ФІЗИКО-МАТЕМАТИЧНІ НАУКИ

DOI: https://doi.org/10.32839/2304-5809/2021-2-90-21
UDC 519.7

Selyutin Yevhen, Kozin Igor
Zaporizhzhia National University

FEATURES OF METAHEURISTIC METHODS

Summary. The essence of tasks and classification, as well as the necessity of fulfillment of conditions and relations on the set are analyzed. The properties of the equivalence relation are specified. As elementary fragments are analyzed all edges of the graph. The conditions of attachment of an edge are the next: this edge is a ray of an existing star or has no common vertices with already constructed stars of coverage. Emphasis is placed on the lack of optimality of such a solution. The aim of this work is to construct metaheuristics for finding a suboptimal classification defined by a tolerance relation on a finite set. This approach allows one to construct partitions close to optimal sets in accordance with the relation of "proximity" of elements. Moreover, this relationship of proximity is not transitive. The proposed algorithms can find wide application in applied problems related to the problem of object classification by a number of attributes. Such problems often arise in the economic, social and technical sciences. The urgency of the fragmentary structure of the problem was emphasized. Specifies the ability to build classes by looking at the entire list of objects that are classified in a specific order. On the basis of fragmentary structure it is proposed to use evolutionary algorithm. The prospect of using a genetic algorithm to find the best classifications was evaluated. The step-by-step sequence of operations of the genetic algorithm with examples is shown: selection, crossing, mutation, selection. Examples of key operators are given, named crossovers and mutations. A detailed algorithm of the evolutionary model is clearly illustrated. The principle of action of the evolutionary fragmentary algorithm is described in detail. As a set of feasible solutions, a subset of maximal fragments on a given fragmentary structure is considered. The mechanism of checking the quality of the genetic algorithm on a fragmentary structure, which reduces to a lot of variants, is defined. The problem of finding optimal classifications on a finite set is investigated. It is shown that the problem of finding the optimal classification generated by the tolerance relation on a finite set is reduced to the problem of optimization on the set of permutations. A modification of the method of mixed jumping frogs for finding suboptimal solutions of the classification problem is proposed.

Keywords: classification, optimal classification, discrete set, evolutionary algorithm, genetic algorithm, method of mixed jumping frogs.

Селютін Є.К., Козін І.В.
Запорізький національний університет

ОСОБЛИВОСТІ МЕТАЕВРИСТИЧНИХ МЕТОДІВ

Анотація. Проаналізовано особливості оптимальної класифікації на дискретній множині. Зазначено властивості відношення еквівалентності. Проаналізовано всі ребра як елементарні фрагменти графу. Визначено наступні умови кріплення ребра: ребро – це промінь існуючої зірки або воно не має спільних вершин з уже побудованими зірками покриття. Акцент робиться на відсутності оптимальності такого рішення. Розглянуто метаевристіки для пошуку субоптимальності класифікації, визначеної ставленням толерантності на кінцевій множині. Такий підхід дозволяє будувати близькі до оптимальних розбиття множини відповідно до відношення «близькості» елементів. Причому це відношення близькості не є транзитивним. Запропоновані алгоритми можуть знайти широке застосування в прикладних задачах, пов'язаних з проблемою класифікації об'єктів за рядом ознак. Визначено, що подібні завдання часто виникають в економічних, соціальних і технічних науках. Наголошено на актуальності фрагментарної структури. Запропоновано використовувати еволюційний алгоритм на основі фрагментарної структури. Було оцінено перспективу використання генетичного алгоритму для пошуку найкращих класифікацій. Показано поетапну послідовність операцій генетичного алгоритму на прикладах: відбір, схрещування, мутація, відбір. Наведено приклади ключових операторів, названих кросоверів та мутацій. Чітко проілюстровано детальний алгоритм еволюційної моделі. Детально описано принцип дії еволюційного фрагментарного алгоритму. Як набір можливих рішень розглядається підмножина максимальних фрагментів на даній фрагментарній структурі. Визначено механізм перевірки якості генетичного алгоритму на фрагментарній структурі, який зводиться до безлічі варіантів. Досліджено проблему пошуку оптимальних класифікацій на кінцевій множині. Показано, що завдання пошуку оптимальної класифікації, яку породжено ставленням толерантності на кінцевій множині зводиться до задачі оптимізації на множині перестановок. Запропоновано модифікацію методу перемішаних стрибаючих жаб для пошуку субоптимальних рішень задачі класифікації.

Ключові слова: класифікація, оптимальна класифікація, дискретна множина, еволюційний алгоритм, генетичний алгоритм, метод перемішаних стрибаючих жаб.

Introduction. One of the most common mathematical problems of today is classification. It arises in the analysis of research results, in the design and forecasting, in the evaluation and decision-making. In general, it is quite complex and contradictory.

An effective approach to the problem of classification based on the theory of fragmentary structures is found [2]. Like every metaheuristic the fragmentary approach is inferior to the classical approximate algorithms and cannot guarantee the closeness of the solution with its help to the optimal one. Simplicity of implementation and high speed of convergence allows finding with this method good approximate solutions of many complex optimization problems [3-5].

This paper discusses the use of a combination of fragmentary and evolutionary algorithms to find optimal classifications and compares results with other metaheuristic methods.

The basic idea of finding a "good" classification is to introduce an optimality criterion. In this case, the classification problem is reduced to the problem of finding the optimal solution for some set of feasible solutions. Then all kinds of optimization methods start to work.

Analysis of recent research and publications. Nowadays, a large number of different classification methods are known, the effectiveness of which depends substantially on the specifics of the domain in which this task is formulated, and the peculiarities of the source information.

In the works [3–5] examples of theoretical application of genetic algorithms for solving classification problems are considered.

Formal problem statement. The problem of classification on the set X is the problem of splitting the set X into disjoint classes, that is, the task of finding such a family of subsets $\{X_\alpha\}$ of the set X for which the following requirements are fulfilled:

1) $X_u \in X, X_u \neq \varnothing$;

2) $\bigcap_\alpha X_\alpha = X$;

3) $X_\alpha \bigcap X_\beta \neq \varnothing$.

The requirement of class emptiness is irrelevant. We will consider the same classifications, which differ only by empty elements.

Recall that an equivalence relation on the set X called and any binary reflexive, symmetric relation on this set (the ratio "~"), which is defined by the following properties:

1. $\forall x \in X\, x \approx x$;

2. $\forall x, y \in X\, x \approx y \rightarrow y \approx x$;

3. 2. $\forall x, y, z \in X\, x \approx y, y \approx z \rightarrow x \approx z$.

Each classification generates a natural equivalence relation on the set X. Two elements are equivalent, if and only if they belong to the same class.

The article purpose is to analyze the main features of metaheuristic methods.

The main material. An example of the classification problem that will be considered in this paper is the problem of covering a star graph, which occurs in many economic applications.

Suppose a graph whose edges are weighted by nonnegative numbers. The task is to find a set of stars in this graph that contain all vertices of the graph and the total weight of the edges is minimal. This task is in fact a classification task. The class in this case is all the vertices of the star. And in each class a representative – the center of the star.

The stars are arranged in an arbitrary manner. The next step in the algorithm is to select the first star in the sequence of stars that has no vertices in common with the fragments already found.

The idea of a fragmentary algorithm is the following: Every feasible solution to a problem is a fragment, which is represented as the union of indivisible parts – elementary fragments. A plurality of fragments is subjected to a join operation that allows one to obtain a new fragment by adding one of the elementary fragments to an existing one. A fragmentary algorithm is an algorithm for finding the maximum for the inclusion of a fragment.

Lets study it more strictly:

a) Determines the set of fragments {f} and the external operation ⊕ combining the fragments. Each feasible solution to the problem consists of a finite number of fragments combined by an external operation.

b) Set a series of linear order {} relationships on a plurality of fragments, and an efficient ordering of fragments, which allows the ordering of fragments in different order according to the rules chosen.

c) Specify the conditions of attachment of the fragment, which can be both deterministic and dynamic, which change at each step. For an already selected subset of fragments, an effective procedure for checking the possibility of joining a fragment that does not belong to the selected subset is specified.

Thus, the fragmentary algorithm allows us to construct a feasible solution of the problem for a linear number of fragments of time.

We show that the problem of covering a graph with stars has a fragmentary structure and, accordingly, any feasible solution to the problem can be obtained by applying a fragmentary algorithm.

As elementary fragments are all edges of the graph. The conditions of attachment of an edge – this edge is a ray of an existing star or has no common vertices with already constructed stars of coverage. It is easy to show that by applying a fragmentary algorithm to a certain choice of the sequence of edges, any set of stars in a graph can be obtained, that is, any valid solution to the classification problem. However, such a solution is not necessarily optimal. In order to find the optimal solution, we describe a combination of evolutionary and fragmentary algorithms.

Evolutionary (genetic) algorithms have been considered in detail in numerous publications. Genetic algorithms are an optimal method based on the evolution of a population of individuals. In 1975 the book John Holland, "Adaptation in Natural and Artificial Systems," in which a genetic algorithm was proposed.

For a number of optimization problems, we have been able to offer sufficiently effective procedures for finding optimal solutions based on the application of evolutionary algorithms. To implement the evolutionary algorithm, it is necessary to distinguish a number of objects and procedures, the set of which will be called the evolutionary model. The main components of the evolutionary model are the following:

– base set of solutions – set of admissible solutions X on which the optimal solution of the problem is sought;

– operator of initial population construction: a procedure that allows to set on set of all admissible solutions its subset $Y \subseteq X$ for the next evolution;

– selection criterion – an algorithm that allows you to compare the quality of the solution within a given population;

– crossover operator that allows for two valid parent solutions to build a new descendant solution from the set of valid solutions;

– a selection operator that separates multiple pairs in Y to perform the crossover operation;

– operator of evolution, allowing to build new populations from many parents and offspring;

is a stop rule that determines the stop condition of an evolutionary algorithm.

Let us briefly describe the principle of evolutionary algorithm. In the initial step, a set of solutions Y_0 is constructed with the help of the initial population operator. At each next step, a certain set of valid solutions is predicted – the current population. The first step is the set $Y = Y_0$. The value of the selection criterion is calculated for each element of the set *Y*. Further, by using the selection in the current population *Y* is chosen set of pairs for the crossover. A crossover operator is applied to each pair of the selected set of pairs, and then a mutation operator is applied to the crossover result. In this way there are many elements – descendants.

The intermediate population $Y \cup \tilde{Y}$, which is the union of the current population and the set of offspring, is applied by an evolution operator, which allocates a new current population on that set. The process of evolution is repeated until the condition of stopping the evolutionary algorithm is fulfilled.

The properties of fragmentary structures allow us to construct a special class of evolutionary algorithms on fragmentary structures – EVF algorithms.

The EVF algorithm is a combination of evolutionary and fragmentary algorithm. We describe the evolutionary model and principle of operation of such an algorithm.

As a set of feasible solutions, a subset of maximal fragments on a given fragmentary structure is considered. Each fragment from this subset is defined as the result of the fragmentary algorithm operation with some given permutation of elementary fragments. Thus, any admissible solution corresponds to a definite permutation of the numbers *1,2, ..., N* , where *N* is the number of elementary fragments. The value of the objective function is defined for each valid solution.

Basic set of *X* evolutionary model – a set of $S_N = \{i_1, i_2,..., i_N\}$ all permutations of the numbers *1, 2, ..., N* . The operator selects constructing initial population subset given power *Q* from the set *X*.

The rule for calculating the selection criterion is arranged as follows: by the given permutation of fragments using the fragmentary algorithm, the maximum allowed fragment is constructed. The value of the task objective function for this fragment is calculated.

We describe the crossover operator y.

Let be $U = (u_1, u_{2,...,} u_N)$ and $V = (v_1, v_2, ..., v_N)$ two random permutations. The descendant permutation is constructed as follows: the *U* and *V* sequences are viewed from the beginning. At the *k*-th step, the smallest of the first elements of the sequence is selected and added to the new offspring. This element is then removed from the two parent sequences. For example,

$$K((2,3,4,7,8,1,6,5),(3,4,6,2,1,5,8,7)) = (2,3,4,6,1,\ 5,7,8) \quad (1)$$

The mutation operator *M* performs a random transposition (replacement of two elements) in the permutation.

The selection operator randomly selects a set of pairs from a given number of pairs in the multiple permutations of the current population.

The evolution operator arranges elements of the intermediate population into a sequence by decreasing the value of the selection criterion. As the new current population elected the first *Q* elements of the sequence.

Average usually stop – number of generations has reached the limits *L*. The best value for the selection criterion is the permutation from the last constructed population that determines the approximate solution to the problem.

Mixed jumping frogs method. The algorithm of the method of mixed jumping frogs is simple to understand and implement, has a small number of parameters, and has been successfully used to solve combinatorial and continuous optimization problems [5; 6].

The essence of the jumping frog algorithm for finding the optimal permutation is reduced to the following sequence of steps.

Step 1. Initialize the initial frog population as a set of points in the permutation space with Kendall's metric S_n.

Step 2. Calculate the value of the optimality criterion for each permutation from the initial population.

Step 3. Arrange the solutions in descending order of the optimality criterion value.

Step 4. Divide virtual frogs (solutions) into memplex blocks in such a way that the first virtual frog in the sorted list falls into the first memplex, the second is entered into the second memplex, etc.

Step 5. Find the best s_{k1} and worst s_{k2} solution in each memplex $k \in \{1,2,...,K\}$.

Step 6. Try to improve the position of the worst virtual frog by randomly moving it in the direction of the best frog $s = Cross(s_{k2}, s_{k1})$.

Step 7. If the previous operation does not improve the solution, then try to improve the position of the worst virtual frog by moving it towards the globally better frog $s = Cross(s_{k2}, s_{11})$.

Step 8. If the last operation does not improve the position of the virtual frog, then instead of it, randomly create a new frog in the search area – a permutation.

Step 9. Combine virtual frogs of all memplexes into one group.

Step 10 . If the conditions for the completion of the algorithm are not met, then go to Step 3.

Step 11. The last globally best virtual frog corresponds to a suboptimal problem solution.

Let us now describe this algorithm formally, taking into account the parameters.

The method parameters are as follows:

1) the number of classes of frogs Q $(Q \geq 2)$;

2) the number of elements *r* in each class (it is assumed that the sizes of the classes are the same and $r \geq 2$);

3) the maximum number of steps *K of the* algorithm;

4) the number *D of the* best frogs in the class, and $0 < D < r$.

In accordance with the specified parameters, the size *N* of the frog population (the set of feasible solutions) is determined by the formula $N = Qr$. In the initial step of the algorithm creates the initial population of frogs by generating random permutations $s^j = (i_{j1}, i_{j2}, .., i_{jn}),\ j = 1,2,...,N$.

Table 1

Results of applying different approaches

Series (number of vertices)	Number of tasks	Algorithm search loc.		Random Search		Evolutionary algorithm		Jumping frog method	
		Record	Rating	Record	Rating	Record	Rating	Record	Rating
A 50	100	44	286	88	468	100	500	100	500
B 100	100	ten	221	0	348	100	500	100	500
H 500	100	0	212	0	280	94	394	100	500
D 1000	100	0	118	0	201	92	392	100	500

The best permutation of the vertices in terms of the goal function is selected, which defines the permutation $s^* = (i_1, i_2, ..., i_n)$, and the value of the objective function is calculated $F(x^*)$ on this permutation:

Step k $(1 \le k \le K)$. The set is ordered $P^{(k-1)}$ by the value of the objective function, that is $F(s^k) \ge F(s^{k-1})$, $k = 2,3,..., N-1$. The population $P^{(k-1)}$ is divided into Q classes of the same cardinality r

$$P_q^{(k-1)} = \{s^{(qi)} \mid s^{(qi)} = x^j,\ j = q + (i-1)Q,\ i = 1,2,..,r,\ q = 1,2,...,q\}.$$

The best solution $s^* = s^1$ is determined by the value of the objective function for the entire population. In each class $P_q^{(k-1)}$, the "best" $s^{(q1)}$ and "worst" $s^{(qr)}$ are determined by the value of the objective function of the solution. In each class $P_q^{(k-1)}$, the positions (sequences of traversing the vertices of the graph) of frogs change with numbers from $D+1$ to r. For each value of the index $i \in \{D+1,2,...,r\}$ a new position of the i-th frog (the sequence of traversing the vertices) in the class with number q is determined according to the following rule: a random permutation s^c is calculated from the interval between the permutations $s^{(q1)}$ and $s^{(qr)}$ in the Kendall metric.

The permutation on the segment between $s^{(q1)}$ and $s^{(qr)}$ is built according to the rule: sequences $s^{(q1)}$ and $s^{(qr)}$ are viewed from left to right. At the next step, the smallest of the first elements of the sequences is selected and added to the new permutation. Then this element is removed from the permutations $s^{(q1)}$ and $s^{(qr)}$. For example, applying this operation to the permutations (2,4,7,6,1,3,5,8) and (5,8,1,3,4,2,6,7) gives the permutation (2,4, 5,7,6,1,3,8).

If $F(s^c) < F(s^{(qi)})$, then we assume $s^{(qi)} = s^c$. If $F(s^c) \ge F(s^{(qi)})$, then a random permutation s^c is chosen in the segment between $s^{(qr)}$ and s^*. If $F(s^c) < F(s^{(qi)})$ then we assume $s^{(qi)} = s^c$. Otherwise, we choose a randomly generated permutation $s^{(qi)}$.

We assume $P^{(k)} = \bigcup_{q=1}^{Q} P_q^{(k-1)}$ and go to the next step of the algorithm.

The algorithm ends when the specified number of steps has been completed. The current permutation s^* determined at the last step is taken as the optimal solution to the problem.

Note that description is, the above algorithm s Resch was the problem of finding the optimal permutations of n elements in the set of all permutations with the objective function $F(s)$ which is defined on the set of permutations. In this case, the specific type of the objective function does not matter. Therefore, the above algorithm can be used to find suboptimal solutions to optimization problems on a set of permutations with arbitrary objective functions.

Numerical experiment. The weights of the graph edges were chosen randomly in the range [1,100]. These weights were considered as a measure of proximity for the respective vertices. The set of vertices of the graph was considered as a set of elements to be classified. A positive number was chosen randomly in the interval [0,1]. Linear orders (permutations) on the set of vertices were not adjusted using the Fisher-Yates shuffle algorithm.

The problems were solved using a local search algorithm, a random search method, evolutionary algorithms, and the method of mixed jumping frogs.

The comparison of algorithms was carried out in the following directions:

Record is the number of problems in a series where the algorithm turned out to be the best among the tested. Bord rating is the sum of the number of points scored on each problem in the series. For the first place in comparison, 5 points were assigned, for the second 4, for the third 3.

The results of the algorithm comparison are presented on Table 1.

Conclusions. In this article, a method for finding optimal ß-classifications based on two well-known metaheuristics was considered. A numerical experiment showed good results of the proposed algorithms in comparison with local and random search. This approach can be transferred practically without changes to other types of classifications, which are based on the concept of proximity of elements.

References:

1. Skobtsov Y. (2008) Fundamentals of evolutionary computing: textbook. Donetsk: DonNTU. (in Russian)
2. Kozin I. V. (2014) Evolutionary-fragmentary model of the pentomino packing problem. Diskr. Analiz i Issled. Operatsii, Vol. 21, No. 6, 35–50.
3. Lin S. & Kernighan B. W. (1973) An Effective Heuristic Algorithm for the Traveling-Salesman Problem. *Operations Research*. Vol. 21, No. 2. doi: https://doi.org/10.1287/opre.21.2.498
4. Eusuff M.M. (2003) Optimization of water distribution network design using the shuffled frog leaping algorithm. J. Water Resour. Planning Mgmt. Vol. 129. P. 210–225. doi: https://doi.org/10.1061/(ASCE)0733-9496(2003)129:3(210)
5. Narimani M.R. (2011) A New Modified Shuffle Frog Leaping Algorithm for NonSmooth Economic Dispath. World Applied Sciences Journal. P. 803–814.
6. Khumawala B.M. (1972) An Efficient Branch-Bound Algorithm for the Warehouse Location Problem. *Management Science*. Vol. 18. P. 718–731. doi: https://doi.org/10.1287/mnsc.18.12.B718
7. Krarup J. & Pruzan P.M. (1983) The simple plant location problem: Survey and synthesis. *European Journal of Operational Research*. Vol. 12. P. 36–81. doi: https://doi.org/10.1016/0377-2217(83)90181-9