Contents lists available at SciVerse ScienceDirect

Swarm and Evolutionary Computation

journal homepage: www.elsevier.com/locate/swevo

Regular paper

A combined approach for clustering based on K-means and gravitational search algorithms

Abdolreza Hatamlou [a,b,*], Salwani Abdullah [b], Hossein Nezamabadi-pour [c]

[a] *Islamic Azad University, Khoy Branch, Iran*
[b] *Data Mining and Optimization Research Group, Center for Artificial Intelligence Technology, Universiti Kebangsaan Malaysia, 43600 Bangi, Selangor, Malaysia*
[c] *Department of Electrical Engineering, Shahid Bahonar University of Kerman, Kerman, Iran*

ARTICLE INFO

Article history:
Received 17 September 2011
Received in revised form
23 February 2012
Accepted 29 February 2012
Available online xxxx

Keywords:
Clustering
K-means
Gravitational search algorithm

ABSTRACT

Clustering is an attractive and important task in data mining that is used in many applications. Clustering refers to grouping together data objects so that objects within a cluster are similar to one another, while objects in different clusters are dissimilar. K-means is a simple and efficient algorithm that is widely used for data clustering. However, its performance depends on the initial state of centroids and may trap in local optima. The gravitational search algorithm (GSA) is one effective method for searching problem space to find a near optimal solution. In this paper, we present a hybrid data clustering algorithm based on GSA and k-means (GSA-KM), which uses the advantages of both algorithms. The GSA-KM algorithm helps the k-means algorithm to escape from local optima and also increases the convergence speed of the GSA algorithm. We compared the performance of GSA-KM with other well-known algorithms, including k-means, genetic algorithm (GA), simulated annealing (SA), ant colony optimization (ACO), honey bee mating optimization (HBMO), particle swarm optimization (PSO) and gravitational search algorithm (GSA). Five real and standard datasets from the UCI repository have been used to demonstrate the results of the algorithms. The experimental results are encouraging in terms of the quality of the solutions and the convergence speed of the proposed algorithm.

1. Introduction

Presently, large amounts of data are stored in various databases of organisations, and with the advent of large memory systems and computer networks the amount of stored data grows very quickly. These data contain useful but hidden information that may be extracted for various purposes. Data mining is one of the effective and powerful techniques that are used to extract information and knowledge from a very large amount of data.

Clustering is a major data mining task that refers to a process of finding groups in a set of observations such that those belonging to the same group are similar, while those belonging to different groups are distinct, according to some criteria of distance or likeness. Cluster analysis is also known as unsupervised learning because it can find and recognise patterns and trends in a large amount of data without any supervision or previously obtained information such as class labels. Clustering algorithms are used in many fields and applications such as document clustering and information retrieval [1–4], data compaction [5], anomaly detection [6,7], biology [8,9], medicine [10], computer vision [11–13], construction management [14], marketing and customer analysis [15] and many other fields as both the primary task for understanding the nature and structure of data and in the pre-processing or post-processing phase for high level tasks.

There are many algorithms that have been proposed to perform clustering. However, due to a large variety of applications, different data types and various purposes of clustering, we cannot find a unique algorithm that can serve all the requirements at once. In general, clustering algorithms can be divided into two groups: hierarchical algorithms and partitional algorithms. Hierarchical clustering algorithms recursively find clusters either in an agglomerative (bottom–up) mode or in a divisive (top–down) mode. Agglomerative methods start with each data object in a separate cluster and successively merge the most similar pairs until termination criteria are satisfied. Divisive methods start with all the data objects in one cluster and repeatedly divide each cluster into smaller clusters, also until termination criteria are satisfied. On the other hand, partitional clustering algorithms find all the clusters simultaneously without forming a hierarchical structure. A well-known class of partitional clustering algorithms is the centre-based clustering method, and the most popular and widely

* Corresponding author at: Islamic Azad University, Khoy Branch, Iran.
E-mail addresses: hatamlou@ftsm.ukm.my, hatamlou@iaukhoy.ac.ir, rezahatamloo@gmail.com (A. Hatamlou), salwani@ftsm.ukm.my (S. Abdullah), nezam@mail.uk.ac.ir (H. Nezamabadi-pour).

2210-6502/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.swevo.2012.02.003

used algorithm from this class of algorithms is a k-means algorithm. K-means is simple to implement and efficient in most cases [16–19]. However, the performance of k-means is highly dependent on the initial state of centroids and may converge to the local optima rather than global optima. The k-means algorithm tries to minimise the intra-cluster variance, but it does not ensure that the result has a global minimum variance [20,21].

Over the last 20 years, many heuristic methods have been proposed to overcome this problem. For instance, a simulated annealing algorithm for the clustering problem has been presented in [22]. In [23,24], a tabu search algorithm has been applied to perform clustering. A genetic algorithm based approach has been proposed and tested on real datasets to evaluate its performance in [25]. In [26] a genetic k-means algorithm for clustering problem has been proposed. Clustering approaches based on a neural gas algorithm have been presented in [27,28]. A method for cluster analysis based on an ant colony optimisation (ACO) has been presented in [29]. An artificial bee colony approach has been proposed in [30]. In [31], a honey bee mating optimisation algorithm has also been applied to perform data clustering. Data clustering using the big bang–big crunch algorithm has been introduced in [32]. The application of the particle swarm optimisation (PSO) algorithm [33] for clustering problems has been proposed in [17,34]. A comprehensive review of clustering algorithms can be found in [35].

Another drawback which is commonly recognized as regarding the K-means algorithm consists in that the number of clusters is needed as input to the algorithm, i.e. the number of clusters is assumed known. Some approaches to solve this problem can be found in [36–39].

In this paper, we address the application of a hybrid algorithm based on the gravitational search algorithm [40] and k-means algorithm [16,18,19] on cluster analysis. The performance of the proposed approach has been tested on several standard and real datasets from the UCI repository [41] and the results have been compared with some other techniques. The rest of the paper is organised as follows: Section 2 provides a brief background on clustering problems and k-means algorithms. In Section 3, we describe our proposed algorithm for solving data clustering problems based on a gravitational search algorithm and k-means algorithm. Experimental results and comparison to other available methods are discussed in Section 4. Finally, Section 5 presents the conclusions of this research with ideas for future work.

2. Background on clustering and *K*-means algorithm

Let $O = \{O_1, O_2, \ldots, O_n\}$ be a set of n data objects. Each object is described by d features. These data points can also be represented by a profile data matrix $O_{n\times d}$ that has n row vectors, where each row vector has d dimension to represent a data object. The ith row vector $O_i = (o_i^1, o_i^2, \ldots, o_i^d)$ denotes the ith object from the dataset, and each element o_i^j in O_i is a scalar denoting the jth component or feature of the corresponding data object. Given such an $O_{n\times d}$, the aim of a clustering algorithm is to find a partition, $C = \{C_1, C_2, \ldots, C_k\}$ of K groups, that produces objects within the same group that are as similar to each other as possible, while objects that belong to different groups differ as much as possible. The resulting partitions are called clusters and should have the following properties:

- Each cluster should contain at least one data object, i.e., $C_i \neq \Phi\ \forall i \in \{1, 2, \ldots, k\}$.
- Different clusters should have no object in common, i.e., $C_i \cap C_j = \Phi$, $\forall i \neq j$ and $i, j \in \{1, 2, \ldots, k\}$.
- Each data object should be assigned to a cluster. In other words, after assigning objects to clusters, the sum of objects in all clusters should be equal to the number of objects in the original dataset, i.e., $\bigcup_{i=1}^{k} C_i = O$.

Because we can partition the given dataset in different ways while satisfying all of the criteria mentioned above, a fitness function must be defined to measure the quality of the cluster obtained. The most widely used and famous function that is used to specify the goodness of a clustering is the total mean-square quantisation error (MSE) [5], which is defined as follows:

$$f(O, C) = \sum_{l=1}^{k} \sum_{O_i \in C_l} d(O_i, Z_l)^2 \tag{1}$$

where $d(O_i, Z_l)$ specifies the dissimilarity between object O_i and the centroid of cluster C_l (Z_l) to be found by calculating the mean value of objects within the respective cluster. To calculate the dissimilarity between objects, many distance metrics have been defined. The most popular distance function is the Euclidean distance. Note that in this work, we used the Euclidean distance function to measure the dissimilarity between data objects. Given two objects X_i and X_j with d dimensions, the distance is calculated as in Eq. (2):

$$d(X_i, X_j) = \sqrt{\sum_{p=1}^{d} (x_i^p - x_j^p)^2}. \tag{2}$$

The k-means algorithm [16,18,19] is one of the most efficient and famous clustering algorithms that have been successfully applied to many applications. In general, it starts with some random or heuristic-based centroids for the desired clusters and then assigns every data object to the closest centroid. After that, the k-means algorithm iteratively refines the current centroids to reach the (near) optimal ones by calculating the mean value of data objects within their respective clusters. The algorithm will terminate when any one of the specified termination criteria is met (i.e., a predetermined maximum number of iterations is reached, a (near) optimal solution is found or the maximum search time is reached). Some of the improved variations of the K-means algorithm can be found in [42–47].

3. Proposed approach

The basic idea behind the proposed approach is the gravitational search algorithm (GSA), which is inspired by a physical phenomenon. The mechanism of GSA is based on the interaction of masses in the universe via Newtonian gravity law. The gravitation is the attraction of masses by other masses. The amount of this attraction depends on the amount of masses and the distance between them. It is defined by Newton as, “Every particle in the universe attracts every other particle with a force that is directly proportional to the product of the masses of the particles and inversely proportional to the square of the distance between them”. This definition is known as the gravitational force and is formulated by the following equation [40]:

$$F = G\frac{M_1 M_2}{R^2}. \tag{3}$$

In this equation, F is the gravitational force (typically in N), G is the gravitational constant with a value of 6.67259×10^{-11} (typically in N (m^2/kg^2)), M_1 and M_2 are the masses of first and second particles, respectively (typically in kg), and R is the straight-line distance between the two particles (typically in m).

Newton’s second law states that after gravitational force acts on a special particle, it then accelerates and moves towards other particles. The value of the acceleration depends on the particle's mass and gravitational force acting upon it, which is based on the equation given below [40].

$$a = \frac{F}{M} \tag{4}$$

where, F and M are the gravitational force and mass of a given particle, respectively.

Based on the above explanation, we can find that the gravitational force between heavy and closer particles is great.

The gravitational search algorithm applies this physical phenomenon when searching a problem space in solving optimisation problems. To describe the GSA, consider a system with N masses (agents) in which the position of the ith mass is defined as follows:

$$X_i = (x_i^1, \ldots, x_i^d, \ldots, x_i^n), \quad i = 1, 2, \ldots, N \tag{5}$$

where x_i^d is the position of the ith mass in the dth dimension and n is the total number of dimensions in the search space. The positions of the masses correspond to the solutions of the problem. Based on [40], the mass of each agent is calculated after computing a current population's fitness as follows:

$$M_i(t) = \frac{\text{fit}_i(t) - \text{worst}(t)}{\sum_{j=1}^{N}(\text{fit}_j(t) - \text{worst}(t))} \tag{6}$$

where $M_i(t)$ and $\text{fit}_i(t)$ represent the mass and the fitness value of the agent i at t, respectively, and $\text{worst}(t)$ is defined as follows (for a minimisation problem):

$$\text{worst}(t) = \max_{j\in\{1,\ldots,N\}} \text{fit}_j(t). \tag{7}$$

To compute the acceleration of an agent, the total forces from a set of heavier masses that act on it should be considered based on the law of gravity (Eq. (8)), followed by the calculation of an agent acceleration using a law of motion (Eq. (9)). After that, the next velocity of an agent is calculated as a fraction of its current velocity added to its acceleration (Eq. (10)). Then, its next position can be calculated using Eq. (11).

$$F_i^d(t) = \sum_{j\in\text{kbest}, j\neq i} \text{rand}_j G(t) \frac{M_j(t)M_i(t)}{R_{ij}(t) + \varepsilon}(x_j^d(t) - x_i^d(t)) \tag{8}$$

$$a_i^d(t) = \frac{F_i^d(t)}{M_i(t)} = \sum_{j\in\text{kbest}, j\neq i} \text{rand}_j G(t) \frac{M_j(t)}{R_{ij}(t) + \varepsilon}(x_j^d(t) - x_i^d(t)) \tag{9}$$

$$v_i^d(t+1) = \text{rand}_i \times v_i^d(t) + a_i^d(t) \tag{10}$$

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1) \tag{11}$$

where rand_i and rand_j are two uniformly distributed random numbers in the range of [0, 1], ε is a small value to avoid division by zero, and $R_{ij}(t)$ is the Euclidean distance between two agents i and j defined as Eq. (2). The set of first K agents with the best fitness value and biggest mass is kbest. kbest is a function of time, initialised to K_0 at the beginning and decreasing with time. Here, K_0 is set to N (total number of agents) and is linearly decreased to 1. G is a decreasing function of time that is set to 1 at the beginning and decreases linearly towards zero at the last iteration.

Based on the above description, we have tried to apply GSA for solving clustering problems. The quality of the resulting clusters and convergence speed of the GSA has been enhanced by incorporating a k-means algorithm in generating the initial population for GSA. It is important to create a good initial population because the performance of GSA and most of the population-based algorithms are affected by the quality of the initial population. In the proposed algorithm, we try to incorporate the advantage of the k-means algorithm into GSA. K-means is a simple and fast algorithm that is able to find a near optimal solution in a reasonable amount of time. The generated solution by k-means later will be used by GSA as one of the candidate solutions. This process increases the performance of GSA in the following two ways: (i) it decreases the number of iterations and function evaluations performed by GSA to find a near global optimum compared to the original GSA alone, and (ii) with the advent of a good candidate solution in the initial population, GSA can search for near global optima in a promising search space and, therefore, find a high quality solution in comparison with the original GSA alone.

To create a high quality population in the proposed algorithm, we also considered the nature of the input datasets. As shown in the pseudocode of the proposed algorithm, three specific candidates are created using test datasets: minimum, average and maximum values of data objects in each dataset. This process ensures that the candidate solutions are spread in the wide area of the search space, which consequently increases the chance of finding a (near) global optima.

Based on the above description, the proposed algorithm is built on three main steps. In the first step, GSA-KM applies k-means algorithm on selected dataset and tries to produce near optimal centroids for desired clusters. In the second step, the proposed approach will produce an initial population of solutions, which will be applied by the GSA algorithm in the third step. The production of an initial population is carried out in several different ways. One of the candidate solutions will be produced by the output of the k-means algorithm, which has been achieved in the previous step. Three of them will be created based on the dataset itself and other solutions will be produced randomly. This process generates a high-quality initial population, which will be used in the next step by the GSA algorithm. Finally, in the third step, GSA will be employed for determining an optimal solution for the clustering problem. The main steps of the GSA-KM algorithm are summarised below (see Fig. 1).

To represent candidate solutions in the proposed algorithm, we used one-dimensional arrays to encode the centroids of the desired clusters. The length of the arrays is equal to $d \times k$, where d is the dimensionality of the considered datasets or the number of features that each data object has and k is the number of clusters. If we consider $P_i = \{Z_1, Z_2, \ldots, Z_k\}$ as the ith candidate solution, then $Z_j = (z_j^1, z_j^2, \ldots, z_j^d)$ is the jth cluster centre for the ith agent or candidate solution ($i = 1, 2, \ldots, S$ and $j = 1, 2, \ldots, k$). S is the number of particles or candidate solutions in which its value in this work is set to 50. Fig. 2 represents an example of a candidate solution.

4. Experimental results

Five real datasets are used to validate our proposed algorithm. These datasets are named Iris, Wine, Glass, Breast Cancer Wisconsin (Cancer), and Contraceptive Method Choice (CMC). Each dataset has a different number of clusters, data objects and features [41]. These datasets have been used by many authors to compare and evaluate the performance of clustering algorithms in the literature and are described as follows:

Iris dataset ($n = 150, d = 4, k = 3$): This dataset contains three classes of 50 objects each, where each class refers to a type of iris flower. There are 150 random samples of iris flowers with four numeric attributes in this dataset. These attributes are sepal length and width in cm, petal length and width in cm. There are no missing values for attributes.

Wine dataset ($n = 178, d = 13, k = 3$): This dataset contains the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. This dataset

```
Step 1: k-means method

1.1. Randomly choose k centroids from dataset for desired clusters
1.2. Assign each data object to the cluster with the closest centroid
1.3. Update the centroids by calculating the mean values of objects within clusters
1.4. Repeat steps 1.2 and 1.3 until termination criteria are met

Step 2: Generate an initial population of size S ({$P_1$, $P_2$, …, $P_s$}).

2.1 $P_1$=k-means (dataset) // Use output of k-means as one of the candidate solutions
2.2 $P_2$=min (dataset) // Generate a candidate solution using the minimum of the dataset
2.3 $P_3$=mean (dataset) // Generate a candidate solution using the mean of the dataset
2.4 $P_4$=max (dataset) // Generate a candidate solution using the maximum of the dataset
2.5 $P_5$…$P_s$=random (dataset) // Generate all other candidate solutions randomly

Step 3: GSA method

3.1. Calculate the fitness function for all of the particles (candidate solutions)
3.2. Calculate M, F and a for all of the particles based on Eq. (6, 8 and 9) as described in the
     GSA algorithm
3.3. Update the velocity and position of particles based on Eq.(10 and 11) as described in the
     GSA algorithm
3.4. If termination criteria are met (i.e., the predefined number of iteration is reached or the
     fitness function is satisfied) output the best particle, which has the best value for the fitness
     function as the final solution; otherwise return to step 3.1.
```

Fig. 1. The main steps of the GSA-KM algorithm.

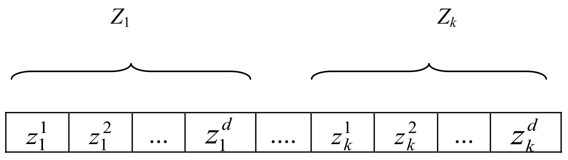

Fig. 2. Example of a candidate solution for the clustering problem.

contains 178 instances with 13 continuous numeric attributes. There are no missing attribute values.

Glass dataset ($n = 214, d = 9, k = 6$): This dataset contains samples from six different types of glass: tableware, building windows float processed, vehicle windows float processed, building windows non-float processed, containers, and headlamps. There are 214 instances with 9 numeric attributes in glass dataset.

Contraceptive Method Choice also denoted as CMC ($n = 1473, d = 10, k = 3$): This dataset is a subset of the 1987 National Indonesia Contraceptive Prevalence Survey. The samples are married women who either were not pregnant or did not know if they were at the time of interview. The problem is to predict the choice of current contraceptive method (no use has 629 objects, long-term methods have 334 objects, and short-term methods have 510 objects) of a woman based on her demographic and socioeconomic characteristics.

Breast Cancer Wisconsin ($n = 683, d = 9, k = 2$): This dataset contains 683 objects characterised by nine features: clump thickness, cell size uniformity, cell shape uniformity, marginal adhesion, single epithelial cell size, bare nuclei, bland chromatin, normal nucleoli, and mitoses. There are two clusters in the data: malignant (444 objects) and benign (239 objects).

We have compared the performance of the proposed algorithm on the selected datasets with the original GSA [48,49], PSO [17,49], GA [50], SA [22], ACO [29], HBMO [31] and *k*-means [16,49] algorithms. We have used two criteria to evaluate the performance of algorithms: (i) the sum of the intra-cluster distances, as defined in Eq. (1), and (ii) the number of fitness function evaluations. For criterion (i), note that the smaller the sum of the distances is the higher the quality of the clustering. In criterion (ii), the smaller value for the number of function evaluations indicates the high convergence speed of the considered algorithm.

The algorithm has been implemented in Matlab 7.6 on a T6400, 2 GHz, 2 GB RAM computer. Since all of the algorithms are stochastic algorithms, to counteract against the randomized nature of the algorithms and to indicate the consistency and robustness of the algorithms, 20 independent runs were conducted for each experiment.

The intra cluster distances obtained by clustering algorithms on test datasets have been summarised in Table 1. The results are the best, worst, average and standard deviation of achieved solutions from 20 simulations. The last criterion in Table 1 is the number of function evaluations (coded as NFE), which indicates the convergence speed of the respective algorithms. NFE is the number of times that the clustering algorithm has calculated the fitness function (Eq. (1)) to reach the (near) optimal solution. It is dependent on the number of iterations to reach the optimal solution.

As we can see from Table 1, the GSA-KM algorithm has produced the highest quality solutions in terms of the best, worst, and average intra cluster distances on all the test datasets. Moreover, the standard deviation of solutions found by GSA-KM is the smallest, which means that GSA-KM can find a near optimal solution in most of the runs while other algorithms may trap local optima in some of the runs. In other words, the results confirm that the proposed algorithm is viable and robust. In terms of the number of function evaluations, the *k*-means algorithm needs the least number of evaluations compared to other algorithms. The proposed GSA-KM is in the second rank as it has employed the output of the *k*-means algorithm in the initial population for the GSA phase. In other words, in the GSA-KM algorithm, the GSA method starts from a good initial state due to the use of the output of *k*-means and consequently reaches the optimal solution faster than the pure GSA because in the pure GSA alone, all candidate solutions are generated randomly and the quality of the initial population is not as good as the initial population in the GSA-KM algorithm.

On the iris dataset, the best, worst, and average intra cluster distances obtained by GSA-KM are 96.679, 96.705, and 96.689, respectively, which are much better than the distances obtained by the *k*-means algorithm. Furthermore, the standard deviation of solutions found by GSA-KM is 0.0076, while this value for the *k*-means algorithm is 14.631, which means that the *k*-means algorithm may trap in local optima in some cases while GSA-KM can converge to the optimal solution in most cases. In comparison with the original GSA alone, the best, worst, and average intra

Table 1
Simulation results for clustering algorithms.

Dataset	Criteria	*K*-means	GA	SA	ACO	HBMO	PSO	GSA	GSA-KM
Iris	Best	97.333	113.98	97.45	97.10	96.75	96.894	96.698	96.679
	Average	106.050	125.19	99.95	97.17	96.95	97.232	96.723	96.689
	Worst	120.450	139.77	102.01	97.80	97.75	97.897	96.764	96.705
	Std	14.631	14.563	2.018	0.367	0.531	0.347	0.0123	0.0076
	NFE	120	38 128	5314	10 998	11 214	4953	4628	1377
Wine	Best	16 555.68	16 530.53	16 473.48	16 530.53	16 357.28	16 345.96	16 315.35	16 294.25
	Average	18 061.00	16 530.53	17 521.09	16 530.53	16 357.28	16 417.47	16 376.61	16 294.31
	Worst	18 563.12	16 530.53	18 083.25	16 530.53	16 357.28	16 562.31	16 425.58	16 294.64
	Std	793.21	0	753.084	0	0	85.49	31.34	0.0406
	NFE	390	33 551	17 264	15 473	7238	16 532	15 300	5523
Glass	Best	215.74	278.37	275.16	269.72	245.73	270.57	220.78	211.47
	Average	235.50	282.32	282.19	273.46	247.71	275.71	225.70	214.22
	Worst	255.38	286.77	287.18	280.08	249.54	283.52	229.45	216.08
	Std	12.47	4.138	4.238	3.584	2.438	4.55	3.4008	1.1371
	NFE	630	199 892	199 438	196 581	195 439	198 765	171 910	1759
CMC	Best	5842.20	5705.63	5849.03	5701.92	5699.26	5700.98	5698.15	5697.03
	Average	5893.60	5756.59	5893.48	5819.13	5713.98	5820.96	5699.84	5697.36
	Worst	5934.43	5812.64	5966.94	5912.43	5725.35	5923.24	5702.09	5697.87
	Std	47.16	50.369	50.867	45.634	12.690	46.95	1.724	0.2717
	NFE	270	29 483	26 829	20 436	19 496	21 456	11 796	1754
Cancer	Best	2999.19	2999.32	2993.45	2970.49	2989.94	2973.50	2967.96	2965.14
	Average	3251.21	3249.46	3239.17	3046.06	3112.42	3050.04	2973.58	2965.21
	Worst	3521.59	3427.43	3421.95	3242.01	3210.78	3318.88	2990.83	2965.30
	Std	251.14	229.734	230.192	90.500	103.471	110.80	8.1731	0.0670
	NFE	180	20 221	17 387	15 983	19 982	16 290	8262	1642

cluster distances are closer to each other. However, in terms of the number of function evaluations, we can see that GSA-KM is more efficient than the original GSA. The GSA-KM converges to 96.679 at 1377 function evaluations, while the GSA converges to 96.698 at 4628 function evaluations. In other words, GSA-KM converges to the optimal solution very quickly.

For the wine dataset, GSA-KM algorithm achieved the best results compared to all other algorithms. The best, worst and average intra cluster distances obtained by GSA-KM are almost the same, near 16 294, which is significantly better than other algorithms, meaning that it can find high quality clusters compared to other methods. Note that other algorithms are unable to obtain this value even once within 20 runs. Moreover, GSA-KM can obtain the best result at 5523 function evaluations, while GSA, which is the closest competitor, needs 15 300 function evaluations to reach 16 315.35 (this is worse than the value obtained by GSA-KM). These numbers indicate that the proposed approach manages to improve the performance of *k*-means and GSA algorithms alone in both the quality criterion (i.e., intra cluster distance) and number of function evaluations that shows how fast the algorithm can converge to the best solution. For other datasets, we also can see that the GSA-KM algorithm is superior to other methods.

In brief, the results confirm that our proposed hybrid approach has three significant merits in comparison to *k*-means and GSA alone. Firstly, it causes the *k*-means algorithm to escape from local optima. Secondly, it improves the quality of solutions found by either the *k*-means or GSA algorithm alone and thirdly it increases the convergence speed of the GSA algorithm.

5. Conclusion

In this work, a hybrid method (coded as GSA-KM) that is based on a gravitational search algorithm (GSA) and *k*-means algorithm is used in clustering data objects. It tries to exploit the merits of two algorithms simultaneously, where the *k*-means is used in generating the initial solution and the GSA is employed as an improvement algorithm. The performance of the proposed algorithm is compared with other approaches. The comparison shows that the proposed algorithm overcomes the shortcomings of *k*-means and GSA alone. It requires minimum number of function evaluations to reach the optimal solution. Moreover, the proposed approach can produce high quality clusters with small standard deviation on selected datasets compared to other methods. In future research, the proposed method may be applied to other applications, such as image segmentation and university timetabling. The combination of the GSA with other heuristic approaches and their application to data clustering is another research direction.

References

[1] A. Abraham, S. Das, A. Konar, Document clustering using differential evolution, in 2006 IEEE Congress on Evolutionary Computation, CEC 2006, 2006.
[2] M. Mahdavi, et al., Novel meta-heuristic algorithms for clustering web documents, Applied Mathematics and Computation 201 (1–2) (2008) 441–451.
[3] R. Gil-Garcia, A. Pons-Porrata, Dynamic hierarchical algorithms for document clustering, Pattern Recognition Letters 31 (6) (2010) 469–477.
[4] H. Anaya-Sanchez, A. Pons-Porrata, R. Berlanga-Llavori, A document clustering algorithm for discovering and describing topics, Pattern Recognition Letters 31 (6) (2010) 502–510.
[5] S. Yang, et al., Evolutionary clustering based vector quantization and SPIHT coding for image compression, Pattern Recognition Letters 31 (13) (2010) 1773–1780.
[6] M. Friedman, et al., Anomaly detection in web documents using crisp and fuzzy-based cosine clustering methodology, Information Sciences 177 (2) (2007) 467–475.
[7] M. Moshtaghi, et al., Clustering ellipses for anomaly detection, Pattern Recognition 44 (1) (2011) 55–69.
[8] G. Kerr, et al., Techniques for clustering gene expression data, Computers in Biology and Medicine 38 (3) (2008) 283–293.
[9] F. Liang, N. Wang, Dynamic agglomerative clustering of gene expression profiles, Pattern Recognition Letters 28 (9) (2007) 1062–1076.
[10] L. Liao, T. Lin, B. Li, MRI brain image segmentation and bias field correction based on fast spatially constrained kernel clustering approach, Pattern Recognition Letters 29 (10) (2008) 1580–1588.
[11] P. Scheunders, A comparison of clustering algorithms applied to color image quantization, Pattern Recognition Letters 18 (11–13) (1997) 1379–1384.
[12] Y. Xia, et al., Image segmentation by clustering of spatial patterns, Pattern Recognition Letters 28 (12) (2007) 1548–1555.
[13] J. Fan, M. Han, J. Wang, Single point iterative weighted fuzzy *C*-means clustering algorithm for remote sensing image segmentation, Pattern Recognition 42 (11) (2009) 2527–2540.
[14] Y.-M. Cheng, S.-S. Leu, Constraint-based clustering and its applications in construction management, Expert Systems with Applications 36 (3, Part 2) (2009) 5761–5767.

[15] B. Saglam, et al., A mixed-integer programming approach to the clustering problem with an application in customer segmentation, European Journal of Operational Research 173 (3) (2006) 866–879.
[16] A.K. Jain, Data clustering: 50 years beyond K-means, Pattern Recognition Letters 31 (8) (2010) 651–666.
[17] C. Ching-Yi, Y. Fun, Particle swarm optimization algorithm and its application to clustering analysis. in Networking, Sensing and Control, 2004 IEEE International Conference on, 2004.
[18] E.W. Forgy, Cluster analysis of multivariate data: efficiency versus interpretability of classifications, Biometrics 21 (1965) 2.
[19] L. Kaufman, P.J. Rousseeuw, Finding Groups in Data: An Introduction to Cluster Analysis, John Wiley & Sons, New York, 1990.
[20] Y.-T. Kao, E. Zahara, I.W. Kao, A hybridized approach to data clustering, Expert Systems with Applications 34 (3) (2008) 1754–1762.
[21] S.Z. Selim, M.A. Ismail, K-means-type algorithms: a generalized convergence theorem and characterization of local optimality, IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-6 (1) (1984) 81–87.
[22] S.Z. Selim, K. Alsultan, A simulated annealing algorithm for the clustering problem, Pattern Recognition 24 (10) (1991) 1003–1008.
[23] K.S. Al-Sultan, A Tabu search approach to the clustering problem, Pattern Recognition 28 (9) (1995) 1443–1451.
[24] C.S. Sung, H.W. Jin, A tabu-search-based heuristic for clustering, Pattern Recognition 33 (5) (2000) 849–858.
[25] U. Maulik, S. Bandyopadhyay, Genetic algorithm-based clustering technique, Pattern Recognition 33 (9) (2000) 1455–1465.
[26] K. Krishna, M. Narasimha Murty, Genetic K-means algorithm, IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics) 29 (3) (1999) 433–439.
[27] A.K. Qin, P.N. Suganthan, Kernel neural gas algorithms with application to cluster analysis, in: Proceedings—International Conference on Pattern Recognition, 2004.
[28] A.K. Qin, P.N. Suganthan, A robust neural gas algorithm for clustering analysis, in: Proceedings of International Conference on Intelligent Sensing and Information Processing, ICISIP 2004, 2004.
[29] P.S. Shelokar, V.K. Jayaraman, B.D. Kulkarni, An ant colony approach for clustering, Analytica Chimica Acta 509 (2) (2004) 187–195.
[30] D. Karaboga, C. Ozturk, A novel clustering approach: artificial bee colony (ABC) algorithm, Applied Soft Computing 11 (1) (2011) 652–657.
[31] M. Fathian, B. Amiri, A. Maroosi, Application of honey-bee mating optimization algorithm on clustering, Applied Mathematics and Computation 190 (2) (2007) 1502–1513.
[32] A. Hatamlou, S. Abdullah, M. Hatamlou, Data clustering using big bang-big crunch algorithm, in: Communications in Computer and Information Science, 2011, pp. 383–388.
[33] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Neural Networks, 1995. Proceedings, IEEE International Conference on, 1995.
[34] P. Jin, Y.L. Zhu, K.Y. Hu, A clustering algorithm for data mining based on swarm intelligence, in: Proceedings of the Sixth International Conference on Machine Learning and Cybernetics, ICMLC 2007, 2007.
[35] S. Das, A. Abraham, A. Konar, Metaheuristic pattern clustering—an overview, Studies in Computational Intelligence (2009) 1–62.
[36] A.K. Qin, P.N. Suganthan, Robust growing neural gas algorithm with application in cluster analysis, Neural Networks 17 (8–9) (2004) 1135–1148.
[37] S. Das, A. Abraham, A. Konar, Automatic clustering using an improved differential evolution algorithm, IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans 38 (1) (2008) 218–237.
[38] S. Das, A. Abraham, A. Konar, Automatic hard clustering using improved differential evolution algorithm, Studies in Computational Intelligence (2009) 137–174.
[39] Part, et al., Automatic Clustering Based on Invasive Weed Optimization Algorithm, in: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2011, pp. 105–112.
[40] E. Rashedi, H. Nezamabadi-pour, S. Saryazdi, GSA: a gravitational search algorithm, Information Sciences 179 (13) (2009) 2232–2248.
[41] C.L. Blake, C.J. Merz, UCI repository of machine learning databases. Available from: http://www.ics.uci.edu/-mlearn/MLRepository.html.
[42] M.B. Adil, Modified global-means algorithm for minimum sum-of-squares clustering problems, Pattern Recognition 41 (10) (2008) 3192–3199.
[43] A.M. Bagirov, J. Ugon, D. Webb, Fast modified global k-means algorithm for incremental cluster construction, Pattern Recognition 44 (4) (2011) 866–876.
[44] J.Z.C. Lai, T.-J. Huang, Fast global k-means clustering using cluster membership and inequality, Pattern Recognition 43 (5) (2010) 1954–1963.
[45] A. Likas, N. Vlassis, J.J. Verbeek, The global k-means clustering algorithm, Pattern Recognition 36 (2) (2003) 451–461.
[46] P.S. Bradley, U.M. Fayyad, Refining initial points for K-means clustering, in: 15th International Conference on Machine Learning, ICML98, Morgan Kaufmann, San Francisco, 1998.
[47] S.J. Redmond, C. Heneghan, A method for initialising the K-means clustering algorithm using kd-trees, Pattern Recognition Letters 28 (8) (2007) 965–973.
[48] A. Hatamlou, S. Abdullah, H. Nezamabadi-pour, Application of gravitational search algorithm on data clustering, in: Rough Sets and Knowledge Technology, Springer, Berlin, Heidelberg, 2011, pp. 337–346.
[49] A. Hatamlou, S. Abdullah, Z. Othman, Gravitational search algorithm with heuristic search for clustering problems, in: Data Mining and Optimization (DMO), 2011 3rd Conference on.
[50] M.C. Cowgill, R.J. Harvey, L.T. Watson, A genetic algorithm approach to cluster analysis, Computers & Mathematics with Applications 37 (7) (1999) 99–108.