UDC 519.6

Vestnik SibGAU
Vol. 16, No. 2, P. 317–325

GREEDY HEURISTIC METHOD FOR LOCATION PROBLEMS

L. A. Kazakovtsev*, A. N. Antamoshkin

Reshetnev Siberian State Aerospace University
31, Krasnoyarsky Rabochy Av., Krasnoyarsk, 660037, Russian Federation
*E-mail: levk@bk.ru

Authors consider multi-source location problems, k-means, k-median and k-medoid. Such problems are popular models in logistics (optimal location of factories, warehouses, transportation hubs etc.) and cluster analysis, approximation theory, estimation theory, image recognition. Various distance metrics and gauges allow using these models for clustering various kinds of data: continuous and discrete numeric data, Boolean vectors, documents. Wide area of application of such problems leads to growing interest of researchers in Russia and worldwide. In this paper, the authors propose a new heuristic method for solving such problems which can be used as a standalone local search method (local search multi-start) or as the main part of a new algorithm based on ideas of the probability changing method. For the parameters self-tuning of such algorithm, the authors propose new meta-heuristic which allows using new algorithm without learning specific features of each solved problem. Algorithms were tested on various data sets of size up to 160000 data vectors from the UCI repository and real data of semiconductor devices examination. For testing purposes, various distance metrics were used. Computational experiments showed the high efficiency of new algorithms in comparison with local search methods used traditionally for the considered problems. In addition, results were compared with the evolutionary methods and a deterministic algorithm based on the Information Bottleneck Clustering method. Such comparison illustrated the ability of new algorithms to reach higher preciseness of the results in reasonable time.

Keywords: p-median, k-medoid, k-means, Information Bottleneck Clustering.

Вестник СибГАУ
Т. 16, № 2. С. 317–325

МЕТОД ЖАДНЫХ ЭВРИСТИК ДЛЯ ЗАДАЧ РАЗМЕЩЕНИЯ

Л. А. Казаковцев*, А. Н. Антамошкин

Сибирский государственный аэрокосмический университет имени академика М. Ф. Решетнева
Российская Федерация, 660037, г. Красноярск, просп. им. газ. «Красноярский рабочий», 31
*E-mail: levk@bk.ru

Рассматриваются задачи множественного размещения – k-средних, k-медианная и задача k-медоид. Данные задачи находят широкое применение как в качестве логистических моделей оптимального размещения складов, производств, транспортных узлов и т. д., так и опосредованно – в качестве наиболее популярных моделей кластерного анализа, автоматической группировки, теории аппроксимации, теории оценивания, распознавания образов. При этом использование различных метрик расстояния или мер сходства позволяет применять данные задачи в качестве моделей автоматической группировки данных самой разнообразной природы: непрерывных и дискретных числовых данных, векторов булевых значений, документов. Широтой области применения рассматриваемых задач определяется возрастающий интерес к ним со стороны как российских, так и зарубежных исследователей. Предложен новый эвристический метод решения таких задач, который может применяться как в качестве самостоятельного алгоритма локального поиска (мультистарт локального поиска), так и в составе нового алгоритма, использующего идеи метода изменяющихся вероятностей (МИВЕР). Для автоматической настройки параметров алгоритма схемы МИВЕР предложена новая метаэвристика, позволяющая использовать разработанный алгоритм без предварительного анализа особенностей конкретной задачи. Работа алгоритмов протестирована на различных наборах данных размерностью до 160000 векторов данных. В качестве тестовых примеров использованы как типовые наборы данных из репозитория UCI, так и реальные данные отбраковочных испытаний полупроводниковых приборов. При этом были использованы различные метрики расстояния. Эксперименты показывают высокую эффективность новых алгоритмов в сравнении с традиционными для данных задач методами локального поиска. Проведено сравнение новых алгоритмов с некоторыми эволюционными алгоритмами, а также с детерминированным

алгоритмом, реализующим метод Information Bottleneck Clustering («информационного бутылочного горлышка»), показана способность новых алгоритмов достигать более высокой точности получаемых результатов за приемлемое время.

Ключевые слова: p-медианная задача, k-медоид, k-средних, метод «информационного бутылочного горлышка».

Introduction. The *k*-means problem is one of the classical problems of the continuous location theory [1–3]. The aim of this problem is searching for *k* points (cluster centers, centroids, medoids) in a *d*-dimensional space such that the sum of distances from each of given points called demand points, data vectors etc. reaches its minimum:

$$F\left(X_1, \ldots, X_k\right) = \\ = \arg\min_{X_1, \ldots, X_k \in R^d} \sum_{i-1}^{N} \min_{X \in \{X_1, \ldots, X_k\}} \left\| A_i - X \right\|_2^2. \quad (1)$$

Squared Euclidean distances and distance metrics based on the Euclidean norm is most commonly used as the distance gauge [1]. However, usage of the rectilinear (rectangle, Manhattan) metric [4] allows to obtain results (coordinates of each center) of the same accuracy (quantity of decimal digits) as given data vectors. In this case, value of each coordinate coincides with one of corresponding coordinates of data vectors [3; 5]. Moreover, using rectilinear metric reduces the influence of the outliers represent some non-standard data vectors or improperly measured values. Another approach leading to the results of the same accuracy as the initial data is solving *k*-medoids problem [6; 7]. In this case, searching for cluster centers (points minimizing the total distance) is performed on the set of data vectors only.

In (1), we have a problem with "classical" squared Euclidean distance. However, other metrics or gauges can be used. Let us denote the distance between *X* and *Y* as *L*(*X, Y*). The *k*-means problem and the *k*-medoids problem are special cases of the *p*-median (*k*-median) problem:

$$F\left(X_1, \ldots, X_k\right) = \\ = \arg\min_{X_1, \ldots, X_k \in R^d} \sum_{i-1}^{N} w_i \min_{X \in \{X_1, \ldots, X_k\}} L\left(A_i, X\right). \quad (2)$$

Here, $w_i > 0$ are weight coefficients. In the case of *k*-means problem, all $w_i = 1$.

The *k*-medoids problem can be stated as follows:

$$F\left(X_1, \ldots, X_k\right) = \arg\min_{X_1, \ldots, X_k \in \{A_1, \ldots, A_N\}} \times \\ \times \sum_{i-1}^{N} w_i \min_{X \in \{X_1, \ldots, X_k\}} L\left(A_i, X\right).$$

Known methods. The most commonly used algorithms for solving *k*-means problem include the ALA procedure (Alternating Location-Allocation) which repeats two simple steps:

Algorithm 1. ALA procedure.

Required: data vectors $A_1 \ldots A_N$, *k* initial cluster centers $X_1 \ldots X_k$.

1. For each center X_i, define its cluster C_i as the subset of data vectors having closest center X_i.

2. For each cluster C_i, recalculate its center X_i:

$$X_i = \arg\min_{X} \sum_{Y \in C_i} w_i L\left(X, Y\right).$$

3. Repeat from Step 1 is steps 1, 2 made any changes.

This procedure can be used for solving continuous *k*-median, *k*-means and *k*-medoids problems. In the general case of *k*-medoids problem, for calculating new cluster center, enumeration of all cluster members is needed. Faster similar local search procedures exist [8–10], however, they do not guarantee an exact solution.

The "classical" *k*-means method with squared Euclidean distances (l_2^2) has an important advantage. In this case, new cluster center recalculation is a simplest task solved in a single step. An average (weighted average) value of each coordinate in the cluster is the corresponding coordinate value of the center [1]. If *i*th cluster center $X_i = (x_{i,1}, \ldots, x_{i,d})$ is a *d*-dimensional vector and data vectors $A_j = (a_{i,1}, \ldots, a_{j,d})$, $j = \overline{1, N}$ are also *d*-dimensional vectors then new center of *i*th cluster for the *k*-median problem is [1]:

$$x'_{i,k} = \sum\nolimits_{y \in C_i} w_i y_k \Big/ \sum w_i, k = \overline{1, d}.$$

If the ALA procedure runs with the rectilinear metric (l_1) then the value of each cluster center coordinate is calculated separately as median (weighted median) value of the corresponding coordinate of data vectors which are cluster members. Such procedure can be described as follows.

Algorithm 2. Calculating *i*th cluster center (median) with the l_1 metric.

1. For each $k = \overline{1, d}$:

1.1. Arrange data vectors $A_i = \left(a_{j,1}, \ldots, a_{j,d}\right) \in C_i$ in ascending order. Denote such sequence $a'_{1,k}, \ldots, a'_{|C_i|,k}$. Here $\left|C_i\right|$ is the power of a set.

1.2. Calculate $m = \left[\left|C_i\right| \Big/ 2\right]$. Store $x'_{i,k} = a'_{m,k}$. Here, square brackets mean integer part.

1.3. Repeat 1.

2. Return $X'_i = \left(x'_{i,1}, \ldots, x'_{i,d}\right)$.

Except several special cases, *k*-means, *k*-medoids and *k*-median problems are *NP* hard problems requiring global search [11].

The result of the ALA procedure depends on selection of the initial centers. Usually, initial centers are selected among data vectors. The *k*-means++ procedure [12] is preferred in comparison with the chaotic selection of the initial centers. The *k*-means++ procedure guarantees accuracy $O(\log(p))$. However, such accuracy cannot be acceptable for most practically important problems. In this case, various initial centers recombination techniques are used.

Authors propose many techniques optimizing the ALA procedure: sampling [13] (solving a simplified problem on a randomly selected subset of data vectors and using the result as the initial solution for solving the

initial problem), various streaming algorithms for running on big data [14] etc.

Dependence of the ALA and other local search procedures on the given initial data embarrasses the reproducibility of the results: the same data vectors can belong to different clusters or the same cluster depending on chosen initial centers. Thus, problem of developing an algorithm resulting in precise and stable result is needed.

Perfect results of clustering and classification problems can be obtained by using the Information Bottleneck Clustering method (IBC) [15]. Running this algorithm starts with considering each data vector as a separate cluster. Then superfluous clusters are eliminated one by one unless we have k clusters. Each time, this algorithm eliminates the cluster center which gives minimal total distance increase after its elimination. Such algorithms are extremely slow [15]. Genetic algorithms with greedy heuristic [16] use the same principles. However, such algorithms developed initially for solving the k-median problems on a network are compromise methods which allow solving bigger problems. Versions of such algorithms described in article [17] can be used for solving continuous problems. The approach can be described as follows [2].

Algorithm 3. Genetic algorithm with greedy heuristic for k-median problems.

Required: Population size N_p.

1. Form (randomly with uniform distribution or using the k-means++ procedure) N_p different initial solutions $\chi_1, ..., \chi_{N_p} \subset \{\overline{1,N}\}$, $|\chi_i| = k \forall i = \overline{1,N_p}$. Each of such solutions is a subset of power k used as an initial solution of the ALA procedure. Here and after, for each of the initial solutions, the fitness function $F_{fitness}(\chi)$ is evaluated by Algorithm 4. Resulting values are stored in variables $f_1, ..., f_{N_p}$.

2. If the stop condition is reached the STOP. The result is the initial solution χ_{i^*} having the minimal corresponding value f_i. For finding the final solution, Algorithm 1 runs again.

3. Select randomly two indexes $k_1, k_2 \in \{\overline{1,N}\}$, $k_1 \neq k_2$.

4. Form an interim solution $\chi_c = \chi_{k_1} \cup \chi_{k_2}$.

5. If $|\chi_c| > k$ then go to step 7.

6. Calculate $j^* = \arg\min_{j \in \chi_c} F_{fitness}(\chi_c \{j\})$. Eliminate j^* from χ_c : $\chi_c = \chi_c \setminus \{j^*\}$. Go to step 5.

7. If $\exists i \in \{\overline{1,N_p}\} : \chi_i = \chi_c$ then go to step 2.

8. Select index $k_3 \in \{\overline{1,N_p}\}$ as follows. Select randomly two indexes $k_4, k_5 \in \{\overline{1,N}\}$. If $f_{k_4} > f_{k_5}$ then $k_3 = k_4$ else $k_3 = k_5$.

9. Replace χ_{k_3} and corresponding fitness function value: $\chi_{k_3} = \chi_c$, $f_{k_3} = F_{fitness}(\chi_c)$. Go to step 2.

Steps 5, 6 of this algorithm realize so called greedy heuristic eliminating sequentially the centers from the interim solution.

Analogous greedy heuristic was proposed in 1963 by Kuehn and Hamburger [18]. The Information Bottleneck Clustering method is based on the same principle of sequential cluster elimination [15]. Both method of Kuehn and Hamburger and the IBC method form the initial unfeasible solution as the set of all data vectors.

The fitness function can be evaluated as follows:

Algorithm 4. Calculating fitness function $F_{fitness}(\chi)$.

Required: initial solution χ.

1. Run Algorithm 1 from initial centers $\{A_i \mid i \in \chi\}$, form set $\{X_1, ..., X_p\}$ of centers.

2. Return $F_{fitness}(\chi) = \sum_{i=1}^{N} w_i \min_{j \in \{1,p\}} L(X_j, A_i)$.

Using this algorithm requires much computational resources. An alternative approach [17] is using the total distance (2) immediately as the fitness function in Algorithm 3. In this case, step 1 of Algorithm 4 is omitted. Actually, this approach solves the k-medoid problem and uses its solution as the initial centers set of the ALA procedure at the final iteration of the greedy heuristic. Such approach reduces the computational complexity, however, it reduces the accuracy.

An indisputable advantage of the Information Bottleneck Clustering method is its determinancy: this method does not use any random values, thus, each run results in the same result. In general case, if an algorithm uses random values, the precise reproduction of the results is impossible. Using the IBC method slows down the computation utterly. Thus, development of an algorithm giving sufficiently precise results is important for solving many practical problems.

In [3], authors propose a modification of the greedy heuristic used at step 6 of Algorithm 3. In this case, sets of points representing cluster centers in d-dimensional space are used as an alphabet of the genetic algorithm instead of the sets of data vector indexes. Authors call such modification Genetic Algorithm with Floating Point Alphabet. Results of the genetic algorithm, with this new heuristic are significantly more precise than the results of the modification proposed in [17] (k-problem solution, in fact). At the same time, iterations of Algorithm 3 with this heuristic can be performed faster in comparison with using Algorithm 4. Moreover, abrupt decrease of the computational complexity of step 6 of Algorithm 3 allows using this algorithm for large-scale problems: in [3], authors present results for problems up to 560000 data vectors. Such heuristic can be described as follows [3].

Algorithm 5. Greedy heuristic for the genetic algorithm with floating point alphabet used instead of steps 4–7 of Algorithm 3).

Required: set of data vectors $V = (A_1, ..., A_N) \in \mathrm{R}^d$, quantity k of clusters, two "parent" sets of centers χ_{k_1} and χ_{k_2}, parameter α.

1. Set $\chi_c = \chi_{k_1} \cup \chi_{k_2}$. Run ALA procedure from initial set of centers χ_c. Store the result to χ_c.

2. If $|\chi_c| = k$ then start the ALA procesure from the initial solution χ_c then STOP and return the result χ_c.

3. Calculate distance from each data vector to the nearest center in χ_c:

$$d_i = \min_{X \in \chi_c} L(X, A_i) \; \forall i = \overline{1, N}.$$

Form clusters around centers χ_c:

$$C_i = \arg\min_{X \in \chi_c} L(X, A_i) \; \forall i = \overline{1, N}.$$

Calculate distances from each data vector to the second closest center in set χ_c:

$$d_i = \min_{Y \in \chi_c \setminus \{C_i\}} L(Y, A_i) \; \forall i = \overline{1, N}.$$

4. For ezch center $X \in \chi_c$, calculate $\delta_X = F(\chi_c \setminus \{X\}) = \sum_{i: X \in C_i} (D_i - d_i)$.

5. Calculate $n_\delta = \max\left\{ \left[\alpha \left(|\chi_c| - k \right) \right], 1 \right\}$. Sort values δ_X in ascending order and select subset $\chi_{e\lim} = \{X_1, ..., X_{n_\delta}\}$ of n_δ data vectors with minimal values δ_X.

6. For each $j \in \left\{ \overline{2, |\chi_{e\lim}|} \right\}$: if $\exists q \in \left\{ \overline{1, (j-1)} \right\}$: $L(X_j, X_q) < L_{\min}$ then remove X_j from set $\chi_{e\lim}$. Here, $L_{\min} = \min_{X \in \chi_c} \left\{ \max \left\{ L(X, X_j), L(X, X_q) \right\} \right\}$.

7. Store $\chi_c = \chi_c \setminus \chi_{e\lim}$.

8. Rearrange the data vectors to the closest centers:

$$C_i^* = \arg\min_{X \in \chi_c} L(X, A_i) \; \forall i = \overline{1, N}.$$

9. For each $X \in \chi_c$: if $\exists i \in \left\{ \overline{1, N} \right\}$: $C_i = X$ and $C_i^* \neq X$ then recalculate center X^* of cluster $C_X^{clust} = \left\{ A_i \mid C_i^* = X, i = \overline{1, N} \right\}$. Store $\chi_c = \left(\chi_c \setminus \{ X^* \} \right) \cup \{X\}$.

10. Go to step 2.

Here, an importan parameter α determines the part of superfluous centers eliminated in a single iteration. Autrors [3] propose value 0.2. Higher values accelerate the algorithm but reduce the preciseness. Small values lead to eliminating the centers one-by-one in accordance with Algorithm 4. In this paper, we used value 0.25.

Iterations of this heuristic combine the fast greedy heuristic [16; 17; 19] eliminating up to 20–25 % superfluous centers with an iteration of the modified ALA procedure. Algorithm 4 requires $p(k_0\text{-}k)$ runs of the ALA procedure (here, k_0 is quantity of initial centers), Algorithm 5 reduces number of iterations down to $O(\log(k_0\text{–}k))$. Moreover, each iteration does not require running the whole ALA procedure. Instead, only its separate optimized steps (location – allocation) are performed.

Obviously, if $k_0 = N$ then Algorithm 5 is similar with the Information Bottleneck Clustering: number of centers (and clusters) at the first iteration coincides with the number of data vectors. Moreoved, if $k_0 = N$ is not random. Thus, a simple deterministic algorithm can be constructed [20].

Algorithm 6. Deterministic algorithm with greedy heuristic.

Required: set of data vectors $V = (A_1, ..., A_N) \in \mathrm{R}^d$, number of clusters k, parameter α.

1. Set $\chi_c = V$.

2. Start Algorithm 5 from step 2.

Table shows somparison of results of various algorithms. This algorithm is deterministic but not very precise.

New algorithms. Deterministic Algorithm 6 is a special case of Algorithm 6 with initial value $|\chi_c| = N$. The crossover procedure of the genetic algorithm with floating point (Algorithm 5) starts from a joint interim solution $\chi_c = \chi_{k_1} \cup \chi_{k_2}$. If $k << N$ then in most cases (at least at the initial iterations of the GA), elements of the "parent" sets χ_{k_1} and χ_{k_2} do not coincide and Algorithm 5 has $|\chi_c| = 2k$ at its initial iteration. In [21; 22], authors propose various efficient modifications of the greedy heuristic for genetic algorithm with partial joining of the "parent" sets. In this case, $|\chi_c| \in \left\{ \overline{(k+1), 2k} \right\}$. The GA with recombination of fixed length subsets [10] operates sets of power k only. All listed algorithms are most efficient for different problem classes. Thus, depending on problem class, various power $|\chi_c|$ of the initial solution can be optimal.

Let us denote quantity of centers in the initial solution as $k_{init} = |\chi_c|$ and a ratio of the superfluous centers to k_{init} as

$$\beta = (k_{init} - k) / k.$$

Greedy heuristic in its original form (steps 4–7 of Algorithm 3) and modified form with floating point alphabet (Algorithm 5) can be used as a standalone method for solving the problem. An algorithm can be described as follows.

Algorithm 7. Greedy heuristic method with multistart.

Required: sed of data vectors $V = (A_1, ..., A_N) \in R^d$, number k of clusters, parameters α and β.

1. Calculate $k_{init} = k + \lceil \beta k \rceil$. Select randomly subset $\chi_c \in V$ of power k_{init}.

2. Start Algorithm 5 from step 2.

3. Check the stop conditions and repeat from step 1.

Such approach requires indicating the value of parameter β. Results given in table (see algorithm "GH") show the importance of this parameter.

Let us consider a known method of solving pseudo-Boolean optimization problems, Probability Changing Method [23–25]. The main idea of this method is selecting elements from some given set randomly in

accordance with the probability vector depending on previously aclieved results. An algorithm based on this method was proposed for solving the *p*-median problem on a network [26; 27]. However, such algorithm has very slow convergence and can be used as an aiding method for the genetic algorithm [26]. Similar algorithm can be used for approximate solution of the *k*-median problem with an arbitrary distance function [28]. Inclusion of the greedy heuristic into the Probability Changing Method leads to new efficient algorithm:

Algorithm 8. Adaptive greedy heuristic method for continuous location problems.

1. Set equal values of probabilities $p_i = 1/N$ for all $i = \overline{1,N}$. Set $\beta = 0.5$.

2. For each $j = \overline{1,N_{pop}}$ do:

2.1. Copy the probabilities $q_i = p_i$ for all $i = \overline{1,N}$. Set $\chi_c = \varnothing$. Generate randomly $r \in [0;2)$ with uniform distribution. Calculate $k_{init} = k + \lceil \beta r k \rceil$.

2.2. While $|\chi_c| < k_{init}$ do:

2.2.1. Generate random $Q \in \left[0; \sum_{i=1}^{N} q_i\right)$ with uniform distribution. Set $S = 0$; $l = 1$.

2.2.2. While $S + q_l < Q$ do: $S = S + q_l$; $l = l + 1$; repeat 2.2.2.

2.2.3. Set $\chi_c = \chi_c \cup \{A_l\}$; set $q_l = 0$.

2.2.4. Repeat 2.2.

2.3. Copy the initial set $\xi_c = \chi_c$.

2.4. Start Algorithm 5 from step 2. Store the result $f_j = F(\chi_c)$; set $\chi_j = \chi_c$; $\beta_j = (k_{init} - k)/k$.

2.5. Next iteration 2.

3. Calculate $$\beta = \beta \sum_{j=1}^{N_{pop}} \frac{\beta_j \left(N_{pop} - n_j\right)^2}{\sum_{l=1}^{N_{pop}} \left(N_{pop} - n_j\right)^2 \sum_{l=1}^{N_{pop}} \beta_j},$$

here, n_j is the number of value f_j in an ascending sequence os these values.

If $4\beta k > N$ then set $\beta = N/4k$.

4. Chose index b having minimal corresponding value f_b and index w having maximal f_w.

5. For each $i = \overline{1,N}$ do:

5.1. If $A_i \in \xi_b$ and $A_i \notin \xi_w$ then set $p_i = \gamma p_i$. Else if $A_i \in \xi_w$ and $A_i \notin \xi_b$ then $p_i = p_i / \gamma$.

5.2. Next iteration 5.

6. Check the stop conditions and repeat step 2.

This algorithm requires indicating values of parameters N_{pop} (population size) and γ (probability change coefficient). Small populations are enough for this algorithm. We used $N_{pop} = 9$ and $\gamma = 1{,}1$. Parameter β is adjusted with special meta-heuristic (step 3).

New algorithms and the deterministic algorithm with greedy heuristic [20] have an important feature. Many clustering problems (see for example [19]) are decomposed into series of *k*-means problems with $k = \overline{k_{\min}, k_{\max}}$ where $k_{\min} = 1$ (the whole data set is supposed to be a single cluster) and $k_{\max}$ is equal to some reasonable value. Algorithm 6 can be used for a problem with $k = k_{\max}$ and then Algorithm 5 from step 2 can be started again for obtaining other results down to $k = k_{\min}$. Thus, all results for $k = \overline{k_{\min}, k_{\max}}$ can be obtained in a single loop.

Computational results. For testing new methods, we used data sets from the UCI repository [29], real data of examination of the EEE components [19] and randomly generated points on a plane with uniform distribution. Some results are given in table. For comparison purposes, we solved the problems with Information Bottleneck Clustering method [15; 20], deterministic algorithm with greedy heuristic [20], the genetic algorithm for p-median and p-medoids problems based on recombination of fixed length subsets [10], the genetic algorithm with classical crossover procedure and the genetic algorithm with greedy heuristic and floating point alphabet for continuous p-median problem [3]. Computational tests show high efficiency of new algorithms. In many cases, new adaptive algorithm shows the best results.

In many cases, a single start of the greedy heuristic cannot be completed within the given time limit. Running time of the algorithm depends on parameter β. In such cases, table contains dashes. Problems with data sets are not solved with deterministic algorithms either due to huge amount of time needed.

The optimal value of parameter α remains an open question. Note that if $\alpha = 0.001$ then centers are eliminated from the interim solution one-by-one in accordance with the original greedy heuristic. For the deterministic algorithm with greedy heuristic [20], value $\alpha = 0.25$ guarantees faster result (not the most precise) in accordance with results obtained with $\alpha = 0.001$. In the case of stochastic algorithm, it is not easy to foresee which value leads to better result in reasonable time. As we can see in table, data vectors quantity (compare data sets MissAmerica1, BIRCH3 and examination of diodes) and metric do not determine the optimal value of this parameter. Developing a meta-heuristic algorithms adjusting this parameter seems to be promising. Adjusting parameter α with fixed value of β can be easily performed. Simultaneous adjustment of both parameters requires factor analysis for evaluating the influence of each parameter on the results. Thus, developing a co-evolutionary algorithm with two concurrent populations having different fixed values of parameter α seems to be simpler. An example of analogous algorithm with concurrent populations evolving with various bionic algorithms was proposed in [30].

Nevertheless, it is obvious that new adaptive algorithm allows to obtain better results than multiple start of the ALA procedure and multiple start of the greedy heuristic with fixed β.

Used algorithms: “ALA” is ALA procedure multistart; “GA-MIX” is genetic algorithm with fixed length subsets recombination [10]; “GA-GHFP” is genetic algorithm with greedy heuristic and floating point alphabet [3]; “GA-classic” is GA with classical recombination, “IBC” is Information Bottleneck Clustering [15; 20]; “Determ. GH” is deterministic algorithm with greedy heuristic [20]; “GH” is new Algorithm 7; “GL” is multistart of the original greedy heuristic with local search (steps 4–7 of Algorithm 3 at $\alpha = 0.001$ or steps 4–7 of Algorithm 3 with fast elimination of centers from the interim solution similar with steps 5–7 of algorithm 5 at $\alpha = 0.25$); “GH” adapt is adaptive greedy heuristic (Algorithm 8). If no result was achieved within specified time limit, the table contains a dash.

Results of various methods

Data set, number of data vectors, dimension, data type	Number of clusters k, metric, problem type	Algorithm	Time, sec.	Average result of 30 runs: total distance (2) $F(X_1, \ldots, X_k)$		Std. deviation of the result (30 runs)	
				Parameter $\alpha = 0.25$	Parameter $\alpha = 0.001$	Parameter $\alpha = 0.25$	Parameter $\alpha = 0.001$
Examination tests of diode, $N = 701$, $d = 18$, real	30, l_2^2 normed.	ALA	15		4896.913		9.41069
		GA-MIX	15		4867.103		2.13825
		GA-GHFP	15		4810.655		1.37712
		GA-classic	15		4882.609		8.51498
		IBC	989.3		4887.930		Determ.
	k-means	Determ. GH	0.02/0.9	4890.675	4888.21	Determ.	Determ.
		GH, $\beta = 0.5$	15	4855.235	4855.534	12.9294	7.6843
		GH, $\beta = 1$	15	4835.241	4825.410	4.0288	6.4184
		GH, $\beta = 3$	15	4834.817	4821.546	7.6421	5.2087
		GL $\beta = 0.5$	15	4854.183	4853.432	10.848	8.8929
		GL, $\beta = 1$	15	4839.104	4829.781	2.8491	3.1062
		GL, $\beta = 3$	15	4836.167	4820.190	4.1217	3.1191
		GH adapt.	15	4827.753	4832.583	1.3607	1.6819
BreastCancer (UCI), $N = 699$, $d = 10$, cathegories	20, Jaccard,	ALA	5		184.1		0.8725
		GA-MIX	5		172.81		0.3650
		GA-GHFP	5		172.62		0.0787
		GA-classic	5		182.98		1.0415
		IBC	370.16		Determ.		Determ.
	k-medoids	Determ. GH	0.4/0.7	175.2	175.7	Determ.	Determ.
		GH, $\beta = 0.5$	5	181.95	181.62	0.816	0.453
		GH, $\beta = 1$	5	177.10	177.77	1.108	0.513
		GH, $\beta = 3$	5	175.00	175.63	0.155	0.755
		GL $\beta = 0.5$	5	183.93	184.43	0.859	2.312
		GL, $\beta = 1$	5	181.43	181.60	3.163	1.012
		GL, $\beta = 3$	5	–	–	–	–
		GH adapt.	5	175.39	175.67	1.741	0.468
Ionosphere (UCI), $N = 351$, $d = 10$, real	20, l_1,	ALA	4		2530.40		2.117
		GA-MIX	4		2526.77		0.0257
		GA-GHFP	4		2527.00		0.0082
		GA-classic	4		2526.93		0.1302
		IBC	370.16		Determ.		Determ.
	k-means	Determ. GH	0.03/0.2	2531.03	2533.31	Determ.	Determ.
		GH, $\beta = 0.5$	4	2526.81	2526.79	0.0286	0.0222
		GH, $\beta = 1$	4	2526.85	2526.81	0.0816	0.0222
		GH, $\beta = 3$	4	2526.86	2526.92	0.0821	0.1100
		GL $\beta = 0.5$	4	2528.02	2527.83	0.4523	0.5249
		GL, $\beta = 1$	4	2528.33	2527.92	0.9843	0.7072
		GL, $\beta = 3$	4	2528.99	2529.99	1.0165	0.8595
		GH adapt.	4	2526.79	2526.79	0.0231	0.0205
BIRCH3 (UCI), $N = 100000$, $d = 2$, real	100, l_2^2,	ALA	60		$3.825 \cdot 10^{13}$		$3.060 \cdot 10^{11}$
		GA-MIX	60		$3.886 \cdot 10^{13}$		$5.155 \cdot 10^{11}$
		GA-GHFP	60		$3.751 \cdot 10^{13}$		$0.878 \cdot 10^{11}$
		GA-classic	60		–		–
		IBC	86400		–		–
	k-means	Determ. GH	86400	–	–	–	–
		GH, $\beta = 0.5$	60	$3.832 \cdot 10^{13}$	$3.735 \cdot 10^{13}$	$5.359 \cdot 10^{11}$	$0.759 \cdot 10^{11}$
		GH, $\beta = 1$	60	$3.827 \cdot 10^{13}$	–	$4.017 \cdot 10^{11}$	–
		GH, $\beta = 3$	60	–	–	–	–
		GL $\beta = 0.5$	60	–	–	–	–
		GL, $\beta = 1$	60	–	–	–	–
		GL, $\beta = 3$	60	–	–	–	–
		GH adapt.	60	$3.772 \cdot 10^{13}$	$3.722 \cdot 10^{13}$	$3.802 \cdot 10^{11}$	$0.216 \cdot 10^{11}$
MissAmerica1 (UCI), $N = 6400$, $d = 16$, real	100, l_2^2,	ALA	60		717488.7		1107.24
		GA-MIX	60		698055.6		460.62
		GA-GHFP	60		698054.5		406.42
		GA-classic	60		714946.3		685.23
		IBC	86400		–		–
	k-means	Determ. GH	93/9977	703786.4	707529.5	Determ.	Determ.
		GH, $\beta = 0.5$	60	714287.9	713244.8	499.037	2263.9
		GH, $\beta = 1$	60	705838.9	–	334.656	–
		GH, $\beta = 3$	60	709468.4	–	195.671	–
		GL $\beta = 0.5$	60	–	–	–	–
		GL, $\beta = 1$	60	–	–	–	–
		GL, $\beta = 3$	60	–	–	–	–
		GH adapt.	60	702860.8	708239.0	576.207	781.084

Completion of table

Data set, number of data vectors, dimension, data type	Number of clusters k, metric, problem type	Algorithm	Time, sec.	Average result of 30 runs: total distance (2) $F(X_1, \ldots, X_k)$		Std. deviation of the result (30 runs)	
				Parameter $\alpha = 0.25$	Parameter $\alpha = 0.001$	Parameter $\alpha = 0.25$	Parameter $\alpha = 0.001$
Generated problem, $N = 500$, $d = 2$, real	20, metric based on angular, distances, k-median	ALA	4		4461.24		25.9199
		GA-MIX	4		4509.49		14.4368
		GA-GHFP	4		4326.72		4.7587
		GA-classic	4		4392.95		79.6913
		IBC	3141		4381.34		Determ.
		Determ. GH	4.9/23.5	4377.12	4376.43	Determ.	Determ.
		GH, $\beta = 0.5$	4	4350.36	4378.30	15.1717	15.4234
		GH, $\beta = 1$	4	4333.02	4371.50	5.8614	38.5620
		GH, $\beta = 3$	4	4389.32	–	22.5053	–
		GL $\beta = 0.5$	4	4428.97	4453.65	59.4167	34.3130
		GL, $\beta = 1$	4	4379.42	4359.41	46.1732	22.5052
		GL, $\beta = 3$	4	4429.37	–	78.2398	–
		GH adapt.	4	4313.67	4309.43	3.7077	1.4427

Conclusion. The greedy heuristic method can be used as a standalone random stochastic or deterministic [20] method for solving continuous k-median, k-mean and k-medoids problems and as a recombination method in genetic algorithms [3; 19; 21]. More precise results can be achieved in appropriate time by using the modified greedy heuristic in a self-adjusting algorithm based on ideas of the probability changing method.

References

1. Farahani R. Z., Hekmatfar M. (editors). Facility Location Concepts, Models, Algorithms and Case Studies. Springer-Verlag Berlin Heidelberg, 2009, 550 p.

2. Kazakovtsev L. A., Stupina A. A. Fast Genetic Algorithm with Greedy Heuristic for p-Median and k-Means Problems. *IEEE 2014 6th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, St.-Petersburg, October 6–8, 2014. 2014, P. 702–706.

3. Kazakovtsev L. A., Antamoshkin A. N. Genetic Algorithm wish Fast Greedy Heuristic for Clustering and Location Problems. *Informatica*. 2014, Vol. 38, Iss. 3, P. 229–240.

4. Deza M. M., Deza E. Encyclopedya of Distances. Springer Verlag. Berlin, Heilderberg, 2009, 590 p., DOI: 10.1007/978-3-642-00234-2_1.

5. Trubin V. A. [An efficient algorithm for the Weber problem with rectilinear metric]. *Kibernetika*. 1978, Iss. 6, P. 67–70, DOI:10.1007/BF01070282/ (In Russ.).

6. Kaufman L., Rousseeuw P. J. Finding Groups in Data: an Introduction to Cluster Analysis. New York: John Wiley & Sons, 1990, 342 p., DOI: 10.1002/9780470316801.ch1.

7. Park H. S., Jun C.-H. A simple and fast algorithm for K-medoids clustering. *Expert Systems with Applications*. 2009, Vol. 36, P. 3336–3341.

8. Lucasius C. B., Dane A. D., Kateman G. On *K*-Medoid Clustering of Large Data Sets with the Aid of a Genetic Algorithm: Background, Feasibility and Comparison. *Analytical Chimica Acta*. 1993, Vol. 282, P. 647–669, DOI: 10.1007/s10732-006-7284-z.

9. Zhang Q., Couloigner I. A New Efficient *K*-Medoid Algorithm for Spatial Clustering. *ICCSA 2005, LNCS*. 2005, Vol. 3482, P. 181–189, DOI: 10.1007/11424857_20.

10. Sheng W., Liu X. A Genetic *K*-Medoids Clustering Algorithm. *Journal of Heuristics*. 2004, Vol. 12, Iss. 6, P. 447–466, DOI: 10.1007/s10732-006-7284-z.

11. Cooper L. Location-allocation problem. *Oper. Res*. 1963, Vol. 11, P. 331–343, DOI: 10.1287/opre.11.3.331.

12. Arthur D., Vassilvitskii S. *k*-Means++: the Advantages of Careful Seeding. *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. 2007, P. 1027–1035, DOI: 10.1.1.360.7427.

13. Mishra N., Oblinger D., Pitt L. Sublinear time approximate clustering. *12th SODA*. 2001, P. 439–447.

14. Ackermann M. R. et al. StreamKM: A Clustering Algorithm for Data Streams. *J. Exp. Algorithmics*. 2012, Vol. 17, Article 2.4, DOI: 10.1145/2133803.2184450/.

15. Sun Zh., Fox G., Gu W., Li Zh. A parallel clustering method combined information bottleneck theory and centroid-based clustering. *The Journal of Supercomputing*. 2014, Vol. 69, Iss. 1, P. 452–467. DOI: 10.1007/s11227-014-1174-1.

16. Alp O., Erkut E., Drezner Z. An Efficient Genetic Algorithm for the p-Median Problem. *Annals of Operations Research*. 2003, Vol. 122, Iss. 1–4, P. 21–42. DOI: 10.1023/A:1026130003508.

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

18. Kuehn A. A., Hamburger M. J. A heuristic program for locating warehouses. *Management Science*. 1963, Vol. 9, Iss. 4, P. 643–666, DOI:10.1287/mnsc.9.4.643.

19. Kazakovtsev L. A., Orlov V. I., Stupina A. A., Masich I. S. [Problem of electronic components classifying]. *Vestnik SibGAU*. 2014, No. 4(56), P. 55–61 (In Russ.).

20. Kazakovtsev L. A. [Deterministic algorithm for the *k*-means and *k*-medoids problems]. *Sistemy upravleniya I informatsionnye tekhnokogii*. 2015, No. 1(59), P. 95–99 (In Russ.).

21. Kazakovtsev L. A., Stupina A. A., Orlov V. I. [Modification of the genetic algorithm with greedy heuristic ffor continuous location and classification problems]. *Sistemy upravleniya i informatsionnye tekhnokogii*. 2014, No. 2(56), P. 35–39 (In Russ.).

22. Kazakovtsev L. A., Orlov V. I., Stupina A. A., Kazakovtsev V. L. Modified Genetic Algorithm with Greedy Heuristic for Continuous and Discrete p-Median Problems. *Facta Universitatis Series Mathematics and Informatics*. 2015, Vol. 30, Iss. 1, P. 89–106.

23. Antamoshkin A. N. Brainware for Searchal Pseudoboolean Optimization. *Transactions of the Tenth Prague Conference Czechoslovak Academy of Sciences Volume.* 1988, P. 203–206, DOI: 10.1007/978-94-009-3859-5_16.

24. Kazakovtsev L. A. [Parallel random search algorithm with adaptation for shared memory systems]. *Sistemy upravleniya I informatsionnye tekhnokogii*. 2012, No. 3 (49), P. 11–15 (In Russ.).

25. Kazakovtsev L. A. Random Constrained Pseudo-Boolean Optimization Algorithm for Multiprocessor Systems and Clusters. *ICUMT 2012, International Congress on Ultra-Modern Telecommunications*. IEEE Press. S-Petersburg. 2012, P. 650–656, DOI: 10.1109/ICUMT.2012.6459711.

26. Antamoshkin A. N., Kazakovtsev L. A. Random Search Algorithm for the *p*-Median Problem. *Informatica*, 2013, Vol. 37, Iss. 3, P. 267–278.

27. Antamoshkin A. N., Kazakovtsev L. A. [Application of the probability changing method for optimal location problems on a network]. *Vestnik SibGAU*. 2014, No. 5(57), P. 10–19 (In Russ.).

28. Antamoshkin A. N., Kazakovtsev L. A. [Random search algorithm for the generalized Weeber problem in a discrete coordinate system]. *Informatika i systemy upravleniya*. 2013, No. 1, P. 87–98 (In Russ.).

29. Patrick M. Murphy P. M., Aha D. W. UCI Repository of Machine Learning Databases. 1994. Available at: http://www.cs.uci.edu/mlearn/mlrepository.html (accessed 02.01.2015).

30. Akhmedova Sh. A., Semenkin E. S. New optimization metaheuristic based on cooperation of biology related algorithms. *Vestnik SibGAU*. 2013, No. 4(50), P. 92–99 (In Russ.).

Библиографические ссылки

1. Facility Location Concepts, Models, Algorithms and Case Studies / R. Z. Farahani, M. Hekmatfar (editors). Berlin Heidelberg : Springer-Verlag, 2009. 550 p.

2. Kazakovtsev L. A., Stupina A. A. Fast Genetic Algorithm with Greedy Heuristic for *p*-Median and *k*-Means Problems // *IEEE 2014 6th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)* (October 6–8, 2014. St.-Petersburg). 2014. P. 702–706.

3. Kazakovtsev L. A., Antamoshkin A. N. Genetic Algorithm wish Fast Greedy Heuristic for Clustering and Location Problems // *Informatica*. 2014. Vol. 38, iss. 3. Pp. 229–240.

4. Deza M. M. and Deza E. Encyclopedya of Distances. Berlin-Heilderberg : Springer Verlag, 2009. 590 p. DOI: 10.1007/978-3-642-00234-2_1.

5. Трубин В. А. Эффективный алгоритм для задачи Вебера с прямоугольной метрикой // *Кибернетика*. 1978. № 6. С. 67–70. DOI:10.1007/BF01070282/.

6. Kaufman L. and Rousseeuw P. J. Finding Groups in Data: an Introduction to Cluster Analysis. New York : John Wiley & Sons, 1990. 342 p. DOI: 10.1002/9780470316801.ch1.

7. Park H. S., Jun C.-H. A simple and fast algorithm for *K*-medoids clustering // *Expert Systems with Applications*. 2009. Vol. 36. P. 3336–3341.

8. Lucasius C. B., Dane A. D., Kateman G. On *K*-Medoid Clustering of Large Data Sets with the Aid of a Genetic Algorithm: Background, Feasibility and Comparison // *Analytical Chimica Acta*. 1993. Vol. 282. P. 647–669. DOI: 10.1007/s10732-006-7284-z.

9. Zhang Q., Couloigner I. A New Efficient K-Medoid Algorithm for Spatial Clustering // *ICCSA 2005, LNCS*. 2005. Vol. 3482. P. 181–189. DOI: 10.1007/11424857_20.

10. Sheng W., Liu X. A Genetic *K*-Medoids Clustering Algorithm // *Journal of Heuristics*. 2004. Vol. 12, iss. 6. P. 447–466. DOI: 10.1007/s10732-006-7284-z.

11. Cooper L. Location-allocation problem // *Oper. Res*. 1963. Vol. 11. P. 331–343, DOI: 10.1287/opre.11.3.331.

12. Arthur D., Vassilvitskii S. *k*-Means++: the Advantages of Careful Seeding // *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. 2007. P. 1027–1035. DOI: 10.1.1.360.7427.

13. Mishra N., Oblinger D., Pitt L. Sublinear time approximate clustering // *12th SODA*. 2001. P. 439–447.

14. StreamKM: A Clustering Algorithm for Data Streams / M. R. Ackermann [et al.] // *J. Exp. Algorithmics*. 2012. Vol. 17. Article 2.4. DOI: 10.1145/2133803.2184450/.

15. A parallel clustering method combined information bottleneck theory and centroid-based clustering / Zh. Sun [et al.] // *The Journal of Supercomputing*. 2014. Vol. 69, iss. 1. P. 452–467. DOI: 10.1007/s11227-014-1174-1.

16. Alp O., Erkut E., Drezner Z. An Efficient Genetic Algorithm for the *p*-Median Problem // *Annals of Operations Research*. 2003. Vol. 122, iss. 1–4. P. 21–42. DOI: 10.1023/A:1026130003508.

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

18. Kuehn A. A., Hamburger M. J. A heuristic program for locating warehouses // *Management Science*. 1963. Vol. 9, iss. 4. P. 643–666. DOI:10.1287/mnsc.9.4.643.

19. Задача классификации электронной компонентной базы / Л. А. Казаковцев [и др.]. // *Вестник СибГАУ*. 2014. № 4(56). С. 55–61.

20. Казаковцев Л. А. Детерминированный алгоритм для задачи *k*-средних и *k*-медоид // *Системы управления и информационные технологии*. 2015. № 59. С. 95–99.

21. Казаковцев Л. А., Ступина А. А., Орлов В. И. Модификация генетического алгоритма с жадной эвристикой для непрерывных задач размещения и классификации // *Системы управления и информационные технологии*. 2014. № 2(56). С. 35–39.

22. Modified Genetic Algorithm with Greedy Heuristic for Continuous and Discrete p-Median Problems / L. A. Kazakovtsev [et al.] // *Facta Universitatis Series Mathematics and Informatics*. 2015. Vol. 30, iss. 1. P. 89–106.

23. Antamoshkin A. N. Brainware for Searchal Pseudoboolean Optimization // *Transactions of the Tenth Prague Conference Czechoslovak Academy of Sciences Volume*. 1988. P. 203–206. DOI: 10.1007/978-94-009-3859-5_16.

24. Казаковцев Л. А. Параллельный алгоритм случайного поиска с адаптацией для систем с распределенной памятью // *Системы управления и информационные технологии*. 2012. № 3(49). С. 11–15.

25. Kazakovtsev L. A. Random Constrained Pseudo-Boolean Optimization Algorithm for Multiprocessor Systems and Clusters // *ICUMT 2012, International Congress on Ultra-Modern Telecommunications*. S-Petersburg : IEEE Press, 2012. P. 650–656. DOI: 10.1109/ICUMT.2012.6459711.

26. Antamoshkin A. N., Kazakovtsev L. A. Random Search Algorithm for the *p*-Median Problem // *Informatica*, 2013. Vol. 37, iss. 3. P. 267–278.

27. Антамошкин А. Н., Казаковцев Л. А. Применение метода изменяющихся вероятностей для задач размещения на сети // *Вестник СибГАУ*. 2014. № 5(57). С. 10–19.

28. Антамошкин А. Н., Казаковцев Л. А. Алгоритм случайного поиска для обобщенной задачи Вебера в дискретных координатах // *Информатика и системы управления*. 2013. № 1. С. 87–98.

29. Patrick M. Murphy P. M., Aha D. W. UCI Repository of Machine Learning Databases [Электронный ресурс]. 1994. URL: http://www.cs.uci.edu/mlearn/mlrepository.html (дата обращения: 02.01.2015).

30. Akhmedova Sh. A., Semenkin E. S. New optimization metaheuristic based on co-operation of biology related algorithms // *Вестник СибГАУ*. 2013. № 4(50). С. 92–99.