now the detailed description of these heuristic methods of DOP-solution.

## 2. Traveling salesman problem and its quasi-metric variant

Often, a mathematical model, as well as algorithms based on this model, created for one area, find application in many other subject areas. As we said before, an example of such a model is the traveling salesman problem. The peculiarity of this problem is that, with the relative simplicity of its formulation, finding the optimal solution (the optimal route) is a very complex problem and relates, both in its generalized formulation and for most of its variations, to the NP-complete class. Moreover, according to the classification given in [1] etc., the traveling salesman problem is an example of the optimization problem included in the most complex NPO(V) class: it contains all optimization problems for which (with some additional “natural” assumption, for example, $P \neq NP$), the time complexity of all possible polynomial algorithms cannot be limited by any polylogarithmic function. (Another example of such a problem is the maximum clique problem.)

Among many various versions of TSP, one of the most studied is the *geometric* version (also referred as *Euclidean*, see also [1]): the cost of the route is equal to the distance between points ("cities") on the plane, calculated as the Euclidean norm. For further discussion, that a characteristic feature of this problem is the fulfillment of the triangle inequality for any three cities. Namely, for any $u, v, w \in E$ the following inequality holds:

$$c(\{u,v\}) \leq c(\{u,w\}) + c(\{w,v\}).$$

We note that this formula assumes only so-called *symmetric* variants of TSP (this term is also defined in [1]); however, in *not* symmetric variants, everything is the same. In general, according to [1], the variants of the problem of TSP with the triangle inequality satisfied for any three cities are called *metric*, that is, all geometric TSP represent a proper subset of metric ones.

However, the main research of the authors of this paper is aimed at studying not geometric, but so-called *pseudo-geometric* version of TSP, [17] etc. In it, the input data is formed as follows. To the data of some *predefined* TSP, which is *geometric*, a vector

$$R = (r_1, \ldots, r_m), \qquad (\text{where} \quad m = |E|\,)$$

is added. Here, all the values $r_i$ are independent and identically distributed random variables (IID); we consider normal distribution only, and allow $\mu = 1$ and some predefined "acceptable" value $\sigma$. In this case, each of the elements of the value matrix (i.e., $c(\{u,v\})$; let it be $c_i$ for some $i \in \{1, \ldots, m\}$) is changed for the following value:

$$\max(c_i \cdot r_i, 0).$$

An important difference between the pseudo-geometric version of TSP and the geometric one is the possibility of violating the triangle inequality for some triples of cities. It is also important to note, that the geometric version of TSP can be considered as a special case of pseudo-geometric one (with the value $\sigma = 0$).

Here is our view on the "ecological niche" of the version of the traveling salesman problem we are considering. As we said before, the geometric version of TSP defined above is one of the most studied; as a result of theoretical and practical research, many approaches to solving its particular cases have been developed. Among them, one can single out the so-called geometric approaches using in the search for the solution information that the coordinates of the cities *were previously obtained* as a particular case of a geometric TSP. As examples of algorithms belonging to a group of such geometric approaches and described 15 and more years ago, we can cite the so-called "onion-peeling" algorithms, see [18], and also "elastic network" ones, see [19]. With a small computational complexity, these approaches make it possible to obtain a fairly good solution: for particular cases containing millions of points, it practically coincides with the optimal one. However, outside the geometric version of TSP, these algorithms usually turn out to be meaningless; and this paper can be considered as an attempt to apply similar algorithms to other versions of TSP.

Thus, for TSP, we use the following names:

- "accidental TSP", when all the elements of the TSP-matrix are generated by the variate having the given equipartition law;
- "metric TSP", when we consider towns as the accidental points of the unit square (both the coordinate have the given equipartition law), and the elements of the TSP-matrix are their distances. And here is also evident the following symmetric condition: $a_{ij} = a_{ji}$ for each possible $i$ and $j$.

- "quasi-metric TSP", when all the elements of the metric TSP-matrix are multiplied a posteriori by a random number, which is obtained by a given normal low.

Some last years, the papers for metric TSP are mostly published. The consideration of metric TSP as the problems of linear programming or the applications of so-called methods of multi-agent optimization was started long before 2000, [20, 21]. And the quasi-metric TSP, which is almost not considered, is more interesting, because of the following:

- first, it is more closely to various practical problems;
- second, various heuristics can be checked up here, which are not connected to use of an arrangement of cities on a plane; moreover, the reduction of this problem to a problem of linear programming is here ineffectively;
- and third, the simplification of the TSP-matrix by one step of BBM is here "less significant", than in other TSP-variants.

Therefore, the metric TSP is the most important scope for algorithms considered in this paper.

However, in [1], the only exact BBM was considered; it finishes by constructing the optimal solution. And in practice, we can rarely obtain the exact TSP-solution using only algorithms of [1]. Using some special programming techniques (e.g., special data structures for quick making the next step of BBM, organization of swapping by the programmer), we can only a little improve the situation. In fact, each of such programming techniques is a new heuristics, which is used in addition to considered applying BBM. However, we are considering the exact solution of TSP (and other DOP) only, and, for now, are not considering the things connected with anytime algorithms. Before the formulation such algorithms, let us consider the following definitions.

Considering a BBM-step, we have to designate the problem for the next solution, obtained by reduction of dimension, exactly *the right problem* (like [1] etc.). For example, the right problem of TSP is obtained when we include the edge between two considered towns; and the right problem of NFA-minimization is obtained when we include the selected grid (see [9] etc.). The other alternative, i.e., when we make a decision about the absence of some element in the optimal solution (e.g., if we consider TSP not containing the edge between two considered towns) is designated by *the left problem.* It is evident, that the object of each modification of BBM (for each DOP) is to obtain the case, when the probability of belonging the optimal solution in the right problem is more then the same probability for the left problem; *we make this thing using some special heuristics.* The explanation of this fact is trivial: dimension of the right problem is less than the left one.

The simple heuristics, which reforms the usual BBM into the unfinished one (and on basis of unfinished BBM we construct any anytime algorithms in this paper), is the following. Each time, when we obtain the next right problem (let us call it problem $T$) we make at the same time also the sequence of the right (sub)problems (SRP), i.e., $T$, then the right problem of $T$, then the right problem of the right problem of $T$, etc. Certainly, we make each time also the corresponding left problems, i.e., the left problem of $T$, then the left problem of the right problem of $T$, etc. This process finishes:

- when we obtain a trivial problem (e.g., of dimension 1), then we use its solution (i.e., its bound, and also the obtained path, and similar behavior) by the current quasi-optimal solution of the considered anytime algorithm;
- or when we obtain the big value of the bound, for example, if this value is more than the current (existing) quasi-optimal solution.

Let us remark, that in practice such process of SRP-constructing does not require a lot of time, and the increasing the dimension of the list of problems for the solution in the future is very reasonable.

Thus, we have described the simple process of constructing the anytime algorithm on the basis of the given version of BBM. And it is unlikely, that such process is described here at the first time (it is really very simple), however, the authors have not the references for this thing. Let us also remark, that this algorithm of SRP-constructing is used as the sub-algorithm not only for the unfinished BBM (like this section), but also for so called algorithm of tournament self-learning.

## 3. Nondeterministic games and dynamic estimation of a position. Dynamic risk functions in games and in discrete optimization

Because of space limitations, the rules of backgammon are not presented in the paper. Among scientific works devoted to programming of this game, we mention the papers [22, 23, 24]. However, the authors of this paper hope, that the use of dynamic risk functions (DRF) considered here simplifies conventional methods of neural network programming and learning; they are an alternative to these methods.

What is the difference between backgammon (and other nondeterministic games) and, to say, chess (or other deterministic games) from the programming standpoint? The difference is that the game tree constructed for backgammon includes not only the vertexes where the players choose a next move but also those where they wait for a particular realization of a random event. Therefore, the standard minimax method is to be generalized for programming of search in nondeterministic games. In this paper, we will only briefly describe this generalization.

We assume that the reader knows the canonical minimax method. And in nondeterministic games, we have the following alternate actions:

- a particular realization of a certain random event;
- a move of one player;
- another realization of the random event;
- and a move of the other player.

The number of possible outcomes of the random event is to be finite (otherwise, we need different models). As a result, the game tree contains additional levels between those corresponding to the players' moves. These new levels correspond to the moments when the random event is realized. It is such a tree that the generalization of the minimax method.

Assume that we can construct a static estimator of a position. By temporally eliminating indeterminacy, we preliminary estimators of the game tree positions. For this purpose, we assume first that a particular outcome of the random event has been already realized and calculate the dynamic estimator of the position in the same way as in the conventional minimax method. Then, we calculate the dynamic estimator for the next outcome of the random event, and so on for all possible outcomes.

The final dynamic estimator of the position is based on the deterministic estimators of all possible outcomes of the random event. The values of the deterministic (usually, static) estimators are averaged in a special way resulting in the dynamic estimator. From the physical standpoint, such averaging gives us the coordinate of the gravity center of a one-dimensional system of masses whose values are determined by a specially chosen function (risk function). Coordinates of the masses are equal to the values of the corresponding deterministic estimator, which is determined by only deterministic factors of the game, like in the conventional minimax. Let ai be values of deterministic estimators and $f$ be a risk function. Then, according to [25, 26], the dynamic estimator is calculated by the formula

$$\frac{\sum_{i=1}^{k} a_i \cdot f(a_i)}{\sum_{i=1}^{k} f(a_i)} \, .$$

Let us note, that purpose of the paper [25] was to generalize the minimax method. The important thing to note, however, is that the program based on this generalization only, with the simplest risk function for static estimation of a backgammon position, showed good results and won most part of the programs that the authors of [25] could find that time in Internet. This program has been gradually improving since then. Note also, that the ways of improving the program were very different from those discussed in "classical" works on programming of backgammon [22, 23, 24]. In the latter works, one or another way of calculation of static estimators of a position is optimized. Some ways to improve programs, which were used by the authors after the simplest dynamic estimator had been already introduced, are described in this paper. Note that they concern not only improvements of the static estimator of a position.

The question is to what extent the weights of the opponent's casts that are favorable for us are to be reduced? Even if we simply take the risk function $y = 1 - 0.4x$ and use this function independent of any other circumstances with the simplest static position estimator, we will get rather good results. The short description of practical results can be found in [25]. Note also that in that work, different decreasing risk functions were considered.

But to get a stronger program, it is required *to change* strategies $y = 1 - 0.4x$ during the game. One way to improve the dynamic estimator of a position (described in [25]) is as follows. We qualitatively estimate the position (whether we are about to win or to loose). Then:

- if we are about to win or loose a little, we should be pessimists and adopt a risk function similar to above-mentioned function $y = 1 - 0.4x$;
- if we loose more, the risk function should be close to constant one;
- and if the loss is great, the risk function should be increasing; in this case we need to be super-optimists and hope against hope (what else can we do?).

Of course, there are many other, intermediate, variants of risk functions. And a possible approach to dynamic selecting these intermediate variants was described in detail in [26]. Thus, the authors of this paper believe that, in the given case, the methods of modifications of plots of risk functions simplify conventional methods of neural network programming and learning. This follows, for example, from the fact that one of the authors created a good program for playing backgammon using less than 3 self-learning coefficients, whereas programs described in [23, 24] (see also above-mentioned web sites) use several hundreds such parameters for neural networks.

## 4. Geometric approach based on the classification of input data

Usually, when developing algorithms, an idealized model of input data is used, most often it is a model with uniform or normal distributions of values of specially allocated *characteristics* of the problem under consideration. However, in practice, the input data, as a rule, come in accordance with some other probability distribution, i.e. distribution, other than uniform or normal. As a result, the performance of the algorithm on real data is calculated inadequately to the input data; moreover, this situation occurs in both the average and the worst case.

Another situation is possible: when the algorithm is designed to take into account the characteristics of the input data that are characteristic of one subject area; and the application of such algorithms to another domain can be extremely inefficient due to the fact that the characteristic features of real data turn out to be different from those considered in the development of the algorithm.

Many scientists are working on a *theory* describing algorithms for a supposedly correct distribution in each particular case, which would sufficiently adequately describe many practical situations. In the opinion of the authors of this paper, the greatest progress in this direction was achieved in the works of Yu. Gurevich ([27] etc.). In some cases, the evaluation of representativeness may not be an end in itself, but should serve as a basis for approaches to evaluating the effectiveness of algorithms.

In the opinion of the authors of this paper, some specific models are needed for each problem under consideration, i.e., the *concrete interpretation* of similar approaches, the so-called. methods "ad hoc". To implement this approach, we have defined special characteristics relating to some initial particular case of TSP; it should be noted that they were chosen similarly to the characteristics used in [28] for a completely different discrete optimization problem. The statistical characteristics used by us for TSP, in fact, are a significant development of the so-called "distance functions" *up to the geometric variant* of TSP $dist(G,c)$, described in [1, Ex. 4.2.3.2]; we can say that these characteristics reflect the cases of non-fulfillment of this inequality in the given particular case of TSP. [1]

For their calculation, similarly to the above example of [1], we considered all possible pairs of points $u, v \in V$ $(u \neq v)$. For each of these pairs and each point $p \in V$, the value

$$\max\Bigl(0, \frac{c(\{u,v\})}{c(\{u,p\}) + c(\{p,v\})}\Bigr), \quad p \neq u, \ p \neq v$$

was calculated. For a collection [2] $D$ of such values, and also for the collection $\mathcal{D}$, composed of non-zero elements of collection $D$, we considered the following characteristics:

- $|D|/N$, where $N$ is the maximum possible number of non-zero elements [3] of collection $D$, i.e.,
$$\frac{n \cdot (n-1) \cdot (n-2)}{2};$$
- $|\mathcal{D}|/N$;
- the expected value of $D$, where $D$ is considered as a realization of a random variable;
- the expected value of $\mathcal{D}$;
- the variance of $\mathcal{D}$; [4]
- the average harmonic of the collection $\mathcal{D}$;
- the median of the collection $\mathcal{D}$;
- value of the Durbin – Watson statistic ([29, 30]) for several variants of collections $\mathcal{D}$ (i.e., if there are different inputs).

Based on these characteristics with a perceptron ([31] etc.), a decision was taken on the class for which the particular case of TSP was generated.

The specific description of computational experiments is as follows. *Previously*, for the chosen dimension of the problem, a collection of these characteristics was created (in this case, we have always considered 70 cities); for them, we used various values $\sigma$, namely, from 0.00 to 2.20 with the step 0.01. Each of the characteristic values was taken equal to the average of the corresponding characteristics, obtained for 10 various random generations of special cases. At the same time, a self-study of the perceptron was carried out, which classified the special case under consideration on the basis of the obtained characteristic vector, i.e., giving an exit on the assumption of the original value $\sigma$.

Further, for each of the following values $\sigma$:

$$\sigma = 0.00\,; \quad \sigma = 0.20\,; \quad \sigma = 0.50\,; \quad \sigma = 0.90\,; \quad \sigma = 1.40\,; \quad \sigma = 2.00$$

[1] Note that *in practice* the most interesting application of the pseudo-geometric version of TSP in the case when the dimension of the problem is approximately 100. (The complexity of the algorithm used to calculate all these characteristics is $O(n^3)$, where, as before, $n$ is the dimension of the problem.)

[2] The elements of a collection can be repeated.

[3] As we said before, we considered a symmetric TSP.

[4] For this and the following characteristics, it hardly makes sense to consider their analogues for the collection $D$.

we made 100 experiments. In each of them, we generated a special case of TSP, and after that *based on the characteristic vector obtained for this special case*, we selected the assumption of the initial value $\sigma$ in two ways:

- based on the maximum probability value given by a previously trained perceptron;
- using the method of least squares.

The results of the experiments are summarized in the following Table 1. In it, the first column (P) corresponds to the results obtained with the perceptron. Each cell contains a segment of the values $\sigma$, into which the results were obtained, as well as the number of cases (from 100 observed), in which the value $\sigma$ differs from the initial value by not more than 10%.

Table 1.

| $\sigma$ | P | MLS |
|---|---|---|
| 0.00 | $[0.00, 0.00]$ 100 | $[0.00, 0.00]$ 100 |
| 0.20 | $[0.16, 0.23]$ 83 | $[0.16, 0.24]$ 81 |
| 0.50 | $[0.43, 0.59]$ 81 | $[0.42, 0.58]$ 80 |
| 0.90 | $[0.76, 1.03]$ 79 | $[0.76, 1.02]$ 79 |
| 1.40 | $[1.20, 1.57]$ 78 | $[1.22, 1.55]$ 78 |
| 2.00 | $[1.74, 2.20]$ 79 | $[1.76, 2.20]$ 79 |

Thus, apparently, we can say that the *both* versions of the classification give acceptable results, which practically do not depend on the specific method of classification (i.e., using a neural network or using the method of least squares). It is also important that both algorithms can be considered a *combination* of:

- the *rule-based approach*;
- the *alternative-based approach*, for which there is as yet no final name. [5]

A rule-based approach is to select the characteristics of the *expert*, and an alternative approach is to a posteriori use of the neural network or the method of least squares.

However, both the more detailed results of the above computations and the possible improvement of both classification algorithms that we have applied are hardly of great interest: as already noted, these calculations (that is, the classification) are regarded as an auxiliary task for another, more important task (i.e., the continuation of the *geometrical solving* of pseudo-geometric TSP), which we propose to consider in subsequent publications.

## 5. Conclusion

Let us mention another difficult problem for the following solution. For different sub-classes of problems, we try to use different genomes (using different genetic algorithms for the self-learning is also possible); they are analogues of classes of positions in nondeterministic games. But this is a simple problem, rather, a problem, which main complication belongs not to the programmer, but to the experts (who understand the whole specificity of the considered DOP). There would be much more important a possibility of automatic generating conditions for belonging some DOP to a class of problems, which should be considered separately from other classes (let us

[5] The final name is not in the literature, not only in Russian, but also in English: the often used words "not rule-based approach" are unlikely to be "claimed" for it. Possible alternatives to a rule-based approach, *in different subject areas*, are the so-called. a connectionism approach, a structural approach, and, for example, in the case of machine translation, the so-called statistical translation..

In connection with the last, it should be noted its long-standing use in the service "Yandex.Translate" [32], as well as the emergence of recently *hybrid* translation systems, actually being a *combination* of the two above-mentioned approaches.

also remark, that in the case of programming intelligent games, this problem is connected with a problem of automatic generating a new parameter for the function of static estimating position). After such automatic generating we use the usual self-learning by some genetic methods, and then we make a testing, whether we really have constructed a new class of sub-problems for considered DOP. Let us remark, that the possible algorithms of such testing are evident (they are similar to usual algorithms of clustering), and the first part of this problem, i.e., automatic generating conditions of a class of problems, is much more important.

Let us remark also the following fact. Using heuristics of this paper in various DOP, the authors have none example, when the optimal solution needs more than 5% of the time required for the common solution of the considered problem. This fact marginally explains all the heuristics considered in this paper, even the optimum solution is unknown, e.g., if it cannot be obtained in the reasonable time.

Here are the links to recent papers related to the subject matter discussed in this paper:[33, 34]. We also cite some recent works by the authors of this paper [35, 36, 37, 38, 39, 40].

## 6. References

[1] Hromkovič J 2004 *Algorithmics for Hard Problems. Introduction to Combinatorial Optimization, Randomization, Approximation, and Heuristics* (Berlin: Springer) p 548

[2] Gutin G and Punnen A 2002 The Traveling Salesman Problem and its Variations (Berlin: Springer) p 829

[3] Applegate D, Bixby R, Chv´atal V and Cook W 2007 *The Traveling Salesman Problem. A Computational Study* (Princeton: Princeton University Press) p 608

[4] Gutin G, Karapetyan D 2010 A memetic algorithm for the generalized traveling salesman problem *Natural Computing* **9(1)** 47-60

[5] Raman G and Gill N 2017 Review of different heuristic algorithms for solving Travelling Salesman Problem *Int. J. of Advanced Research in Computer Science* **8(5)** 423-449

[6] Kameda T and Weiner P 1970 On the state minimization of nondeterministic finite automata *IEEE Transactions on Computers* **C-19** 617-627

[7] Hashiguchi K 1991 Algorithms for determining the smallest number of nonterminals (states) sufficient for generating (accepting) a regular language *Automata, Languages and Programming Lecture Notes in Computer Science* **510** 641-648

[8] Jiang T and Ravikumar B 1993 Minimal NFA problems are hard *SIAM J. Comput.* **22(6)** 1117-1141

[9] Melnikov B 2000 Once more about the state-minimization of the nondeterministic finite automata *Journal of Applied Mathematics and Computing* **7(3)** 655-662

[10] Pol´ak L 2005 Minimizations of NFA using the universal automaton I*nternational Journal of Foundations of Computer Science* **16(5)** 999-1010

[11] Han Y-S 2013 State elimination heuristics for short regular expressions *Fundamenta Informaticae* **128** 445-462

[12] Melnikov B 2010 Once more on the edge-minimization of nondeterministic finite automata and the connected problems *Fundamenta Informaticae* **104(3)** 267-283

[13] Hashiguchi K 1988 Algorithms for determining relative star height and star height *Information and Computation* **78** 124-169

[14] Kirsten D 2005 Distance desert automata and the star height problem *Theoretical Informatics and Applications* **39** 455-509

[15] Bojanczyk M 2015 Star Height via Games *30th Annual ACM/IEEE Symposium on Logic in Computer Science* **104(3)** 214-219

[16] Melnikov B and Panin A 2012 On a parallel implementation of the multi-heuristic approach in the problem of comparison of genetic sequences *Vector Science of Togliatti State University* **4(22)** 83-86 (in Russian)

[17] Makarkin S and Melnikov B 2013 Geometrical methods of solving pseudo-geometrical version of traveling salesman problem *Stochastic optimization in informatics* **9(2)** 54-72 (in Russian)

[18] Liew S 2012 Introducing convex layers to the Traveling Salesman Problem *Preprint arXiv:1204.2348*

[19] Somhom S, Modares A and Enkawa T 1999 Competition-based neural network for the multiple travelling salesmen problem with minimax objective *Computers & Operations Research* **26(4)** 395-407

[20] Dorigo M and Gambardella L 1997 Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem *IEEE Transactions on Evolutionary Computation* **1(1)** 53-66

[21] Johnson D and McGeoch L 1997 The Traveling Salesman Problem: A Case Study in Local Optimization *Local Search in Combinatorial Optimization* 215-310

[22] Berliner H 1980 Computer Backgammon *Scientific American* **243** 64–72

[23] Tesauro G and Sejnowski T 1989 A Parallel Network that Learns to Play Backgammon *Artificial Intelligence* **39** 357-390

[24] Tesauro G 1985 Temporal Difference Learning and TD-Gammon *Temporal Difference Learning and TD-Gammon* **38(3)** 58-68

[25] Melnikov B and Radionov A 1998 A choice of strategy in nondeterministic antagonistic games *Programming and Computer Software* **24(5)** 247-252

[26] Melnikov B 2001 Heuristics in programming of nondeterministic games *Programming and Computer Software* **27(5)** 277-288

[27] Gurevich Y and Veanes M 2007 Can abstract state machines be useful in language theory? *Theoretical Computer Science (Developments in Language Theory)* **376(1-2)** 17-29

[28] Melnikov B, Pivneva S and Rogova O 2010 The representativeness of randomly generated nondeterministic finite automata from the point of view of the corresponding basis automata *Stochastic optimization in informatics* **6(1)** 74-82 (in Russian)

[29] Watson G 1962 Goodness-of-fit tests on a circle *Biometrika* **48(1-2)** 109-114, **49(1-2)** 57-63

[30] Lagutin M 2012 *Visual mathematical statistics* (Moscow: Binom) p 472 (in Russian)

[31] Haikin S 2008 *Neural networks: the full course* (Moscow: Williams) p 1104 (in Russian)

[32] Yandex.Translate (Access mode: https://en.wikipedia.org/wiki/Yandex.Translate)

[33] Evdokimova N I and Kuznetsov A V 2017 Local patterns in the copy-move detection problem solution *Computer Optics* **41(1)** 79-87 DOI: 10.18287/2412-6179-2017-41-1-79-87

[34] Evsutin O O, Shelupanov A A, Meshcheryakov R V and Bondarenko D O 2017 An algorithm for information embedding into compressed digital images based on replacement procedures with use of optimization *Computer Optics* **41(3)** 412-421 DOI: 10.18287/2412-6179-2017-41-3-412-421

[35] Melnikov B and Dudnikov V NFA project (Access mode: https://github.com/va-dudnikov/nfa)

[36] Melnikov B, Korabelshhikova S and Churikova N 2017 On verification algorithms for some binary relations on the general supermonoid of a free monoid *Izvestiya of Higher Educational Institutions. Volga Region. Physics and Mathematical Sciences* **3(43)** 87-99 (in Russian)

[37] Melnikov B and Dudnikov V 2018 Problem of pseudo-optimal placement on the graph and one heuristic approach of its solution *Informatization and Communication* **1** 63-70 (in Russian)

[38] Melnikov B and Zubova T 2018 Mathematical modeling of organization management by value guidelines: algorithms for complex estimation and selection of pseudo-optimal actions *International Journal of Open Information Technologies* **3** 1-8 (in Russian)

[39] Melnikov B and Davydova E 2018 Mathematical modeling of increasing the level of safety in case of failures of space technology *International Journal of Open Information Technologies* **5** 1-6 (in Russian)

[40] Melnikov B and Trenina E 2018 On a problem of the reconstruction of distance matrices between DNA sequences *International Journal of Open Information Technologies* **6** 1-13 (in Russian)

**Acknowledgements**
The reported study was partially supported by the research project of Russian State Social University.