FACTA UNIVERSITATIS (NIŠ)
Ser. Math. Inform. Vol. 35, No 4 (2020), 1091–1105
https://doi.org/10.22190/FUMI2004091L

COMPARATIVE STUDY OF MUTATION OPERATORS IN THE GENETIC ALGORITHMS FOR THE K-MEANS PROBLEM *

Riu Li and Lev A. Kazakovtsev

Abstract. The k-means problem and the algorithm of the same name are the most commonly used clustering model and algorithm. Being a local search optimization method, the k-means algorithm falls to a local minimum of the objective function (sum of squared errors) and depends on the initial solution which is given or selected randomly. This disadvantage of the algorithm can be avoided by combining this algorithm with more sophisticated methods such as the Variable Neighborhood Search, agglomerative or dissociative heuristic approaches, the genetic algorithms, etc. Aiming at the shortcomings of the k-means algorithm and combining the advantages of the k-means algorithm and revolutionary approach, a genetic clustering algorithm with the cross-mutation operator was designed. The efficiency of the genetic algorithms with the tournament selection, one-point crossover and various mutation operators (without any mutation operator, with the uniform mutation, DBM mutation and new cross-mutation) are compared on the data sets up to 2 millions of data vectors. We used data from the UCI repository and special data set collected during the testing of the highly reliable semiconductor components. In this paper, we do not discuss the comparative efficiency of the genetic algorithms for the k-means problem in comparison with the other (non-genetic) algorithms as well as the comparative adequacy of the k-means clustering model. Here, we focus on the influence of various mutation operators on the efficiency of the genetic algorithms only..

Keywords: k-means problem; Variable Neighborhood Search; genetic algorithms; cross-mutation operator.

1. Introduction

With the increasing popularity of 5G commercialization and the development of the IT hardware, data has grown exponentially in recent years. According to

Received July 21, 2020; accepted September 28, 2020
2020 *Mathematics Subject Classification.* Primary 90C57; Secondary 90C27, 90C09
*This work was supported by the Ministry of Science and Higher Education of the Russian Federation (State Contract No. FEFE-2020-0013)

statistics, the global data usage has reached 40 Zb [1], and researchers in various fields are increasingly interested in big data research. One of the most important directions of intelligent data processing is cluster analysis. Clustering algorithm is a technique for statistical data analysis and is widely used in many fields, including machine learning, data mining, image analysis, and biological information processing. In the commercial field, it can be used in a recommendation system to improve efficiency of the company, and it can also solve problems such as reducing the size of the initial data set and pattern recognition [2, 3]. Cluster analysis, also known as group analysis or automated grouping, is a statistical analysis method performed on a set of several patterns (data set), usually a pattern which is a metric vector, or a point in a multidimensional space. In this paper, we call them data vectors. The criterion to estimate the result of most clustering algorithms is the distance between the elements in the same cluster and the distance between different clusters [4]. According to the principle of clustering objects, similar objects are grouped into a subset so that objects in the same subset are as close as possible, and the distance between different subsets is as far as possible.

The k-means algorithm is one of the most popular clustering algorithms due to its simplicity and remarkable effect [5]. However, the k-means algorithm is a local search method which depends on the initial solution that is given or generated randomly. Genetic algorithms have global optimization capabilities. The Genetic k-Means algorithm is designed by combining the advantages of both. This article summarizes the current research status of genetic clustering algorithms based on the k-means optimization model.

Although the idea of the k-means clustering goes back to Steinhaus in 1957 [6], the term "k-means" was first used by James MacQueen in 1967 [7]. The k-means algorithm is one of the most widely used clustering algorithms due to its simple and convenient implementation principle and good experimental results. This algorithm accepts the parameter k, randomly selectsk cluster centers (called centroids) among the data vectors (if they are not given), and calculates the distance between the N data in the data set and the closest center points. This experiment uses Euclidean distance in a d-dimensional space [8]:

$$d(X,Y)=\sqrt{\sum_{i=1}^{d}(x_i-y_i)^2}.$$

Here, $X=(x_1,...,x_d)$ and $Y=(y_1,...,y_d)$ are two given points (vectors).

The minimized objective function (sum of squared distances also called sum of squared errors, SSE) of the k-means optimization model and algorithm is as follows:

$$f(C_1,...,C_k)=\sum_{i=1}^{N} min_{j=\overline{1,k}} d^2(A_i,C_j). \tag{1.1}$$

Here, $A_1,...,A_N$ are the clustered data vectors, and $C_1,...,C_k$ are the searched cluster centers (centroids) which must be found.

According to the obtained results, each data sample is assigned to its nearest cluster, and the cluster center with the newest average value of the data samples in each cluster is calculated. The cluster center is updated repeatedly until the convergence condition is reached. The running process of the k-means algorithm is as follows:

Step 1: Select k objects from the data object as the initial cluster center (this step is optional, only is the initial solution is not given);

Step 2: Calculate the distance of each object to each cluster center separately, and assign the object to the nearest cluster;

Step 3: Recalculate the center of k clusters after all object assignments are completed;

Step 4: Compare with the k cluster centers obtained in the previous calculation. If the cluster center changes, then turn to Step 2, otherwise output the clustering results.

The advantages and disadvantages of the k-means algorithm are obvious. The advantage is that the algorithm is simple and the convergence speed is fast. The results (local optima) obtained with different initial cluster centroid positions may vary significantly. It is easy to get a local optimal solution instead of a global optimal solution.

In order to solve the shortcomings of the k-means algorithm, the k-means++ algorithm proposed by Arthur in 2007 improved the initialization step of the k-means algorithm [9]. This improvement can be intuitively understood so that the k initial cluster centers should be separated from each other as much as possible. However, the k-means++ algorithm and similar "smart initialization" algorithms [10, 11] are still random search methods which fall into a local minimum.

The Genetic Algorithm (GA) was first proposed by Holland of the United States in the 1970s [12]. The algorithm was designed and proposed according to the evolutionary laws of organisms in nature. In 1967, Bagley, a student of Professor Holland [13], first proposed the term "Genetic Algorithm" in his doctoral dissertation and discussed the application of the GAs in games, but early research lacked guiding theory and the development of computing tools. In 1975, Holland et al. [14] proposed a model theory that is extremely important for the study of genetic algorithm theory.

The genetic algorithm has several basic frameworks of coding, fitness function, and initial group selection [15, 16, 17]. Many genetic algorithms for the k-means problem [18, 19, 3] use the direct coding: the chromosome (a solution in a population of solutions) is the set of the coordinates of the cluster centers (centroids).

The fitness function is used to express the adaptability of an individual to the environment. In this research, we use directly (1.1) as the fitness function.

The basic operation process of genetic algorithm is as follows [20]:

Step 1 (Initialization): set the evolution algebra counter $t = 0$, set the maximum evolution algebra T, and randomly generate n individuals as the initial group $P(0)$.

Step 2 (Individual evaluation): Calculate the fitness of each individual in the group $P(t)$.

Step 3 (Selection operation): Apply the selection operator to the group. The purpose of selection is to directly inherit the optimized individuals to the next generation or to generate new individuals through pairing and crossover to the next generation. The selection operation is based on the assessment of the fitness of the individuals in the group.

Step 4 (Crossover operation): Apply crossover operator to the group. The crossover operator plays a central role in genetic algorithms.

Step 5 (Mutation operation): Apply mutation operators to groups. That is, the gene values at certain loci of individual strings in the group are changed. After selection, crossover and mutation operations, the population $P(t)$ obtains the next generation population $P(t+1)$.

Step 7 (Termination condition judgement): if $t = T$, the individual with the maximum fitness obtained in the evolution process is used as the optimal solution output to terminate the calculation. Instead of the limitaion of generations, the time limitation can be used.

Since the k-means is an NP-hard optimization problem [21, 22, 23], the results are easily stuck by the local optimal solution. The genetic algorithms are popular instruments for global optimization. Krishna and Morty proposed a new clustering method called Genetic K-means Algorithm (GKA) [17] combining the global search capabilities of genetic algorithms with traditional k-means algorithms.

The flow of GA-k-means algorithm [17, 24, 25, 26, 27] is as follows.

Step 1: K samples are randomly selected from data set as the cluster center, and the k cluster centers are considered as a chromosome. This operation is repeated n times to obtain a population of size n.

Step 2: Use ordinary k-means algorithm to cluster data set with each chromosome as the cluster center. Get the new clustering center and the fitness function value corresponding to each chromosome.

Step 3: Get the next generation through selection, crossover, and mutation operations, and retain the best chromosomes from the previous generation. Repeat Steps 2 and 3 until the termination condition is met.

Each chromosome is a sequence of real numbers representing k cluster centers. For a d-dimensional data set, the length of the chromosome is kd. The sum of the squared distances within the clusters in the data set (1.1) is used as the fitness function.

Such algorithms can use two main selection operations. The first one is the most commonly used method of proportional (roulette wheel) selection. The main idea is that the probability that an individual is selected depends directly on the corresponding fitness of the individual. Another one is tournament selection strategy.

For the k-means problem, after several iterations, the fitness function values of all individuals become very close to each other. Thus, the roulette wheel selection is inefficient, and we use the tournament approach. The algorithm randomly selects 3 chromosomes from the population, and then selects an individual with the highest fitness value from these 3 chromosomes [28]. Since we need to select two "parent" chromosomes for the crossover operator, the second one can be chosen using the same approach or selected randomly from the population with equal probabilities.

The crossover is a random process. The first type is a single-point crossover [24]. A point is randomly selected as the crossover point in the range of 1 to chromosome length, and the two chromosomes are exchanged to the right of the cross point. Two different points are randomly selected as the intersection point in the range of 1 to chromosome length, and the middle part of the two points is exchanged to obtain two new offspring. One of them is randomly selected to survive and enter next generation. The third type is uniform crossover. For each node on the chromosome, there is a certain chance that the crossover operation will occur. After the entire process is completed, two new chromosomes will be obtained, and one will be randomly selected to survive. However, in this paper we use the one-point crossover only and focus on the efficiency of various mutation operators.

In the genetic algorithm, the mutation operation is to imitate the mutation link of biological evolution in nature to change the individual. Although the chance of mutation is relatively small, it is an indispensable link to generate new species. Constantly fine-tune the new individuals generated by the crossover operator to increase species diversity and search area. Traditionally, such genetics algorithms algorithms with real-coded chromosomes for the k-means problem do not use any mutation operators [18, 3] mutation operations commonly used in other GAs. However, this reduces the population diversity and may make the final results premature and converge prematurely [25].

We compare several mutation operators and propose a simple idea of using the one-point crossover operator as the mutation operator. The efficiency of such approach is proved experimentally.

Our comparison is performed only with respect to squared distance between points and centroids (1.1). There are lots of other clustering quality measures and lots of clustering models (minimized or maximized objective functions), and there is a perpetual question, which clustering model is more adequate. In this paper, we do not compare the adequacy of the clustering models. We do not use any internal or external criteria [29, 30, 31, 32, 33] which allow us to compare the adequacy (preciseness) of various clustering models. The only aim of this research was to improve the solution of the k-means optimization problem (not the accuracy of the clustering result), i.e., to build algorithms which allow us to obtain better values of the sum of squared distances.

2. Mutation in the Genetic k-Means Algorithm

In comparison with the crossover operator, the mutation operator in standard genetic algorithms is usually considered as a secondary operator with low probability [34]. Nevertheless, some evolutionary algorithms without any crossover operator are able to work better than standard genetic algorithms due to mutation and selection [35, 36, 37, 38].

The majority of the bibliographical sources describe the evolutionary algorithms for the k-means problems which use integer or binary chromosome encoding [39, 40, 30, 41, 42, 43], and thus these approaches cannot be implemented in our study because the considered greedy heuristic algorithms use the real encoding only. The other part of the sources propose the algorithms which actually solve a problem other than k-means (other than sum of squared distances minimization) while we focused on the improvement of the k-means problem solution only without any change in the clustering model. The third part of the sources including authors' papers do not use any mutation operators at all or use special operators called mutation which actually run local search algorithms [18, 3, 44, 19, 45].

The mutation operator [46] changes each allele (a part of the chromosome) a_n ($n = 1, ..., k$) to a new value a'_n (a'_n might be equal to a_n) with probability M_P independently, where $0 < M_P < 1$ is a parameter called the mutation probability that is specified by the user. weak mutation, average mutation, and strong mutation. Usually, the probabilities are $1/5n$ (weak mutation) , $1/n$ (average mutation), and $5/n$ (strong mutation), and n is the length of the individual (the number of alleles in the chromosome). There are two purposes for introducing mutations into genetic algorithms: one is to make the genetic algorithm have local random search ability. When the genetic algorithm is close to the optimal solution neighborhood through the crossover operator, the local random search ability using the mutation operator can accelerate the convergence to the optimal solution. Obviously, the probability of mutation in this case should be a small value, otherwise the building blocks close to the optimal solution will be destroyed by the mutation. The second is to enable genetic algorithms to maintain group diversity to prevent immature convergence. The termination condition of the genetic algorithm is that the individual's fitness reaches a preset threshold, or the fitness does not rise any more, or the number of iterations reaches the preset algebra.

In [26], Sheng proposed a simple inversion approach. Generate randomly a number within 0-1. If this number is less than the probability of mutation, then perform an inversion operation on a value in the chromosome. This method has certain limitations and can only be used for populations whose chromosomes are binary-coded, but not for populations whose chromosomes are real-coded. For the real-valued chromosomes, only few approaches were proposed.

Maulik proposed the Uniform random mutation in [27]. Randomly generate a number from 0 to 1, if the number is less than mutation probability, the point on the chromosome will mutate. The mutation strategy is as follow. Randomly generate the number b from 0 to 1, if the value at a gene (cenreoid coordinate) position is v, after mutation it becomes:

$$v \leftarrow \begin{cases} v \pm 2bv, & v \neq 0, \\ \pm 2b, & v = 0 \end{cases} .$$

Positive and negative signs have the same probability. This mutation operation is simple, however, the disadvantage is that when the value of a certain data is very large or very small, the impact of this mutation will also be very large or very small, which does not conform to the principle of mutation. If the range of the initial data is very large, the gap between the mutated data and the initial data will be very large.

In 1999, Krishna and Murty proposed a mutation strategy called distance-based mutation (DBM) [28, 46]. Authors believed that the mutation must change the allele value according to the distance of the cluster centroid from the corresponding data point. Each allele (part of the chromosome) corresponds to a data point, and its value represents the cluster to which the data point belongs. Define operators so that if the corresponding cluster center is closer to the data point, the allele value is more likely to be changed to the cluster number. After determining that an allele is about to mutate, replace the allele with a randomly selected value from the following distribution:

$$P_j = \frac{c_m d_{max} - d_j}{\sum_{(i=1)}^{K} (c_m d_m ax - d_i)}.$$

Here, $d_j = d(A_i, C_j)$ is the Euclidean distance between point x_i and centroid c_j, and c_m is a constant.

3. Idea of the Cross-Mutation Operator

The idea of our new mutation operator is very simple: to implement the crossover operator to the individual being mutated and to a randomly generated individual.

We call this new mutation operator the cross-mutation. A randomly generated result improved by the standard k-means algorithm is used as an input of the mutation operator. The solution being mutated is the second input. For this two input solutions (chromosomes), we implement the single-point crossover and then run the k-means algorithm again to improve the result. Similar ideas are used in known variable neighborhood search algorithms [47]. Observe the performance of the genetic clustering algorithm using cross-mutation-like operators by comparing the cross-mutation-like operators with the other three mutation operators.

The cross-mutation operator can be described as follows:

Required: Chromosome to be mutated S.

Step 1: Randomly generate a chromosome (set of centroids) S';

Step 2: $S' \leftarrow kmeans(S')$;

Step 3: $S \leftarrow crossover(S, S')$;

Step 4: $S = kmeans(S)$.

Our computational experiments show that the genetic algorithms based on this idea are able to outperform both the algorithms without any mutation and the algorithms with the uniform random mutation and DBM algorithms.

4. Computational Experiments

In our experiments, we used data sets from the UCI repository [48] and data collected during the process of testing the highly reliable electronic components (semiconductor devices 140UD25) [49] in a specialized testing center [50]. The aim of clustering the highly reliable semicinductor devices is to detect the homogeneous production batches in a mixed lot of the shipped devices.

The semiconductor device data set contains 1125 objects of dimensionality 18 (18 tests), and each dimension represents a certain attribute of the tested device.

Five clustering algorithms are used: k-means algorithm in the multi-start mode, Genetic k-Means algorithm clustering algorithm without any mutation operator, Genetic k-means algorithm with the uniform random mutation operator, Genetic k-means algorithm with cross-mutation operator, and the DBM genetic clustering algorithm.

For distance measure, Euclidean distance. For all data sets, we used the 0-1 normalization. All algorithms ran 30 times limited by 150 generations. Population size is equal to 20.

All the experiments were performed with the average mutation probability $1/n$ where n is the length of the chromosome ($n = k$ for the Genetic k-Means algorithm).

As the result of a randomized algorithm may be accidental. In order to make the experimental results statistically significant, run the entire experiment 30 times and record the experimental results. The averaged results for the semiconductor device data set are summarized in Tables 4.1 and 4.2.

Table 4.1: Computational experiments with semiconductor testing data set (1125 data vectors of dimensionality 18), 150 generations, 30 attempts

Mutation	Obj. function	(1.1) value
strategy	Average	Median
Without mutation	92.219	92.255
Uniform random mutation	91.862	91.905
Crossover-like mutation	91.638	91.635
DBM mutation	91.909	91.815

Table 4.2: Statistical significance of the difference in results (Mann–Whitney U test) for the semiconductor testing data set, 30 attempts

Mutation strategies	Significance level	Conclusion
Without mutation vs. uniform	0.012	Significant difference
Without mutation vs. cross-mutation	0.005	Significant difference
Uniform mutation vs. cross-mutation	0.011	Significant difference
DBM vs. uniform	0.110	Difference is insignificant
DBM vs. cross-mutation	0.008	Significant difference

Fig. 4.1 shows that the convergence speed of two algorithms is almost the same. However, the median convergence speed of 30 runs is better for the GA with our new mutation operator, and this difference is statistically significant.

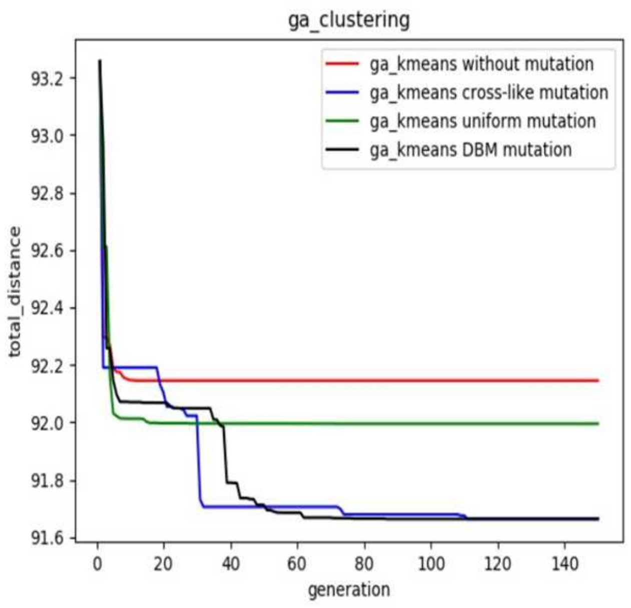

Fig. 4.1: Comparative convergence speed of four genetic algorithms with various mutation operators on the semiconductor testing data set

The advantage of the new mutation operator over the three other variants is statistically significant.

Clustering algorithms can be used in recommendation systems, based on user portraits, to identify products or videos that may be of interest to users. For example, in the e-commerce industry and short video industry that have emerged in recent years, by using real-time recommendation systems using big data tech-

Table 4.3: Computational experiments with household power consumption data set (2075259 data vectors of dimensionality 6), 150 generations, 30 attempts

Mutation strategy	Obj. function (1.1) value Average	Median
Without mutation	13468.54	13470.05
Uniform random mutation	13485.99	13487.90
Crossover-like mutation	13457.07	13457.70
DBM mutation	14674.96	13468.35

nology, by analyzing user behavior, making user portraits and clustering users to recommend more products to users, this method brought a lot of revenue to many companies [51, 52].

The second experiment uses data on the household power consumption. The power consumprion information may be a simplest but important indicator of the behaviour of people. The second data set contains data of electric power consumption in the households with a one-minute sampling rate over a period of almost 4 years [48]. Different electrical quantities and some sub-metering values are available. This archive contains 2075259 measurements gathered in a house located in Sceaux (7km of Paris, France) between December 2006 and November 2010 (47 months). Each data contains 8 attributes, namely data, time, global active power, global reactive power, voltage, submeterings. Date and time attributes were removed, and the other attributes were 0-1 normalized.

The results of running our algorithms are shown in Fig. 4.2 and Tables 4.3, 4.4. As it can be seen from the above figure, the genetic clustering algorithm without mutation operator has the fastest convergence speed, and has converged in about 10 generations, and the result stays at 13471.5. For the DBM mutation, it started to decline particularly fast. From the 5th generation to the 25th generation, the downward trend began to become slow. It converged around the 65th generation, and the result stayed at 13469.3. The uniform mutation operator declined very rapidly before the 15th generation, and the downward trend slowed down from the 15th generation to the 35th generation, it converged at the 75th generation. The cross-like mutation operator also had a process of hormonal decline before the 5th generation, and it has been steadily decreasing after the 5th generation until it converges at the 110th generation, and the result is better than the other three operators. Repeat this procedure for 15 times, and record the final results of various mutation operators each time and record them.

For this comparatively large data set, the overall conclusion is the same: our new mutation operator outperforms the other three versions of the genetic algorithms.

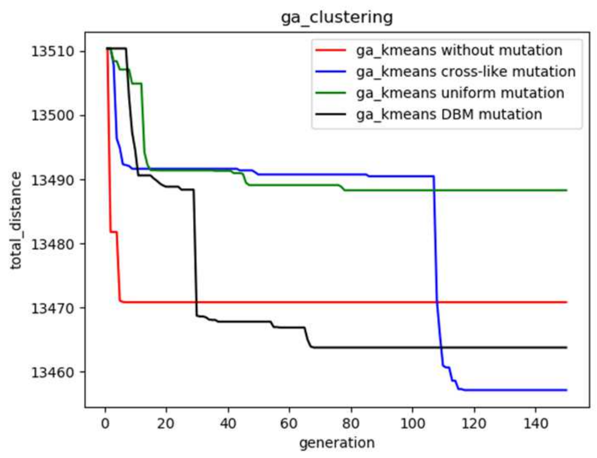

FIG. 4.2: Comparative convergence speed of four genetic algorithms with various mutation operators on household power consumption data set [48]

Table 4.4: Statistical significance of the difference in results (Mann–Whitney U test) for the household power consumption data set, 30 attempts

Mutation strategies	Significance level	Conclusion
Without mutation vs. uniform	0.013	Significant difference
Without mutation vs. cross-mutation	0.005	Significant difference
Uniform mutation vs. cross-mutation	0.011	Significant difference
DBM vs. uniform	0.010	Significant difference
DBM vs. cross-mutation	0.008	Significant difference

5. Conclusions

The modern scientific literature offers only few approaches to building the mutation operator for the genetic algorithms with real coded chromosomes for solving the k-means problem. Traditionally, these algorithms do not use any mutation. However, the simple idea of using the same single-point crossover operator for both crossover and mutation is able to improve the results of the genetic algorithm. In this case, the one-point crossover is applied to the chromosome being mutated and a randomly generated chromosome improved by running the k-means algorithm. This new mutation operator is efficient for both small and large data sets.

However, investigation of the new operator efficiency with various mutation probabilities and various quantity of clusters as well as its applicability for the other crossover operators are subject of our further research.

6. Acknowledgements

Results were obtained in the framework of the state task No. FEFE-2020-0013 of the Ministry of Science and Higher Education of the Russian Federation.

REFERENCES

1. Z.-W. Xu: *Cloud-Sea Computing Systems: Towards Thousand-Fold Improvement in Performance per Watt for the Coming Zettabyte Era.* Journal of Computer Science and Technology, **29 (2)** (2014), 177-181, DOI:10.1007/s11390-014-1420-2.
2. S. Vempala and G. Wang: *A spectral algorithm for learning mixtures of distributions.* FOCS. (2002), 841-860.
3. L. Kazakovtsev and A. Antamoshkin: *Genetic Algorithm with Fast Greedy Heuristic for Clustering and Location Problems.* Informatica, **38 (3)** (2014), 229-240.
4. A. S. Shirkhorshidi, S. Aghabozorgi and T. Y. Wah: *A Comparison Study on Similarity and Dissimilarity Measures in Clustering Continuous Data.* PLoS ONE **10 (12)** (2015), e0144059, DOI:10.1371/journal.pone.0144059.
5. P. Olukanmi, F. Nelwamondo and T. Marwala: *Rethinking k-means clustering in the age of massive datasets: a constant-time approach.* Neural Comput & Applic. (2019), DOI:10.1007/s00521-019-04673-0
6. H. Steinhaus: *Sur la division des corps materiels en parties.* Bull. Acad. Polon. Sci. Cl. III. **IV** (1956), 801-804.
7. J. B. MacQueen: *Some Methods of Classification and Analysis of Multivariate Observations.* Proceedings of the 5th Berkley Symposium on Mathematical Statistics and Probability, **1** (1967), 281–297.
8. H. Norman: *SPSS Statistical Package for the Social Sciences.* Encyclopedia of Information Systems **13(1)** (2003), 187-196.

9. D. Arthur and S. Vassilvitskii: *K-Means++: The Advantages of Careful Seeding.* Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, (SODA 2007), New Orleans, Louisiana, USA (2007), DOI: 10.1145/1283383.1283494.

10. B. B. Bhusare and S. M. Bansode: *Centroids Initialization for K-Means Clustering using Improved Pillar Algorithm.* International Journal of Advanced Research in Computer Engineering & Technology (IJARCET). **3 Issue 4** (2014).

11. S. Mahmud, M. Rahman and N. Akhtar: *Improvement of K-means clustering algorithm with better initial centroids based on weighted average.* 7th International Conference on Electrical and Computer Engineering. IEEE. (2012), ISBN 9781467314367. DOI:10.1109/icece.2012.6471633.

12. M. Mitchell, J. H. Holland *and* S. Forrest: *When Will a Genetic Algorithm Outperform Hill Climbing.* Advances in neural information processing systems (1994), 51-58.

13. J. D. Bagley: *The behavior of adaptive systems which employ genetic and correlation algorithms: technical report.* University of Michigan, 1967.

14. J. Holland: *Genetic Algorithms, computer programs that evolve in ways that even their creators do not fully understand.* Scientific American **267 (1)** (1992), 66-72.

15. D. E. Goldberg: *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley, New York, 1989.

16. A. Konak, D. W. Coit and A. E. Smith: *Multi-objective optimization using genetic algorithms: a tutorial.* Reliability Engineering & System Safety **91(9)** (2006), 992-1007, DOI:10.1016/j.ress.2005.11.018.

17. J. J. Grefenstette: *Optimization of control parameters for genetic algorithms.* IEEE Transactions on Systems Man and Cybernetics **16 (1)** (1986), 122-128.

18. O. Alp, E. Erkut and Z. Drezner: *An Efficient Genetic Algorithm for the p-Median Problem.* Annals of Operations Research **122** (2003), 21-42, DOI:10.1023/A:1026130003508.

19. M. N. Neema, K. M. Maniruzzaman and A. Ohgai: *New Genetic Algorithms Based Approaches to Continuous p-Median Problem.* Netw. Spat. Econ. **11**, (2011), 83-99, DOI:10.1007/s11067-008-9084-5.

20. M. Srinivas, L. M. Patnaik: *Genetic algorithms: a survey.* Computer **27 (6)** (1994), p.17-26, DOI: 10.1109/2.294849.

21. M. Garey, D. Johnson and H. Witsenhausen: *The complexity of the generalized Lloyd - Max problem.* IEEE Transactions on Information Theory **28 (2)** (1982), 255-256. DOI:10.1109/TIT.1982.1056488.

22. D. Aloise, A. Deshpande, P. Hansen and P. Popat: *NP-hardness of Euclidean sum-of-squares clustering.* Machine Learning **75 (2)** (2009) 245-249, DOI:10.1007/s10994-009-5103-0.

23. S. Dasgupta and Y. Freund: *Random Projection Trees for Vector Quantization.* IEEE Transactions on Information Theory **55 (7)** (2009), 3229–3242, DOI:10.1109/TIT.2009.2021326.

24. D. Goldberg: *Genetic Algorithms in Search, Optimization, And Machine Learning.* Addison-Wesley, New York, (1989).

25. P. Chi: *Genetic Search with Proportion Estimation* Proceedings of the Third Int. Con. on Genetic Algorithms (ICGA), San Mateo, California (1989), 92-97.

26. Y. Hu, J. Bi: *K-means clustering algorithm based on genetic optimization.* Journal of Computer System Applications **19(6)** (2010), 52-55.

27. J. A. Hartigan and M. A. Wong: *Algorithm AS 136:A K-means clustering algorithm.* Appl. Stat. **28(1)** (2013), 100-108.

28. K. Krishna and M. Murty: *Genetic K-means algorithm.* IEEE Transactions on Systems, Man and Cybernetics - Part B: Cybernetics **29(3)** (1999), 433-439.

29. P. Rousseeuw: *Silhouettes: a graphical aid to the interpretation and validation of cluster analysis.* Journal of Computational and Applied Mathematics **20** (1987), 53-65.

30. B. Auffarth: *Clustering by a genetic algorithm with biased mutation operator.* IEEE Congress on Evolutionary Computation, Barcelona (2010), 1-8, DOI: 10.1109/CEC.2010.5586090.

31. G. Schwarz: *Estimating the Dimension of a Model.* Annals of Statistics **6 (2)** (1978), 461-464, DOI:10.1214/aos/1176344136.

32. R. Tibshirani, G. Walther and T. Hastie: *Estimating the number of clusters in a data set via the gap statistic.* Journal of the Royal Statistical Society **36** (2001), 411-423.

33. W. M. Rand: *Objective Criteria for the Evaluation of Clustering Methods.* Journal of the American Statistical Association **66(336)** (1971), 846-850, DOI:10.1080/01621459.1971.10482356.

34. J. H. Holland: *Adaptation in natural and artificial systems.* MIT Press, Cambridge (1992).

35. D. B. Fogel and J. Atmar: *Comparing genetic operators with gaussian mutations in simulated evolutionary processes using linear systems.* Biol. Cybern. **63** (1990), 111-114.

36. C. Liu and A. Kroll: *On designing genetic algorithms for solving small- and medium-scale traveling salesman problems.* LNCS **7269** (2012), 283-291.

37. E. Osaba, R. Carballedo, F. Diaz, E. Onieva, I. de la Iglesia and A. Perallos: *Crossover versus mutation: a comparative analysis of the evolutionary strategy of genetic algorithms applied to combinatorial optimization problems.* Sci. World. J. (2014), DOI:10.1155/2014/154676

38. J. Walkenhorst, T. Bertram: (2011) *Multikriterielleoptimierungsverfahren für pickup-and-delivery-probleme.* Proceedings of 21. Workshop computational intelligence, Dortmund, Germany (2011), 61–76.

39. S. S. Cheng, Y. H. Chao, H. M. Wang and H. C. Fu: *A Prototypes-Embedded Genetic K-means Algorithm.* 18th International Conference on Pattern Recognition (ICPR'06), Hong Kong (2006), 724-727, DOI:10.1109/ICPR.2006.155.

40. E. S. Correa, M. T. A. Steiner, A. A. Freitas and C. Carnieri: *A Genetic Algorithm for the P-median Problem.* Proc. 2001 Genetic and Evolutionary Computation Conference (GECCO-2001), San Francisco (2001), 1268-1275.

41. Y. Alkhalifah and R. L. Wainwright: *A genetic algorithm applied to graph problems involving subsets of vertices.* Proceedings of the 2004 Congress on Evolutionary Computation (IEEE Cat. No.04TH8753), Portland, OR, USA **1** (2004), 303-308, DOI:10.1109/CEC.2004.133087.

42. R. Dash and R. Dash: *Comparative Analysis of K-means and Genetic Algorithm based Data Clustering.* International Journal of Advanced Computer and Mathematical Sciences **3 (2)** (2012), 257-265.

43. M. Mahmoudi and K. Shahanaghi: *A Genetic Algorithm For P-Median Location Problem.* IJERA **3 (1)** (2013), 386-389.

44. L. A. Kazakovtsev, V. I. Orlov, A. A. Stupina and V. L. Kazakovtsev: *Modified genetic algorithm with greedy heuristic for continuous and discrete p-median problems.* Facta universitatis - series: Mathematics and Informatics **30 (1)** (2015), 89-106.

45. N. Alibabaie, M. Ghasemzadeh and C. Meinel: *A variant of genetic algorithm for non-homogeneous population.* ITM Web of Conferences **9** (2017), 02001, DOI:10.1051/itmconf/20170902001.

46. O. Hall and I. Barak, J. C. Bezdek: *Clustering with a genetically optimized approach.* IEEE Trans. Evo. Computation **3(3)** (1999), 103-112.

47. I. P. Rozhnov, V. I. Orlov and L. A. Kazakovtsev: *VNS-Based Algorithms for the Centroid-Based Clustering Problem.* Facta Universitatis - Series: Mathematics and Informatics **34 (5)** (2019), 957-972, DOI:10.22190/FUMI1905957R.

48. *Individual household electric power consumption Data Set.* UCI Machine Learning Repository [http://archive.ics.uci.edu/ml/datasets/Individual+household+electric+power+consumption], access date 28.05.2020.

49. V. I. Orlov and V. V. Fedosov: *ERC clustering dataset* (2016) [http://levk.info/Data1526_7parts.csv].

50. V. I. Orlov, D. V. Stashkov, L. A. Kazakovtsev, I. P. Rozhnov, O. B. Kazakovtseva and I. R. Nasyrov: *Improved method of forming production batchs of electronic components with special quality requirements.* Modern high technology. **1** (2018), 37-42.

51. X. Tao, X. Hu and Y. Liu: *Review of big data research.* Journal of System Simulation (2013), 142-146.

52. Y.Wang, X. Jin and X. Cheng: *Network Big Data: Status and Prospect.* Chinese Journal of Computers **06** (2013), 3-16.

Riu Li
Reshetnev Siberian State University of Science and Technology
Department of Systems Analysis and Operations Research
prosp. Krasnoyarskiy Rabochiy, 31
660037 Krasnoyarsk, Russia
646601833@qq.com

Lev A. Kazakovtsev
Reshetnev Siberian State University of Science and Technology
Department of Systems Analysis and Operations Research
prosp. Krasnoyarskiy Rabochiy, 31
660037 Krasnoyarsk, Russia
levk@bk.ru