DETERMINISTIC ALGORITHM WITH AGGLOMERATIVE HEURISTIC FOR LOCATION PROBLEMS*

L Kazakovtsev and A Stupina

1 PhD in Engineering, Assoc. Professor Siberian State Aerospace University named after Academician M.F. Reshetnev
31 “Krasnoyarskiy Rabochiy” prospect, Krasnoyarsk, 660037, Russia.
2 Professor Siberian State Aerospace University named after Academician M.F. Reshetnev
31 “Krasnoyarskiy Rabochiy” prospect, Krasnoyarsk, 660037, Russia.
Professor Siberian State University, 3 Vuzovskii, Krasnoyarsk, 660025, Russia
E-mail: h677hm@gmail.com

Abstract. Authors consider the clustering problem solved with the k-means method and p-median problem with various distance metrics. The p-median problem and the k-means problem as its special case are most popular models of the location theory. They are implemented for solving problems of clustering and many practically important logistic problems such as optimal factory or warehouse location, oil or gas wells, optimal drilling for oil offshore, steam generators in heavy oil fields. Authors propose new deterministic heuristic algorithm based on ideas of the Information Bottleneck Clustering and genetic algorithms with greedy heuristic. In this paper, results of running new algorithm on various data sets are given in comparison with known deterministic and stochastic methods. New algorithm is shown to be significantly faster than the Information Bottleneck Clustering method having analogous preciseness.

The k-means problem can be classified as a continuous problem of the location theory [1, 2, 3]. The aim is to find k points (centers, centroids, medoids) in a d-dimensional space such that the total distance from each of the data vectors (known points, measurement result vectors) to the nearest of k chosen centers reaches its minimum:

\[ F(X_1, ..., X_k) = \sum_{i=1}^{N} \min_{k \in \{1, ..., k\}} w_i \| A_i - X_k \|_2^2. \]  

(1)

Here, \( w_i \) are nonnegative weight coefficients. If all weight coefficient are equal to 1 then we have a k-means problem.

Euclidean metric is commonly used for logistic problems concerned with real objects location on a plane or in a 3D space. Such objects are factories, warehouses [1], gas or oil wells [4, 5, 6], steam generators in heavy oil fields [7], transportation or telecommunication nodes [1].

In case of clustering problems, the squared Euclidean norm is most popular distance metric.

* Results were obtained in the framework of the state task № 346 of the Ministry of Education and Science of the Russian Federation
used for calculating differences (distances) in a normalized space of characteristics [1]. Using the rectilinear (Manhattan) [8] norm as a distance metric in the k-means model allows to reach results of the same precision as the precision of data vectors. In this case, the value of each coordinate of the result coincides with the value of the same coordinate of one of the data vectors [1, 9]. Moreover, the result of the k-means problem rectilinear metric are more stable under the influence of the outliers in the data which exist due to measurement errors and defective components in the lot. Other approach which allows to achieve the results of the same precision as data vectors is solving the k-medoid problem [8, 9]. In this problem, the cluster centers which minimize the total distance are searched among the data vectors only.

The k-means method uses the ALA procedure (Alternating Location-Allocation) which includes 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 $X_i$, determine its cluster $C_i$ as a subset of the data vectors for which this center $X_i$ is the closest one.
2. For each cluster $C_i$, recalculate its center $X_i$.

$$X_i = \arg \min_x \sum_{y \in C_i} \|x - y\|$$

3. Repeat Step 1 unless Steps 1, 2 made no change in data.

Traditional usage of the $k$-means method with squared Euclidean metric ($l^2$) has one important advantage: in this case, calculating a center of a cluster is a simple problem solved in one iteration as calculating the mean value for each coordinate of all data vectors in the cluster. These mean values are coordinates of the new cluster center [1]. If the $i$-th cluster center $X_i=(x_{i,1}, \ldots, x_{i,d})$ is a vector in a $d$-dimensional space and data vectors $A_j=(a_{j,1}, \ldots, a_{j,d})$, $j=1,N$ also have $d$ dimensions then the new cluster center for a $k$-means problem with squared Euclidean metric is calculated as [1]:

$$x_{i,k} = \sum_{y \in C_i} y_k / |C_i| k = 1,d .$$

In the ALA procedure with the rectilinear metric, each coordinate of the cluster center is calculated as the median value of this coordinate among all data vectors which belong to the cluster. This process can be described as follows.

**Algorithm 2.** Calculating $i$-th cluster center (median) in case of the $l_1$ metric.

1. For each $k = 1,d$ do:
   1.1. Sort vectors $A_i=(a_{i,1}, \ldots, a_{i,d}) \in C_i$ in an increasing order of the $k$-th coordinate, form a sequence $a_{i,k}^{1}, \ldots, a_{i,k}^{|C_i|}$ . Here, $|C_i|$ is the power of the set (cluster).
   1.2. Calculate $m = \left[ |C_i| / 2 \right]$ . Store $x_{i,k} = a_{m,k}^{i}$ . Here, $[\bullet]$ is the integer part.

1.3. Next iteration 1.
2. Return $X_i=(x_{i,1}, \ldots, x_{i,d})$

This algorithm returns a value of new center which has values of each coordinate coinciding with a coordinate of some data vector.

In the case of the $k$-medoid problem, procedure of determining of each cluster center requires the exhaustive search among all data vectors in the cluster. However, many researchers propose faster analogous local search procedures [11, 12, 13] which do not guarantee an exact solution.
Except special cases, the \( k \)-means and \( k \)-medoids problems are \( NP \) hard and require global search [14].

The result of the ALA procedure depends on the choice of the initial cluster centers. Known \( k \)-means++ algorithm [15] has an advantage in comparison with the chaotic choice of the initial centers and guarantees the statistical preciseness of the result \( O(\log(p)) \). However, such preciseness is insufficient for many practically important problems. For such cases, researchers propose various recombination techniques for initial center sets [1].

The ALA procedure can be optimized with use of many techniques. For example, sampling procedures [16] solve the \( k \)-means problem for the randomly selected subset of the data vectors and use the achieved result as an initial set of centers for solving the original problem. Authors propose various algorithms for streaming data processing [17] applicable for big data analysis and many other methods.

The dependence of the results of the ALA procedure on the initial centers seeding is a serious problem for the reproducibility of the classification algorithm results: depending on the initial centers seeding, different algorithm starts classify the same data vectors as elements of various clusters. Thus, an algorithm for solving \( k \)-means problem which returns a stable result is preferred.

The Information Bottleneck Clustering method (IBC) is a deterministic method for solving the cluster analysis and classification problems able to achieve perfect results in many cases [18]. This algorithm starts from considering each data vector as a separate cluster. Then, clusters are removed one-by-one until the desired clusters quantity remains. Each time, the algorithm eliminates such cluster that its elimination gives the smallest increment of the objective function value. For the \( k \)-means problem, this algorithm eliminates the cluster center which gives the smallest total distance from data vectors to the closest remaining centers. Such algorithms are extremely slow [18]. The genetic algorithms with greedy heuristic initially designed for the discrete \( k \)-median problem on a network [19] are compromise variants. However, they are not determined algorithms. In [20], author proposes an approach for adaptation of these algorithms for the continuous location problems. The idea of such approach can be described as follows [17].

**Algorithm 3.** Genetic algorithm with greedy heuristic for \( k \)-median problems.

**Required:** data vectors \( A_1 \ldots A_N \), population size \( N_p \).

1. Form (randomly or using the \( k \)-means++ procedure) \( N_p \) various initial solutions \( \chi_1 \ldots \chi_{N_p} \subset [1,N] \), \( |\chi_i|=k \forall i \in [1,N_p] \). Each of such solutions is a set of \( k \) data vectors used as the initial solutions of the ALA procedure. Calculate the fitness function value (total distance) \( F_{\text{fitness}}(\chi) \) for each of the initial solutions using Algorithm 3 and store the fitness function values to \( f_1 \ldots f_{N_p} \).

2. If the stop conditions are satisfied then STOP. Return solution \( \chi_{i^*} \), with the smallest value \( f_i \). For the final solution, the ALA procedure (Algorithm 1) runs.

3. Choose randomly two indexes \( k_1, k_2 \in [1,N], k_1 \neq k_2 \).

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 \( j^* = \arg\min_{j \in \chi_c} F_{\text{fitness}}(\chi_c \setminus \{j\}) \). Exclude \( j^* \) from \( \chi_c : \chi_c = \chi_c \setminus \{j^*\} \). Go to Step 5.
7. If $\exists i \in \{1, N_p\}: \chi_i = \chi_c$ then go to Step 2.
8. Choose index $k_3 \in \{1, N\}$. First, two indexes $k_4, k_5 \in \{1, N\}$ are chosen randomly, then if $f_{k_4} > f_{k_5}$ then $k_3 = k_4$ else $k_3 = k_5$.

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

Steps 5-6 of this algorithm realize the greedy heuristic, the successive elimination of the centers from an interim solution.

An analogous heuristic was proposed by Kuehn and Hamburger in 1963 [21]. The IBC method is based on the same principle of the successive elimination of the clusters from an interim solution [22]. Both method of Kuehn and Hamburger and the IBC chose an unfeasible solution coinciding with the whole data vectors set as the initial solution.

The fitness function for the $k$-means problem can be calculated for an initial or interim solution as follows:

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

Required: solution $\chi$.

1. Run Algorithm 1 with initial centers set $\{A_i|i \in \chi\}$ resulting with centers set $\{X_1, ..., X_p\}$.
2. Return $F_{\text{fitness}}(\chi) = \sum_{i=1}^{N} \sum_{j=1}^{k} w_j \frac{m_{ij}}{m_{ij}^{\chi_i}} L(X_j, A_i)$.

This algorithm is a computationally intensive procedure. Other approach [20] is based on the immediate usage of the total distance from data vectors to the closest center in the interim solution as the fitness function value for Algorithm 3. In this case, Step 1 of Algorithm 4 is omitted. In fact, this approach solves a $k$-medoid problem and adjusts the solution with the ALA procedure at the final iteration of the greedy heuristic. Such approach is much faster. However, it reduces the preciseness.

An advantage of the IBC method is its determinacy. This method does not use any random values and each start of the algorithm results in the same set of cluster centers. The exact reproduction of the results is impossible when using algorithms which include random search elements. The IBC method slows down the calculation.

In [3], authors propose a modification of the greedy heuristic used in Step 6 of Algorithm 3. This method uses points in the $d$-dimensional space instead of data vector indexes as an alphabet of the genetic algorithm. Such points represent the cluster centers. Authors entitle this modification Algorithm with Floating Point Alphabet. Results of the genetic algorithm with this heuristic are much more precise than results of the modification described in [20] (in [20], authors propose solving a $k$-medoid problem). At the same time, the iterations in Algorithm 3 with this heuristic are performed much faster than Algorithm 4 as the greedy heuristic. Moreover, decrease of the computational complexity of Step 6 of Algorithm 3 makes solving large-scale problems possible. In [3], authors report results of solving problems with up to 560000 data vectors. Such heuristic can be described as follows.

Algorithm 5. Greedy heuristic for the GA with floating point alphabet (used instead of Steps 4-7 of Algorithm 3).

Required: set of data vectors $V = (A_1, ..., A_N) \in \mathbb{R}^d$, number $k$ of clusters, two “parent” center sets $\chi_{k_1}$ and $\chi_{k_2}$, parameter $\alpha$. 
1. Form interim solution \( \chi_c = \chi_{k_1} \cup \chi_{k_2} \). Run ALA algorithm from initial solution \( \chi_c \). Store its result to \( \chi_c \).
2. If \( |\chi_c| = k \) then start ALA procedure from initial solution \( \chi_c \), STOP and return \( \chi_c \).

2.1. Calculate distances to the closest element of \( \chi_c \):
\[
d_i = \min_{x \in \chi_c} L(X, A_i) \quad \forall i = 1, N.
\]
Form clusters around each center in \( \chi_c \). \( C_i = \arg \min_{x \in \chi_c} L(X, A_i) \quad \forall i = 1, N \).

3. For each center \( X \in \chi_c \), calculate \( \delta_X = F(\chi_c \setminus \{X\}) = \sum_{i : X \in C_i} (D_i - d_i) \).
4.1. Calculate \( \delta_X = \max_{i \in \chi_c} \left\{ \alpha \left| \chi_c \setminus \chi_{\min} \right| \right\} \).
Sort values \( \delta_X \) in ascending order and choose a subset \( \chi_{\min} = \{X_1, ..., X_{n_{\min}}\} \) of \( n_{\min} \) data vectors with minimal values \( \delta_X \).
4.2. For each \( j \in \{1, 2, ..., \chi_{\min}\} \) do: if \( \exists q \in \{1, (j-1)\} : L(X_j, X_q) < L_{\min} \) then eliminate \( X_j \) from set \( \chi_{\min} \).

4.3. Set \( \chi_c = \chi_c \setminus \chi_{\min} \).

4.4. Reallocate data vectors to the closest centers: \( C_i^* = \arg \min_{x \in \chi_c} L(X, A_i) \quad \forall i = 1, N \).

4.5. For each \( X \in \chi_c \) if \( \exists i \in \{1, N\} : C_i = X \) and \( C_i^* \neq X \) then recalculate center \( X^* \) of cluster \( C_X^{\text{clus}} = \{A_i | C_i^* = X, i = 1, N\} \). Store \( \chi_c = (\chi_c \setminus \{X^*\}) \cup \{X\} \).

5. Go to Step 2.

Value \( \alpha \) is an important parameter of this heuristic. This value determines the percentage of the superfluous cluster centers eliminated in a single iteration. Authors propose value 0.2. Bigger values make the algorithm run faster and reduce its preciseness. Small values \( \alpha \) make it work as Algorithm 4 and eliminate a single center at each iteration. We used \( \alpha = 0.25 \).

This heuristic combines the greedy heuristic [19, 20, 23] with elements of the modified ALA procedure performed at each iteration and allows to eliminate up to 20-25% superfluous cluster centers until the required quantity of centers remains. Algorithm 4 requires performing \( p(k_0-k) \) runs of the ALA procedure (here, \( k_0 \) is the initial centers quantity). Algorithm 5 reduces quantity of iterations down to \( O(\log(k_0-k)) \). Moreover, each iteration does not require perform the whole ALA procedure. Instead, its separate optimized elements (location – allocation) are performed.

It can be easily seen that if \( k_0 = N \) then both Algorithm 5 and IBC method start their initial iterations analogously: number of cluster centers coincides with the number of data vectors. Moreover, if \( k_0 = N \) then choosing initial centers is not random: all data vectors are chosen as the initial centers. Thus, the following deterministic algorithm can be proposed.

**Algorithm 6.** New deterministic greedy heuristic algorithm.

Required: set of data vectors \( V = (A_1, ..., A_N) \in \mathbb{R}^d \), number \( k \) of clusters, parameter \( \alpha \).
1. Set \( \chi_c = V \).
2. Run Algorithm 5 starting from Step 2.
Such algorithm was successfully applied for middle-scale problems, up to \( N = 6500 \) data vectors. The results are gathered in Table 1. We used example problems from the UCI library [23] and problems with real data of EEE components examination [1]. For small-scale problems, the results are shown in comparison with the IBC method, genetic algorithm with greedy heuristic and genetic algorithm with recombination of fixed length subsets [24]. We used various distance metrics. In addition, \( k \)-medoid problems [9, 10] were solved. For several problems, the results of new algorithm are insignificantly worse than the results of the IBC. At the same time, time needed for problem solution reduces many times. In Table 1, all results of new algorithm are shown for \( \alpha = 0.25 \) and \( \alpha = 0.001 \). Last value makes the algorithm work as an IBC procedure which eliminates exactly one center at a time. At the same time, the approach to the inclusion of elements of the ALA procedure in this greedy heuristic remains unchanged and the new algorithm with \( \alpha = 0.001 \) works slower than one with \( \alpha = 0.25 \) and still much faster than the IBC algorithm which uses Algorithm 4 for fitness function evaluation on each iteration. In addition, Table 1 shows results of the simplified IBC method which evaluates the fitness function value without running ALA or another local search procedures. In fact, this simplified IBC procedure solves a \( k \)-medoid problem and then adjusts the result with ALA procedure. Such algorithm is entitled “IBC w/o local search”. Such algorithm can be constructed after excluding Steps 3 and 4.5 from Algorithm 5 with \( \alpha = 0.001 \). If we remain \( \alpha = 0.25 \), i.e. we allow simultaneous eliminating several centers, we have a new deterministic algorithm entitled “IBC w/o local search, \( \alpha = 0.25 \)”. This version of the algorithm shows the best time and the worst preciseness.

For all inspected problems except those with the Jaccard metric, new algorithm shows the best results among all considered deterministic algorithms except IBC with local search which exceeds new algorithm by preciseness in several problems. However, new algorithm takes much less time. Results of new algorithm are less precise than the results of evolutionary algorithms. However, except problems with the Jaccard metric, the difference does not exceed 2.3% for problems with real data vectors and 3.8% for problems with Boolean data vectors.

Note that new algorithm has one important feature for solving a problem of automatic classification of the EEE components. Such problem is solved [23] as series of the \( k \)-means problems with \( k = k_{\text{min}}, k_{\text{max}} \) where \( k_{\text{min}} = 1 \) (a single production batch without clusters assumed) and \( k_{\text{max}} \) is chosen by a decider equal to some reasonable number. Algorithm 6 can be used for the \( k \)-means problem with \( k = k_{\text{max}} \). Then, starting from Step 2, Algorithm 5 can run again for solving the succeeding problems until \( k = k_{\text{min}} \). Thus, results for all values \( k = k_{\text{min}}, k_{\text{max}} \) can be calculated in a single run.

### Table 1. Comparative results of new algorithm

| Data set, quantity of data vectors, dimension, data type | Clusters q-ty k, metric, problem type | Algorithm | Result | Time, sec. | Std. dev. of 10 runs |
|--------------------------------------------------------|--------------------------------------|-----------|---------|-----------|---------------------|
| Chess (UCI), \( N = 3197, d=50, \) Boolean           | 50, \( l_i \), \( k \)-means          | Local search multistart   | 10015.89 | 30        | 19.08               |
|                                                       | GA with fixed subsets recomb.        | GA with float, point alphabet  | 9331.8  | 30        | 6.074               |
|                                                       | IBC                                  | IBC w/o local search         | 9286    | 30        | 9.135               |
|                                                       | IBC w/o local search, \( \alpha = 0.25 \) | IBC w/o local search, \( \alpha = 0.25 \) | 9796    | 29.18     | -                   |
|                                                       | New algorithm, \( \alpha = 0.25 \)   | New algorithm, \( \alpha = 0.25 \) | 10057   | 0.688     | Determin.           |
|                                                       |                                      |                                      | 9649    | 0.696     | Determin.           |
### Conclusion

Proposed algorithm allows solving $k$-means and $k$-medoids problems in appropriate time. Achieved results are insignificantly less precise than the results of the evolutionary algorithms. However, new algorithm is deterministic and this fact makes its results easy for checking and interpreting by all concerned parts.

### References

[1] Farahani R Z and Hekmatfar M editors 2009 Facility Location Concepts, Models, Algorithms and Case Studies, Springer-Verlag Berlin Heidelberg.

[2] Kazakovtsev L A, Stupina A A 2014 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 pp.702-706.

[3] Kazakovtsev L A, Antamoshkin A N 2014 Genetic Algorithm wish Fast Greedy Heuristic for Clustering and Location Problems Informatica 38(3) pp. 229-240.

[4] Eiselt H A 1992 Location Modelling in Practice American Journal of Mathematical and Management Sciences 12(1) pp. 3-18, DOI: 10.1080/01966324.1992.10737322

[5] Kabadi S, Murty K G, Spera C 1996 Clustering Problems in Optimization Models Computational Economics 9 pp.229-239

[6] Church R L, Scaparra M P, Middleton R S 2004 Identifying Critical Infrastructure: The Median and Covering Facility Interdiction Problems Annals of the Association of American Geographers 94(3) pp.491-502.

[7] Rosing K E 1992 The optimal location of steam generators in large heavy fields American Journal of Mathematical and Management Sciences 12 pp.19-42.

[8] Deza M M and Deza E 2009 Encyclopedya of Distances Springer Verlag, Berlin, Heiderberg.

[9] Trubin V A 1978 Effektivnyy algoritm dlya zadachi Vebera s pryamougol'nnoy metrikoy Kibernetika 6 pp.67-70, DOI:10.1007/BF01070282/

[10] Kaufman L and Rousseuw P J 1990 Finding Groups in Data: an Introduction to Cluster Analysis. New York: John Wiley & Sons.

[11] Park H S, Jun C-H 2009 A simple and fast algorithm for K-medoids clustering Expert Systems with Applications 36 pp.3336–3341

[12] Lucasius C B, Dane A D and Kateman G 1993 On K-Medoid Clustering of Large Data Sets with the Aid of a Genetic Algorithm: Background, Feasibility and Comparison Analytical Chimica Acta 282 pp.647-669.

[13] Zhang Q, Couloigner I 2005 A New Efficient K-Medoid Algorithm for Spatial Clustering ICCSA 2005, LNCS 3482 pp. 181–189.

[14] Cooper L 1963 Location-allocation problem Oper. Res. 11 pp. 331–343

[15] Arthur D and Vassilvitskii S 2007 k-Means++: the Advantages of Careful Seeding Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms pp. 1027–1035

[16] Mishra N, Oblinger D, Pitt L 2001 Sublinear time approximate clustering 12th SODA pp. 439–447

[17] Ackermann M R et al. 2012 StreamKM: A Clustering Algorithm for Data Streams J. Exp. Algorithmics vol.17 article 2.4, DOI: 10.1145/2133803.2184450/

[18] Sun Zh, Fox G, Gu W, Li Zh 2014 A parallel clustering method combined information bottleneck theory and centroid-based clustering The Journal of Supercomputing 69(1) pp. 452–467. DOI: 10.1007/s11227-014-1174-1.

[19] Alp O, Erkut E, Drezner Z 2003 An Efficient Genetic Algorithm for the p-Median Problem Annals of Operations Research 122(1–4) pp. 21–42

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

[21] Kuehn A A, Hamburger M J 1963. A heuristic program for locating warehouses Management Science 9(4) pp.643-666

[22] Kazakovtsev L A, Orlov V I, Stupina A A, Masich I S 2014 Problem of electronic components classifying Vestnik SibGAU 4(56) pp.55-61.
[23] Patrick M Murphy P M, Aha D W 1994 UCI Repository of Machine Learning Databases. URL http://www.cs.uci.edu/ mlearn/mlrepository.html (accessed 02.01.2015)

[24] Sheng W, Liu X 2004 A Genetic K-Medoids Clustering Algorithm Journal of Heuristics 12(6) pp. 447-466, DOI: 10.1007/s10732-006-7284-z