Applied Intelligence (2023) 53:20346–20364
https://doi.org/10.1007/s10489-023-04580-x

Multi-start local search algorithm based on a novel objective function for clustering analysis

Xiaolu Liu[1] · Wenhan Shao[2] · Jiaming Chen[1] · Zhipeng Lü[2] · Fred Glover[3] · Junwen Ding[2]

Accepted: 17 March 2023 / Published online: 6 April 2023

Abstract

Clustering is the process of partitioning data into different clusters with the goal of minimizing the difference of objects within each cluster, where the commonly used evaluation function is defined as the sum of the squared distance from each point to the cluster center to which it belongs. Nevertheless, this general evaluation function is extremely vulnerable to outliers and noisy data, and it is sensitive to initial cluster centers. More seriously, this evaluation function cannot effectively represent the core of clustering results; even if the partition achieves the global optimum value according to the evaluation function, the clustering results may not be good. In this study, we propose a multi-start local search algorithm (MLS) with several techniques to tackle this problem. First, the center of each cluster is no longer its centroid, which reduces the dependence of the cluster algorithm on difference in size and shape of ideal clusters. Second, the number of adjacent points shared between clusters is defined as the new objective function. Third, two basic meta-operations, merge and split, are used to optimize the objective function and make the iterative process insensitive to the initial solution. The novelty of our approach is the selection criterion of the initial centers and the new objective function, which enables MLS to explore more promising search area. Experimental results demonstrate that MLS outperforms traditional centroid-based clustering algorithms in terms of both solution quality and computational efficiency, and it is quite competitive to other reference algorithms such as spectral, density, and geometric based clustering algorithms.

Keywords Clustering analysis · Multi-start local search · Merge and spilt

1 Introduction

Clustering is a fundamental problem in unsupervised learning and data analysis. Its aim is to explore the implicit structure in the data without prior labeling of classes according to predefined attributes [29]. As an efficient method for deriving knowledge from big data, it is widely used in pattern recognition [30], image segmentation [32], image object extraction [72], data compression [47] and data stream processing [11]. An effective method for clustering can aggregate data having the same attributes and separate data having different attributes.

As an NP-hard problem, clustering analysis can be done by strategies that try to find subgroups of samples based on features or on the properties of samples. Clustering is generally achieved by assigning data points to clusters by reference to a chosen optimization criterion [13]. For example, [26] solves the clustering problem by seeking several cluster solutions with different number of clusters and preferring the one that minimizes the value of evaluation metrics. The K-means algorithm is undoubtedly one of the most studied centroid-based clustering algorithms [31] that has a variety of applications in domain spanning areas [36]. Its essential idea is to find a set of k centers that minimizes the sum of the squared distances from points to their closest centers and to assign each point to its closest center [46].

The main contributions and novelties of this study can be summarized as follows:

First, the center of each cluster is no longer its centroid, which reduces the dependence of the algorithm on the configuration of the points and their distribution. We select the center point based on the distances among the points, without knowing the specific coordinates or attribute values of the points. In this way, a preprocessing step is employed and therefore the objective function is calculated.

✉ Junwen Ding
junwending@hust.edu.cn

Extended author information available on the last page of the article.

Second, we define the minimization of the number of adjacent pairs (NAP) between clusters as the new objective function. Given a threshold value δ calculated from the instance, if the distance between two points is less than δ, they are deemed adjacent as illustrated in Fig. 1. K-means only focuses on the distance within clusters, while neglecting the distance between clusters. This results in that, in some cases, as the objective function value becomes small, the quality of the clustering result deteriorates because it depends on the shapes and sizes of the clusters, which are not known before clustering. By using NAP to define the objective function, our algorithm aims to minimize the overlap of the fields associated with the clusters via focusing on the cluster boundaries.

Third, we use two basic meta-operations, merge and split, to optimize the objective function and make the iterative process insensitive to the initial solution. By this approach, we select the two clusters with the largest NAP in the current solution. Then, these two clusters are merged into one cluster and a new center is chosen for them. Following this, we choose the cluster with the minimum objective value after segmentation to split it into two clusters. The above steps are repeated until the iteration converges.

2 Related work

A large number of clustering algorithms have been developed for different types of applications, which can be generally classified into the following categories: centroid, hierarchical, density, spectral, grid, and others.

Centroid-based (or partition-based) clustering algorithms attempt to partition the data points into k clusters where usually a distance-based specific criterion function is optimized. A well-known method of minimizing the sum of squared distances in clustering is the Generalized Lloyd algorithm [46], also referred to as K-means algorithm, which iteratively repartitions the data vectors and recalculates of the cluster centers until the solution converges or a predefined maximum number of iterations has been executed. Although this algorithm is easy to implement, it relies heavily on the initial solution and a poor initial solution often leads to disappointing results. Several techniques and strategies have been adopted to improve the original K-means method. The global K-means algorithm dynamically adds a cluster center one time by a deterministic global search procedure in an incremental way [34]. The well-known variant K-means++ chooses centers in a specific way for the K-means algorithm, which selects the first center randomly and the next one according to the probability that is negatively correlated with the distance between this point and the previously selected centers [56]. A comparative study of different initialization methods for the K-means clustering algorithm was presented in [14] where eight commonly used initial solution procedures were compared. Duwairi and Abu-Rahmeh [16] applied the novel heuristic for choosing the initial centroid that is known non-trivial solutions of random centroids based methods.

Hierarchical clustering algorithms create a hierarchical decomposition of the data that can be presented as a dendrogram. Previous representative works includes the balanced iterative reducing and clustering using hierarchies (BIRCH) [74], clustering using representatives (CURE) [24], etc. Recent advances in hierarchical clustering varies in a wide range: [77] proposed a novel hesitant fuzzy agglomerative hierarchical clustering algorithm for hesitant fuzzy sets. Xu et al. [67] presented a density peak based hierarchical clustering method. A novel hierarchical clustering based on minimum spanning tree algorithm was proposed in [10] for graph clustering. Fan [19] proposed optimal probabilistic estimation approach by exploiting the survival of the fittest principle. A unified validity index framework for the hierarchical clustering was proposed in [68], which improved the deficiencies of the measurements

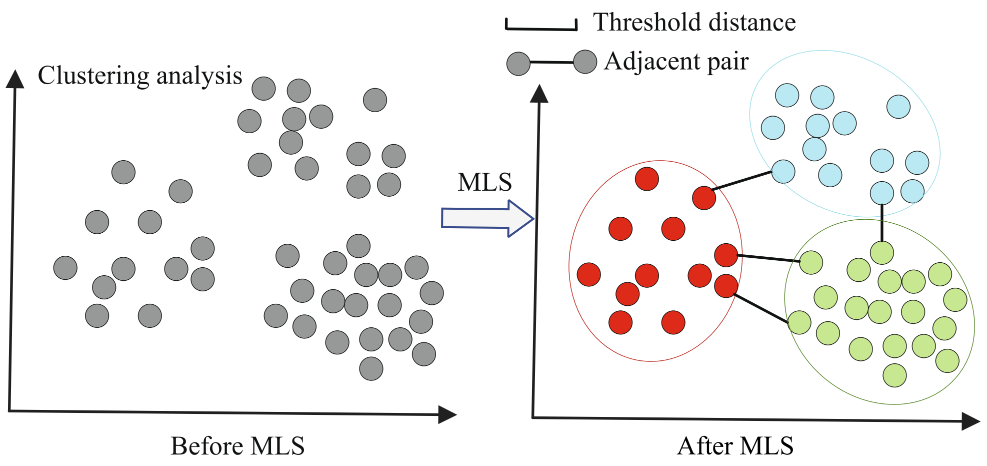


Fig. 1 The illustration of the proposed MLS clustering algorithm

in the existing validity indices. Yin et al. [71] proposed an improved hierarchical clustering algorithm based on the idea of population reproduction and fusion.

Density-based algorithms search for dense regions in the data space that are separated from one another by low density noise regions. Typical algorithms include DBSCAN [17] and OPTICS [8]. A fuzzy density peaks clustering algorithm based on an improved DNA genetic algorithm and k-nearest neighbors was proposed by [5] to assure that the subjective choice has a greater impact on the clustering results when the size is small. Xu et al. [69] proposed an improved density peaks clustering algorithm with fast finding cluster centers, which improves the efficiency of density peaks clustering algorithm by screening points with higher local density based on two novel pre-screening strategies. Based on the conception that cluster centers are featured by a higher density than their neighbors and by a relatively large distance from points with higher densities, [44] proposed a fast search method for clustering. Subsequently, [35] proposed a shared-nearest-neighbor-based clustering by fast search and find of density peaks algorithm to overcome the low robustness and poor fault tolerance.

Base on the information from the eigenvalues (spectrum) of special matrices built from the graph or the data set, spectral clustering becomes one of the most popular modern clustering algorithms. Wang et al. [61] designated a weighted-spectral clustering algorithm for detecting the community structure of a complex network with no prior knowledge of the cluster number. Zhu et al. [79] proposed a low-rank sparse subspace clustering method by dynamically learning the affinity matrix from low-dimensional space of the original data. Wang et al. [63] proposed a spectral clustering method with semantic interpretation based on axiomatic fuzzy set theory to make it competitive in both classification rate and comprehensibility. Subsequently, [55] presented a spectral clustering using density-sensitive distance measure with global and local consistencies. Besides, an ultra-scalable spectral clustering and an ensemble clustering were proposed in [28] for extremely large-scale data sets with limited resources. A deep spectral clustering method was proposed in [80], where several techniques are combined into a unified framework. Sun et al. [54] proposed a lifelong spectral clustering to learn a model for a new spectral clustering task by selectively transferring previously accumulated experience from knowledge library. An enhanced spectral clustering based on a churn prediction model was proposed in [48]. Alshammari et al. [6] proposed a method by refining a k-nearest neighbor graph for a computationally efficient spectral clustering.

The grid-based clustering methods use a multi-resolution grid data structure. It quantizes the object areas into a finite number of cells that form a grid structure on which all the operations for clustering are implemented. Sheikholeslami et al. [49] proposed the WaveCluster algorithm for clustering which was based on wavelet transforms. Wang et al. [62] presented the Sting algorithm, which is a hierarchical statistical information grid based approach for spatial data mining to reduce computational burden. A study of density-grid based clustering algorithms on data streams can be found in [7]. Wu and Wilamowski [64] proposed a fast density and grid based clustering method for data with arbitrary shapes and noise. Brown et al. [12] proposed a fast density-grid clustering method which works by dividing the data space into a grid structure and then assigning a density measurement to each grid cell.

With the development of cluster analysis in recent years, some new methods, such as subspace clustering [2], ensemble clustering [27, 53], deep embedded clustering [66], clustering with deep learning [3], multi-omic and multi-view clustering [41], graph clustering [37], and hybrid approaches [1, 40, 73], have been proposed. Frey and Dueck [21] proposed affinity propagation method which takes the measures of similarity between pairs of data points as input. Pourbahrami and Hashemzadeh [39] proposed a geometric-based clustering method which uses the concept of natural neighborhoods to extract the local density of data points. Wang et al. [59] proposed a general graph-based multi-view clustering method which takes the data graph matrices of all views and uses them to generate a unified graph matrix. A historical perspective of fuzzy clustering was reviewed in [45]. A novel binary multi-view clustering scheme was proposed in [78], which enables the handling of multi-view image data and can be applied to large data. A size-constrained label propagation was introduced in [37]. A general graph-based system for multi-view clustering was proposed in [60] where the impact of different graph metrics on the multi-view clustering performance within the proposed framework is evaluated.

Recently, several K-menas based clustering improved by metaheuristics have appeared. A K-PC algorithm based on the principle of the K-means algorithm was proposed in [23]. A novel clustering method by minimizing the difference between pairwise sample assignments for each data point was proposed in [38]. The performance of K-means and its properties and effectiveness on commonly used benchmark instances was studied by [20]. Another approach in [76] adopted a decision graph to find cluster centers easily and efficiently. Besides, An improved deep embedded clustering algorithm was proposed in [25] to handle the data structure preservation by manipulating feature space. A two-leveled symbiotic evolutionary algorithm was proposed in [51] for clustering problems. An adaptive local learning regularized non-negative matrix factorization approach was proposed in [50] for data clustering. Ashraf et al. [9] proposed a meta-heuristic based genetic algorithm to optimize the centroid initialization process. Two new algorithms based on iterative local optimization for clustering graphs and networks were proposed in [52]. A grey wolf optimization algorithm was proposed in [4] to alleviate the stagnation in local optima and premature convergence.

3 Research gaps

Through literature review, it is found that data clustering attracts wide attention from research community and a variety of algorithms of different categories are reported for different types of clustering applications. Here we focus on the research gaps in the centroid based clustering methods in the existing studies, as well as the metaheuristic algorithms incorporated to enhance the performance of centroid based methods: (1) the traditional K-menas algorithm suffers from the drawback that it is sensitive to the initial selection of the cluster as the starting points. (2) metaheuristic based clustering algorithms may stuck into local optima if the search intensification and diversification is not well balanced.

To overcome the above disadvantages, we propose in this paper an effective multi-start local search (MLS) algorithm with the goal of minimizing the number of adjacent pairs instead of the sum of the squared distance from points to their closest centers. The center of the cluster is no longer its centroid and is selected based on the distance between the points. By taking the number of adjacent pairs as the new objective function to be minimized, MLS is insensitive to the initial solution. Besides, the mlti-start local search framework iteratively searches the solution space by initialization, optimization, and adjustment phases. The interaction among the three phases creates a balance between search diversification and intensification, thus to yield an effective means for generating high-quality solutions.

4 Problem formulation

The clustering problem can be defined as follows. Consider a set X of n points, given by $X = \{x^1, \ldots, x^n\}$, or equivalently $X = \{x^i \mid i \in N\}$ where $N = \{1, \ldots, n\}$. Let $x^r = (x^r_1, \ldots, x^r_d), r \in N$ be the feature vector of the r-th point of the set X. Let $d(x^i, x^j)$ denote the distance measure defined between points x^i and x^j, $(i, j \in N)$, such as Euclidean distance.

We are interested in clustering X into subsets $C = \{C_1, \ldots, C_k\}$ for specified cluster number k. The partition $P = \{p_1, \ldots, p_n\}$ defines the cluster index to which each point is assigned (i.e., $x^i \in C_{p_i}$, $p_i \in \{1, \ldots, k\}$) and $c = \{c^1, \ldots, c^k\}$ is the set of cluster's centroids. The problem formulation is then given by

$$\text{minimize} \quad f_k = \sum_{i=1}^{n} \sum_{j=1}^{k} I\{x^i \in C_j\} \times \parallel x^i - c^j \parallel^2 \tag{1}$$

$$s.t \begin{cases} \emptyset = C_i \bigcap C_j, & \forall C_i, C_j \in C, i \neq j, \\ X = \bigcup_{i=1}^{k} C_i \end{cases} \tag{2}$$

where $I\{cond\}$ in (1) is a binary function which returns 1 if condition $cond$ holds and returns 0 otherwise. Constraint (2) ensures that each point must be assigned to exactly one cluster. The objective function (1) used in K-means minimizes the sum of squared distance between nodes that lie in the same cluster. To express the objective function proposed in this paper, we introduce the following new notations.

The *threshold value* δ determines whether two points are adjacent. If $d(x^i, x^j) < \delta$, points x^i and x^j are regarded as an *adjacent pair*. $Ns(x^i) = \{x^j \mid d(x^i, x^j) < \delta, x^j \in X\}$ denotes the set of points adjacent to point x^i. $SumAdjs(C_p, C_q) = \sum_{x^i \in C_p, x^j \in C_q} I\{d(x^i, x^j) < \delta\}$ identifies the number of adjacent pairs of points where one point lies in cluster C_p and the other lies in cluster C_q. Based on the above definition, the proposed *objective function* f_a is defined in (3), which minimizes the sum of the number of adjacent pairs between any two clusters.

$$f_a = \sum_{C_i, C_j \in C, i \neq j} SumAdjs(C_i, C_j) \tag{3}$$

5 Cluster centers and objective function

The two most important components of clustering are the selection of centers and the definition of the objective function [42]. In each iteration, the traditional K-means algorithm defines the mean of all points in each cluster to be a new center and reassigns each point to the new center closest to it to identify the new collection of clusters [34]. This method is not only time-consuming, but also depends on the coordinates of the points. More seriously, in some situations outliers or noisy data cause points to be reassigned to clusters with fewer points, while points incorporating normal data are reassigned to create several clusters which should be a single large cluster. Further processing by increasing the number of iterations

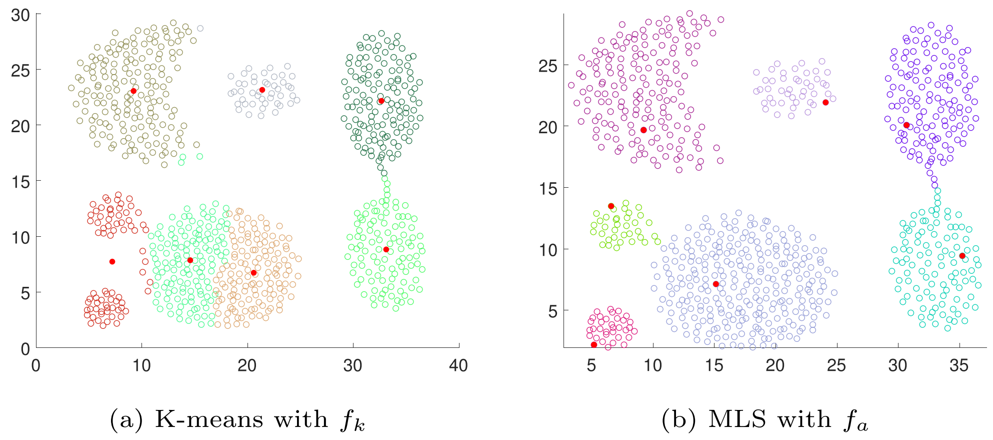


Fig. 2 Comparison of clustering results obtained by K-means and MLS

is usually not helpful in this situation. To overcome these disadvantages, we select cluster centers based on distances between points that define adjacency, utilizing the neighborhood to be described in Section 5.1. By this means, inappropriate points are prevented from becoming centers.

The most widely used objective function is the clustering error criterion, which computes the squared distance of each point from its corresponding cluster center and then takes the sum of these distances over all points in the data set [35]. Although this criterion can overcome the dependency on the initial solution if the process is restarted multiple times, it still does not work when cluster sizes are unbalanced. In other words, even if the objective function reaches the optimum value for this criterion, the resulting set of clusters may not be ideal. An example is provided in Fig. 2, where Fig. 2a is the result obtained by K-means and Fig. 2b is the result obtained by our proposed MLS procedure. Both K-means and MLS are set to run 1000 iterations. Clearly, the result on the right is significantly better than the one on the left, which illustrates that the MLS procedure, guided by the new objective function f_a, can achieve much better performance than the K-means procedure guided by f_k.

5.1 The MinSum-centroid

To avoid sparse points from hindering the search for the next cluster center point, a threshold α is introduced. In a cluster C_i, if $|Ns(x)| < \alpha, x \in C_i$, the point x is regarded as a sparse point and will not allowed to become a cluster center. The MinSum-centroid is used to select or update the centers as proposed in [20]. The sum of the distances from the point x to all the other points among the cluster C_i to which x belongs is given in (4) and the MinSum-centroid of cluster C_i is defined in (5). The advantage of choosing the center in this way is that new nodes are no longer generated to become centers in the iterative process, and only one preprocessing is needed to calculate the distance between nodes if it is not directly given (This advantage is also shared by the pseudo-centroid approach of [23]). In addition, the reference to $Ns(x)$ excludes the possibility of sparse points becoming the center, which can avoid the interference of abnormal or peripheral points.

$$SumDist(x) = \sum_{x, x^j \in C_i, x \neq x^j} d(x, x^j) \tag{4}$$

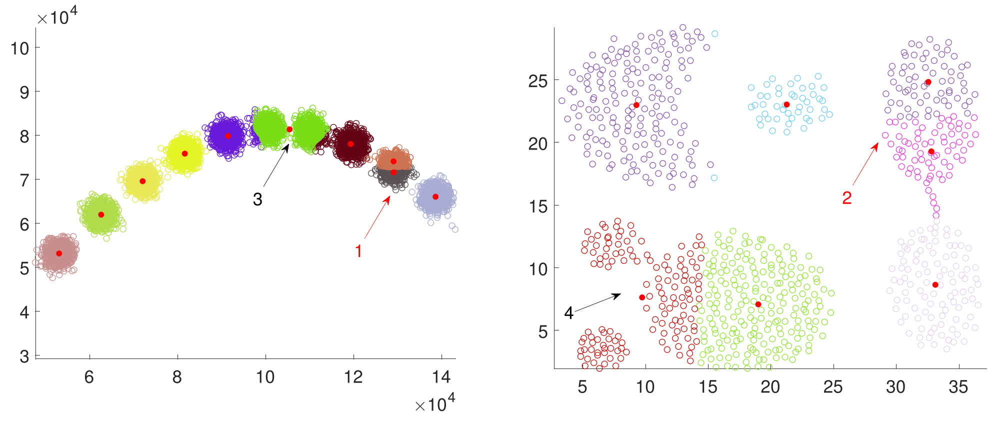
$$c^* = \arg \min_{x \in C_i, |Ns(x)| \geq k} SumDist(x) \tag{5}$$

In order to further reduce f_a, c^* is adjusted when a new node is added or the point minimizing f_a has not been found. Since this step is time-consuming, the search process is restricted to traverse the adjacent points $N(c^*)$ of c^*.

5.2 The proposed objective function

The performance of iterative centroid-based clustering algorithms, which generates a large number of local optimal solutions, is highly related to the initial cluster centers. If the initial cluster centers are not chosen properly, it is difficult to get a desired solution during the procedure. The objective function in these cases only focuses on the sum of the distances between the center and other points, while ignoring the boundary between the clusters. This can lead to dividing a large cluster into two or more small clusters (1,2), and to causing some small clusters to be mistakenly combined into a large cluster (3,4), as shown in Fig. 3. If minimizing the number of adjacent pairs whose elements lie in different clusters is taken as the objective function, this phenomenon can be avoided. This alternative objective can divide clusters that are far apart and make all clusters well bounded.

As previously noted, the newly proposed objective function f_a is the number of adjacent pairs (NAP) among all clusters. The fewer the number of adjacent pairs whose elements lies in different clusters, the more appropriate are the boundaries between the clusters. Consequently, as shown in Fig. 2, the clustering result obtained by f_a is better



(a) K-means on birch2 (b) K-means on Aggregation

Fig. 3 The results obtained by K-means on data sets birch2 and Aggregation

than the one by f_k. Most clustering algorithms based on f_k require that the cluster size is roughly the same across all clusters and the shape of the clusters is approximately round or square, which makes them deeply dependent on the initial solution. By contrast, f_a does not depend on the shape or size of the cluster, so it is insensitive to the initial solution.

5.3 Parameter δ

The parameter δ is a threshold value that determines whether points are adjacent to others. It also reflects the degree of concentration within clusters and the degree of separation between clusters, so it is related to the average distance of points and the number of clusters k as defined in (6).

$$\delta = \frac{\sum_{i=1}^{n}\sum_{j=i}^{n} d(x^i, x^j)}{kn(n-1)/2} \tag{6}$$

6 Multi-start local search based on NAP

We now describe the main scheme and each component of multi-start local search based on NAP. In general, our MLS procedure consists of three phases: initialization, optimization, and adjustment. Since the algorithm is insensitive to the initial solution, a greedy construction algorithm is used in the initialization phase. The optimization phase mainly uses two basic meta-operations called *merge* and *split* to optimize the objective function f_a. The adjustment phase adjusts c^* within its neighborhood $Ns(c^*)$ to make f_a as small as possible. These three phases are invoked repeatedly until the stop condition is met. MLS terminates when the solution has not been improved for $\max_i ter$ consecutive repetitions of the above three phases. The general scheme of the overall clustering procedure is described in Algorithm 1.

```
1: while stop condition is not met do
2:     Initialization()
3:     Optimize()
4:     CenterAdjust()
5: end while
```

Algorithm 1 The main scheme of the multi-start local search algorithm.

6.1 Initialization phase

In the initialization phase, whose pseudo code appears in Algorithm 2 below, an arbitrary seed element in X is used as the first element of c (line 3), and then the point that is the farthest from the chosen center is selected as the next element of c, until c covers k elements (lines 5∼9). Then, remaining elements in X are assigned to the closest center according to the measure $d(x^i, x^j)$ (lines 10∼12). After initial clusters have been constructed, the centers in the set c are updated by minimizing $SumDist(c_i, C_i)$ (line 15) and each element is reassigned to its center (lines 16∼18). This process is repeated until set c does not change or a given stopping criterion is met (lines 14∼25). The initial solution procedure by Algorithm 2 is a locally optimal solution based on prototype clustering.

In the following, when referring to distances, it is convenient to refer to the points x^i and x^j by their indices i and j, writing $d(i, j)$ for $d(x^i, x^j)$, $MinD(i, c)$ for $MinD(x^i, c)$, and writing $i \in C_p$ and $j \in C_q$ for $x^i \in C_p$ and $x^j \in C_q$.

6.2 Optimization phase

The pseudo-code of the optimization phase of our local search algorithm is presented in Algorithm 3. First, it initializes the min_f_a value to be "nfinity" (i.e., a large

```
1: Input: Instance of the considered problem
2: Output: The initial solution with $(C, c)$
3: Initialize an arbitrary point $x \in X$
4: Set $c_1 = x, h = 1, c = \{c_1\}, I_0 = N \setminus \{r\}$
5: for $h = 2$ to $k$ do
6:     $MinD(i, c) = \min(d(i, j) | j \in c)$, for each $i \in I_0$
7:     $c_h = \arg\max_{i \in I_0} MinD(i, c)$;
8:     $c = c \cup \{c_h\}, I_0 = I_0 \setminus \{c_h\}$;
9: end for
10: for each $h \in K$ do
11:     $C_h = \{i \mid d(i, c_h) = MinD(i, c), i \in N\}$;
12: end for
13: Set $pre\_c = c, pre\_C = C$;
14: while (no improved) do
15:     $c_h = \arg\min_{i \in C_h} SumDist(i, C_h)$, for each $h \in K$;
16:     for each $h \in K$ do
17:         $C_h = \{i \mid d(i, c_h) = MinD(i, c), i \in N\}$;
18:     end for
19:     if $pre\_c = c$ then
20:         $break$;
21:     else
22:         $pre\_c = c$;
23:         $pre\_C = C$;
24:     end if
25: end while
```

Algorithm 2 Pseudo code of the *Initialization* procedure.

```
1: Input: An incumbent solution with $(C, c)$
2: Output: An optimized solution with $(C_{best}, c_{best})$
3: Set $\min\_f_a = +\infty$
4: while ( no improved) do
5:     Choose the pair of clusters $(C_p, C_q)$ that has the largest value of $SumAdjs$:
6:     $(C_p, C_q) \leftarrow ChooseMaxPair(C, c)$    /* see Algorithm 4 */
7:     $C_p = C_p \bigcup C_q$; $C = C \setminus C_q$; $K = K \setminus \{q\}$
8:     $c_p = \arg\min\{SumDist(i, C_p) | i \in C_p\}, c = c \setminus \{c_q\}$
9:     for each $h \in K$ do
10:         $C_h = \{i \mid d(i, c_h) = MinD(i, c), i \in N\}$
11:     end for
12:     Set $f_2 = +\infty, add\_i = -1$
13:     for $i = 1$ to $k - 1$ do
14:         Divide $C_i$ into $C_{i,1}$ and $C_{i,2}$
15:         $cur\_f_2 = SumAdjs(C_{i,1}, C_{i,2})$
16:         if $(cur\_f_2 < f_2)$ then
17:             $f_2 = cur\_f_2, add_i = i$
18:         end if
19:     end for
20:     Divide $C_{add_i}$ into $C_{i,1}$ and $C_{i,2}$
21:     $f_a = \sum_{i \neq j} SumAdjs(C_i, C_j), i, j \in K$
22:     if $(f_a < \min\_f_a)$ then
23:         $\min\_f_a = f_a$
24:         $C_{best} = C, c_{best} = c$
25:     end if
26: end while
```

Algorithm 3 Pseudo code of the *Optimize* procedure.

value) in order to record the best solution (line 3) and then starts from the initial solution generated by Algorithm 2.

The local search based on the operations *merge* and *split* is designed to optimize the objective function f_a in the optimization phase. The *merge* operation merges two clusters with the maximum NAP value into one cluster, while the *split* operation selects the cluster with the smallest NAP value to divide into two clusters. Upon starting from an initial solution generated by Algorithm 2, the algorithm repeats the merge operation (lines 6∼11) and the split operation (lines 13∼25) until reaching a given iteration limit. After the merge operation, a new center for the merged cluster is chosen (line 8). Then all elements are reassigned to produce $k - 1$ clusters by the measure $d(x^i, x^j)$ (lines 9∼11). The split operation then identifies a tentative new center for each cluster, whereon the algorithm computes the NAP after splitting the cluster and records the minimum NAP and the corresponding cluster after the split operation in f_2 and add_i (lines 12∼19). After all the clusters have been checked, the cluster C_{add_i} is divided into two clusters (line 20), and the solution is updated according to f_a (lines 21∼25). The procedure terminates when the solution has not been improved for $I_{\max}$ consecutive iterations.

The procedure of choosing the pair of clusters (C_p, C_q) that has the largest value of $SumAdjs$ in Algorithm 3 is detailed in Algorithm 4. For this propose, we first use preprocessing for each point $i = 1$ to n, to create the sets $S(i) = \{j > j|d(i, j) < \delta\}$. Then, while generating and updating each cluster C_r, we keep track of its membership by recording $Cluster(i) = r$ for each element i in the cluster C_r (lines 3∼6). After that, we examine the sets $S(i)$ for $i = 1$ to $n - 1$, and for each j in $S(i)$ set $V(r, s) = V(r, s) + 1$ for $r = Cluster(i)$ and $s = Cluster(j)$ (lines 12∼18). Finally, the pair of clusters (C_p, C_q) that has the largest value of $SumAdjs$ can be identified simply as $V(r, s)$ for $r = 1$ to $k - 1$ and $s = r + 1$ to k (line 24).

6.3 Adjustment phase

After the inter-cluster tuning in the optimization phase, intra-cluster requires readjustment to further improve f_a. This is accomplished in Algorithm 5 by sequentially adjusting each center and exchanging it with one of its

```
1: Input: An incumbent solution with $(C, c)$
2: Output: a pair of clusters $(C_p, C_q)$ that has the largest value of $SumAdjs$
3: for $i = 1$ to $n$ do
4:     $S_i = \{j > i \mid d(i, j) < \delta, i, j \in X\}$
5:     $Cluster(i) = r, i \in C_r$ /* update the membership accordingly */
6: end for
7: for $r = 1$ to $k$ do
8:     for $s = 1$ to $k$ do
9:         $V(r, s) = 0$
10:    end for
11: end for
12: for $i = 1$ to $n - 1$ do
13:    for each $j \in S_i$ do
14:        Identify the corresponding clusters:
15:        $r = Cluster(i), s = Cluster(j)$
16:        $V(r, s) = V(r, s) + 1$
17:    end for
18: end for
19: for $r = 1$ to $k - 1$ do
20:    for $s = r + 1$ to $k$ do
21:        $SumAdjs(r, s) = V(r, s) + V(s, r)$
22:    end for
23: end for
24: $(C_p, C_q) = \arg\max\{SumAdjs(C_r, C_s) | r \in \{1, \ldots, k - 1\}, s = r + 1\}$
```

Algorithm 4 Pseudo code of the *ChooseMaxPair* procedure.

neighbors $Ns(c^*)$ to obtain a better center (line 6). Then, each point is reassigned to the closest center by the measure $d(i, j)$ (line 7) followed by recording the minimum f_a (lines 8∼11).

```
1: Input: An incumbent solution with $(C, c)$
2: Output: An adjusted solution with $(C_{best}, c_{best})$
3: for $h = 1$ to $k$ do
4:     $odd\_center = c_h$
5:     for each $t \in Ns(old\_center)$: do
6:         $c_h = t$
7:         Reassign each point according to $d(i, j)$
8:         $f_a = \sum_{i \neq j} SumAdjs(C_i, C_j), i, j \in K$
9:         if $(f_a < \min\_f_a)$ then
10:            $\min\_f_a = f_a, C_{best} = C, c_{best} = c$
11:        end if
12:    end for
13: end for
```

Algorithm 5 Pesudo code of *CenterAdjust* procedure.

7 Experimental results

We report the outcomes to evaluate the performance of the MLS algorithm on a variety of clustering benchmarks which are widely used in the literature [22]. The clustering performance takes three aspects into account: the Jaccard Coefficient (JC), the Fowlkes and Mallows Index (FMI) and Accuracy, which can eliminate the influence of the measurement standard on the results and make the outcome more conclusive. For visualization to compare with other clustering methods, most examples will be displayed in 2D space, using Euclidean distance as the distance measure.

7.1 Benchmark sets

The benchmark sets of our study consist of the following types.

A-sets [26]: the data sets contain a number of spherical clusters ($k = 20, 35, 50$). The number of points and their proximity to each other increases with the number of clusters. As shown in Fig. 4, our algorithm obtains convincing results in spite of the increasing number of clusters, because its performance is rarely affected by the initial solution. The clusters marked with red circles on the right could be either combined into one cluster or divided into two clusters. With the increasing number of clusters, it is difficult to obtain a good initial solution.

Aggregation-sets [75]: These data sets produce slightly unbalanced cluster sizes and their shapes are a little different from those obtained from the circular data set. When using f_k as the objective function as in the standard K-means algorithm, the final result is not ideal no matter how many times the algorithm is restarted, as show in Fig. 2. However, when our f_a objective is used, a good clustering effect can be easily obtained, as is shown in Fig. 5.

Birch-sets [43]: These data sets consist of two types, birch1 and birch2, which are extracted separately from an original data set consisting of 100,000 points with 100 clusters. The number of clusters in the birch1 data set is very large ($k = 100$), compared with the example introduced above. In contrast, birch2 has a very large number of points per cluster. We tested birch1 and birch2 with sum of the squared errors (SSE) and NAP as the objective function, and found that SSE has difficulty converging to the global optimum unless the associated algorithm is restarted thousands of times. Because the algorithm could not jump out of a local optimum and could only rely on successive restarting, the quality of the final solution was rendered very uncertain. In spite of the challenge presented by the data, our algorithm using the NAP-based objective only required a few iterations to converge, again demonstrating little dependence on the initial solution. The results of MLS are plotted in Fig. 6.

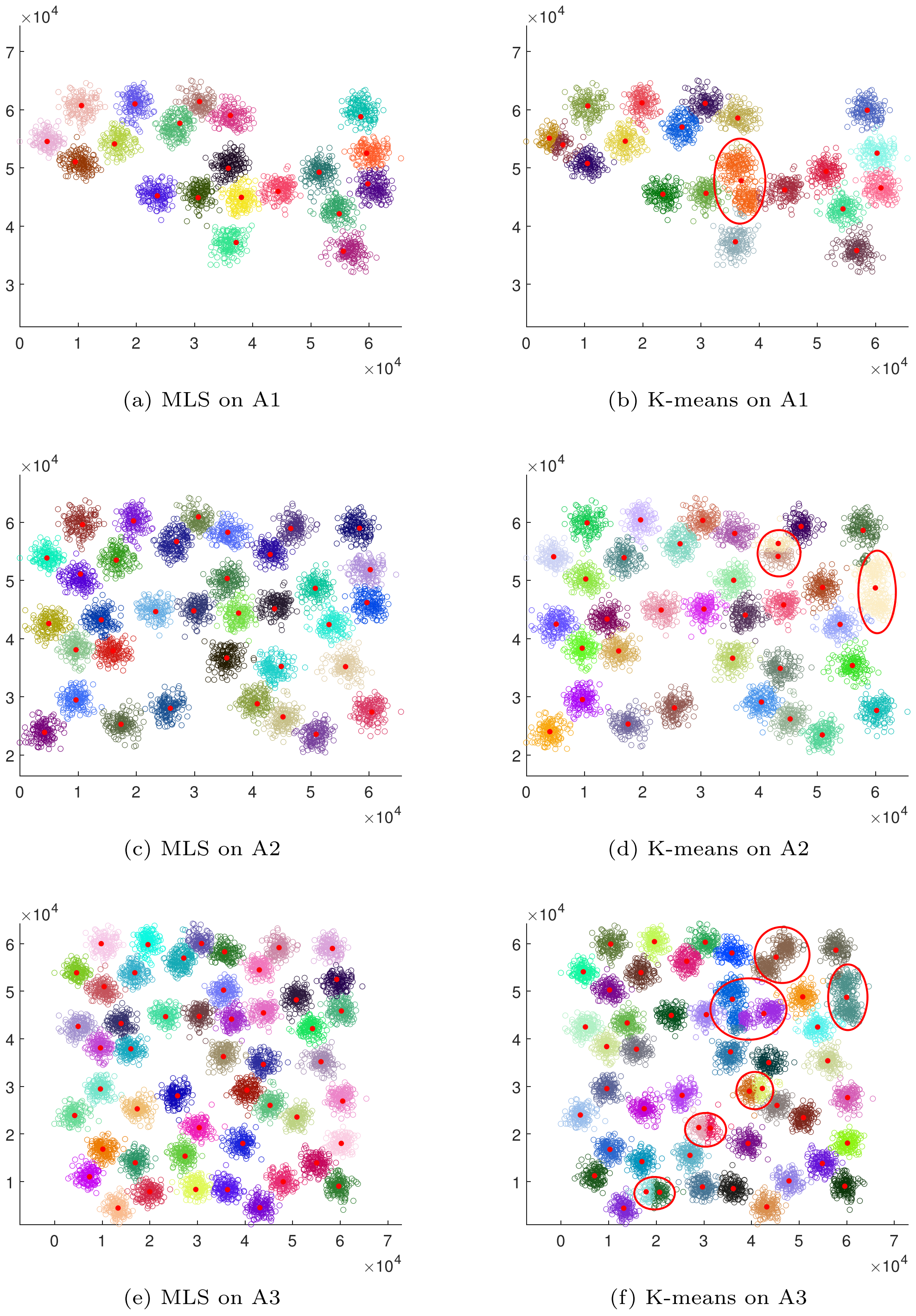


Fig. 4 The results obtained by MLS (on the right) and K-means (on the left) on instances A1, A2, and A3 in A-sets

Unbalance-sets [58]: This data set gives rise to 8 strongly unbalanced clusters. The three clusters on the left in Fig. 7 are very dense because each of them contains 2000 points. The other five clusters on the right are relatively sparse and each of them contains 100 points. If the initial centers are not uniformly distributed among the best clusters, the K-means approach will only converge to a local optimal solution, yielding different clusters from that shown in Fig. 6. However, owing to the NAP-based objective function and the center adjustment phase, MLS is able to obtain a highly separable solution for the unbalanced cluster data set and succeeds in identifying the clusters plotted in Fig. 7.

S-Sets [57]: The S-sets contain Gaussian clusters with different overlaps. The majority of them are spherical, but a small part of them have been transformed to be likely to the non-spherical Gaussian clusters. The second set (S2) has strong overlapping points but the general distribution of the clusters are still visible. The visualization is shown

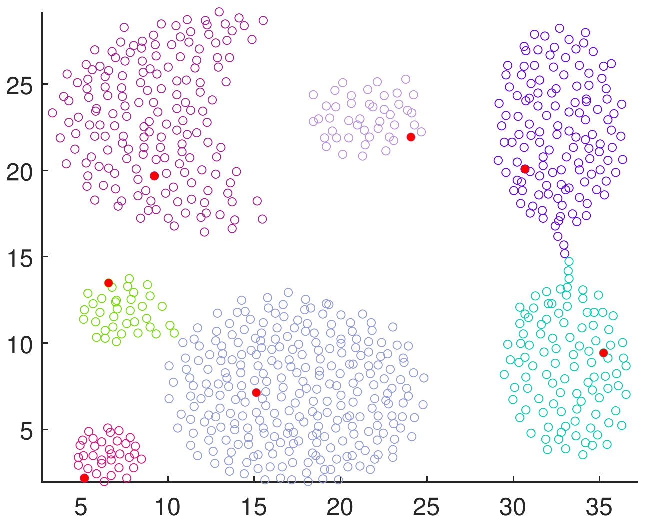


Fig. 5 The results obtained by MLS on data set Aggregation($N = 788$, $k = 7$)

in Fig. 8. R15 and D31-set [34]: These data sets contain varying numbers of spherical clusters ($k = 15, k = 31$). The visualization of the results obtained by MLS on data sets R15 and D31 are shown in Fig. 9.

7.2 Performance metrics

Clustering partitions the sample set into several non-intersecting subsets. In this paper, the clustering results are compared with a "reference model". For example, the segmentation results given by domain experts are used as a reference model.

As above we represent the data set by $X = \{x^i \mid i \in N\}$ and the collection of clusters by $C = \{C_h \mid h \in K\}$. The cluster collection generated by the reference model will be denoted as $C* = \{C_h^* \mid h \in K\}$. Let λ and λ^* refer to the points in C and C^*, respectively, we define:

$$a =| SS |, SS = \{(x_i, x_j) \mid \lambda_i = \lambda_j, \lambda_i^* = \lambda_j^*, i < j\}$$

$$b =| SD |, SD = \{(x_i, x_j) \mid \lambda_i = \lambda_j, \lambda_i^* \neq \lambda_j^*, i < j\}$$

$$c =| DS |, DS = \{(x_i, x_j) \mid \lambda_i \neq \lambda_j, \lambda_i^* = \lambda_j^*, i < j\}$$

The interpretation of these definitions is as follows. The set SS contains sample pairs that are both in the same cluster in C and C^*. The set SD contains sample pairs that are in the same cluster in C but in different clusters in C^*. The set DS contains sample pairs that are in different clusters in C but in the same cluster in C^*. We define three performance metrics based on a, b, and c.

- Jaccard Coefficient(JC):

$$JC = \frac{a}{a+b+c}$$

- Fowlkes and Mallows Index(FMI):

$$FMI = \sqrt{\frac{a}{a+b} * \frac{a}{a+c}}$$

- Accuracy:

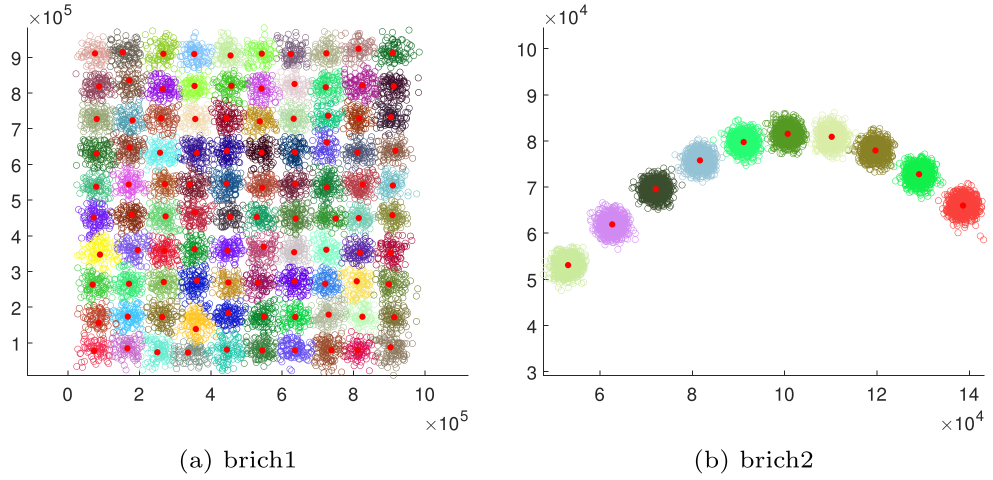
$$accuracy = \frac{a}{a+c}$$

The above performance measurement values all lie in the interval [0, 1], and the larger the values, the better the results.

7.3 Comparison with traditional K-means variants

As previous tested by K-means* and K-means++, the data sets A1, A2, A3, Aggregation, Birch, R15, D31 and Iris are also used as the testbed of MLS for comparison. The difference among K-means, K-means* and K-means++ lies in the construction of the initial solution. K-means randomly selects k points as the initial solution, and K-means*

Fig. 6 The results obtained by MLS on data set birch1 ($N = 10000$, $k = 100$) and birch2 ($N = 10000$, $k = 10$)



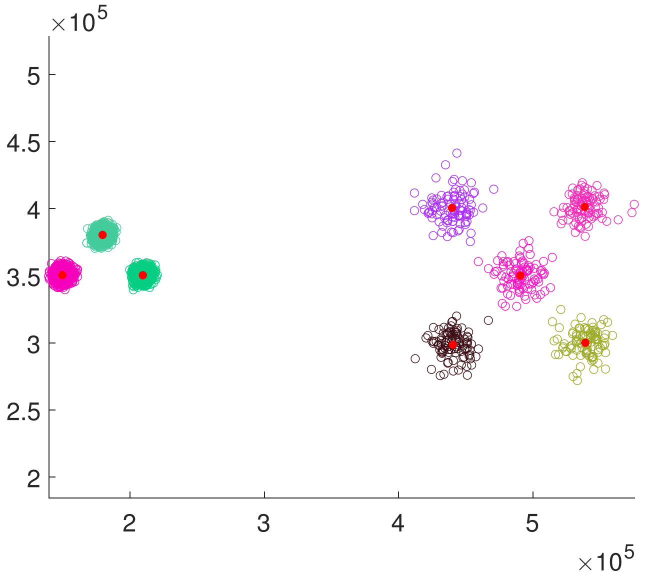


Fig. 7 The results obtained by MLS on data set unbalance-sets($N = 6500, k = 8$)

chooses the first center c_1 arbitrarily and the following i-th ($i \in \{2, 3, \dots, K\}$) center c_i to be the point that has the largest minimum distance to the previously selected centers, i.e., $c_1, c_2, \dots, c_{i-1}$. K-means++ randomly selects one point as the first center and chooses the following i-th ($i \in \{2, 3, \dots, k\}$) center to be $x \in X$ with a probability of $\frac{md(x)^2}{\sum_{j=1}^{n} md(x_j)^2}$, where $md(x)$ denotes the minimum distance from point x to the previous selected centers.

Next, we evaluate the proposed MLS algorithm in terms of both solution quality and computational efficiency. To evaluate its performance more precisely, we apply the following two kinds of experiments on all the common instances for the four algorithms: MLS, K-means, K-means*, and K-means++.

First, we run each of the four algorithms once and record the corresponding results, comparing their performance with respect to the three criteria defined above. The comparison results are plotted in Fig. 10 on the left, where the x-axis represents different data sets and the y-axis represents the value of the performance evaluation criteria. It can be seen from Fig. 10a, b, and c that the JC, FMI, and Accuracy values obtained by MLS are larger than those values obtained by K-means, K-means*, and K-means++, which indicates that MLS is superior to other three algorithms in terms of JC, FMI, and Accuracy.

Second, we execute MLS and the reference algorithms for 100 times and record the best solutions found by each algorithm and the corresponding computation time. The comparison results are plotted in Fig. 10. Supplementing these plots, Table 2 presents the average computational time in seconds to reach the minimum value for the employed objective. From Fig. 10d, e, f, and Table 2, one observes that MLS achieves the best of JC, FMI, and Accuracy with the smallest computation time, demonstrating that MLS outperforms the three versions of K-means algorithms in terms of both solution quality and computational efficiency.

7.4 Comparison with K-means variants based on metaheurstics

Moreover, to further evaluate the performance of MLS, we apply experiments with the recent K-means methods enhanced by metaheuristics: a genetic algorirthm (GA) [9], two firefly algorithms, namely inward intensified exploration FA (IIEFA) and compound intensified exploration

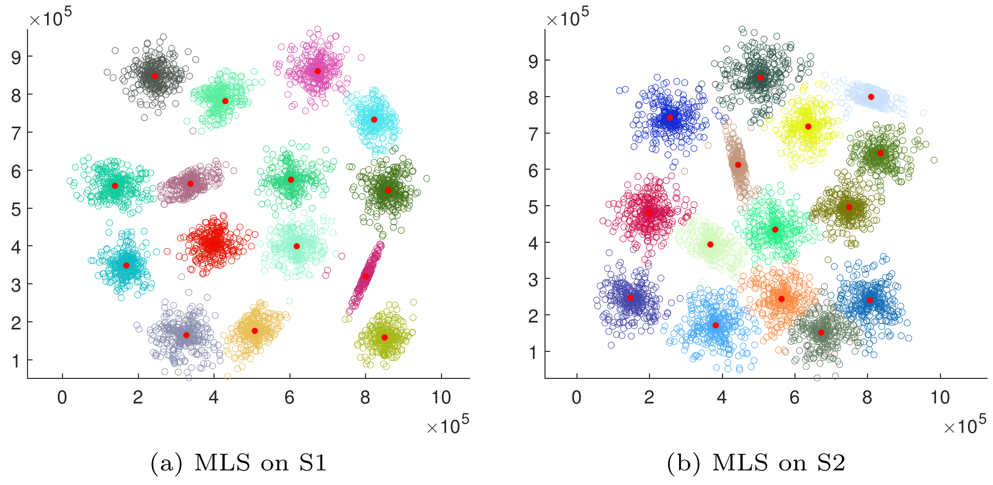


Fig. 8 The results obtained by MLS on data sets S1 ans S2 ($N = 5000, k = 15$)

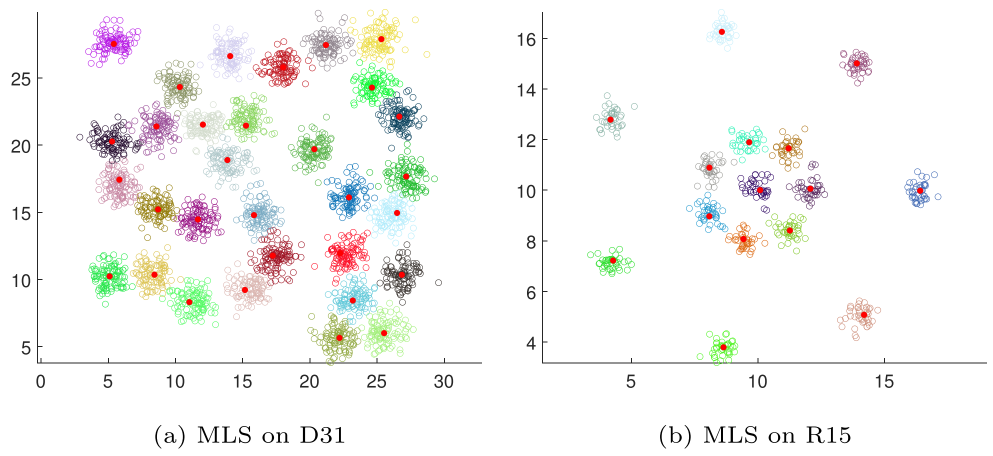


Fig. 9 The results obtained by MLS on data sets R15 ($N = 600$, $k = 15$) and R31 ($N = 3100$, $k = 31$)

FA (CIEFA) [65], and a bargaining game based K-means (GBK-means) [42].

Since GA only reports the Davies–Bouldin index (DBI) as the performance metric, which is an internal evaluation scheme introduced in [15]. A smaller DBI value indicates a relative superiority of the method. Therefore, we compare MLS with GA regarding to DBI values. Table 1 reports the DBI values obtained by MLS, K-means, K-means*, and GA on the 10 UCI machine learning benchmark data sets which are commonly used in the literature [1]. One observes that MLS outperforms the other three algorithms on all the instances and achieves the minimum average value 0.874 for DBI value. This demonstrates the effectiveness of the proposed MLS algorithm (Table 2).

Table 3 reports the accuracy values obtained by IIEFA, CIEFA, GBK-means, and MLS on different data sets. One observes that MLS improves the accuracy values for 4 out of 7 data sets compared with the two firefly algorithms, and IIEFA and CIEFA only outperforms MLS on 3 data sets. Besides, MLS improves the accuracy values for 6 out of 12 data sets compared with GBK-means. These results demonstrate that MLS is comparable to the recent K-means variants based on metaheurstics.

7.5 Comparison with other state-of-the-art clustering algorithms

To further identify the effectiveness of MLS, we compare it with the recent state-of-the-art clustering algorithm based on other mechanisms such as spectral, geometric, and density. The comparison results are presented in Table 4 which reports the accuracy, NMI (normalized mutual information), and F1-score [63] obtained by MLS and the reference algorithms. The larger the NMI or F1-score is, the better the algorithm's performance is. Note that "-" represents that the corresponding values are not available. The reference algorithms are as follows:

- AFS: A spectral clustering method with semantic interpretation based on axiomatic fuzzy set theory [63].
- KNSC: A spectral clustering algorithm based on refining a k-nearest neighbor graph [6].
- GLCDC: A spectral clustering algorithm using density-sensitive distance measure with global and local consistencies [55].
- ADBC: A novel density-based clustering algorithm using nearest neighbor graph [33].
- GDPC: An improved density peaks clustering algorithm with fast finding cluster centers [69].
- FSDPC: A fast density peaks clustering algorithm with sparse search [70].
- ACSC: An ant colony stream clustering based on density for dynamic data streams [18].
- GBC: A geometric-based clustering method using natural neighbors [39].

Table 4 reports the comparison results of MLS and the recent state-of-the-art clustering algorithms based on spectral, density, and geometric. One observes that MLS obtains the best accuracy, NMI, and F1 score for 12, 6, and 11 out of all the 15 data sets, respectively. In particular, MLS outperforms the reference algorithms regarding to all the metrics on four data sets, i.e., wdbc, bre-wi, waveform, and gamma. These results again identify the effectiveness of the proposed MLS method.

8 Conclusion

Clustering is a fundamental technique in machine learning that involves classifying data points by partitioning them into separate groups. In view of the disadvantages of traditional K-means clustering algorithms, which encounter several shortcomings: (i) centroid initialization sensitivity

[1] https://archive.ics.uci.edu/ml/datasets.php.

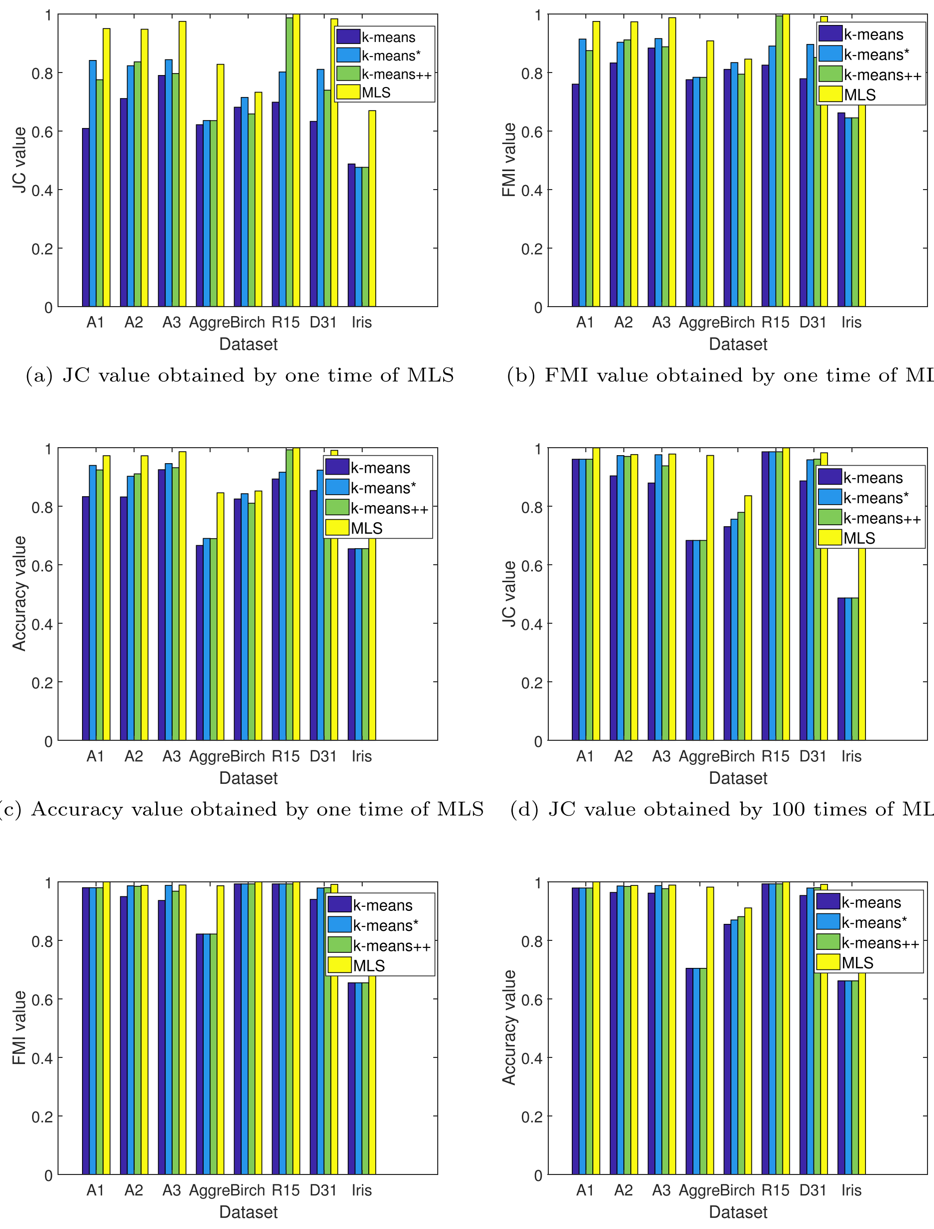


Fig. 10 The JC, FMI, and Accuracy values obtained by MLS (run once on the left and 100 times on the right

(a) JC value obtained by one time of MLS (b) FMI value obtained by one time of MLS

(c) Accuracy value obtained by one time of MLS (d) JC value obtained by 100 times of MLS

(e) FMI value obtained by 100 times of MLS (f) Accuracy value obtained by 100 times of MLS

Table 1 The comparison among MLS and the reference algorithms with respect to the DBI metric

Data set	K-means	K-means*	GA	MLS
Iris	0.78	0.76	0.465	0.435
Heart failure	2.08	1.989	1.402	1.257
Seeds	1.033	0.952	0.661	0.549
Thyroid	0.923	0.682	0.182	0.181
Wine	1.366	1.278	1.179	1.133
Yeast	2.227	1.661	1.191	1.624
Breast	0.877	0.761	0.827	0.768
Glass	1.568	1.292	1.165	1.158
wdbc	0.897	0.869	0.82	0.761
Average	1.306	1.138	0.877	0.874

Table 2 The average computational time required by each algorithm to obtain the minimum objective function value

Data set	K-means	K-means*	K-means++	MLS
A1	3628	1587	4717	1556
A2	100554	35785	44771	2611
A3	280588	94591	115829	6968
Aggregation	4119	720	1109	781
Birch	126715	266099	192214	45195
R15	681	114	532	35
D31	81874	24233	21453	1414
Iris	297	94	161	37
Average	74807.00	52902.88	47598.25	7324.63

occurs due to the random allocation of k centroids. (ii) the stagnation in local minima appears in metaheuristics, and (iii) the improper criteria for evaluating cluster membership. To address these issues, we introduce a new objective function of minimizing the number of adjacent pairs between clusters, and propose a multi-start local search method which incorporates merge and spilt meta-operations. The experimental results show that the proposed method is superior to commonly used clustering algorithms in the K-means class, evaluated by standard performance metrics.

The main contributions of this paper consist of four elements: the selection of the cluster centers, the objective function definition, the method of improving a current cluster configuration, and the performance analysis of experimental results. First, the center of each cluster is no longer its centroid, which reduces the dependence of the algorithm's performance on data points that can be grouped in roughly spherical structures and whose densities do not vary widely in the solution space. Second, we introduce a characterization of adjacency that underlies our new objective function to minimize the number of adjacent pairs whose elements lie in different clusters. Third, we use two basic meta-operations denoted merge and split to successively improve the clusters and make the iterative process insensitive to the initial solution. Finally, we apply three performance evaluation criteria to conduct a computational analysis of our procedure. The experimental results demonstrate that our MLS algorithm can provide much better classification accuracy than traditional K-means based clustering algorithms, and it is quite competitive to others such as spectral, density, and geometric based clustering algorithms.

This research can be extended in several directions. First, since the new objective function has little correlation with parameter k, the problem of specifying a good k value may be handled effectively by a well-designed dynamic self-

Table 3 Comparison of MLS with IIEFA, CIEFA, and GBK-means

Data set	IIEFA	CIEFA	MLS	data set	GBK-means	MLS
Iris	0.88	0.87	**0.96**	s1_real	1	0.99
wdbc[1]	0.91	0.91	**0.93**	s2_real	**1**	0.97
Wine	0.95	**0.97**	0.92	s3_real	**0.99**	0.86
Sonar	0.56	0.56	**0.62**	wdbc	**0.93**	**0.93**
Ecoli	0.77	0.79	**0.80**	ionosphere	0.58	**0.71**
Balance[2]	**0.80**	0.79	0.67	heart	**0.77**	0.79
Thyroid0387	0.82	**0.83**	0.40	australian	**0.86**	0.79
				haberman	0.52	**0.75**
				bre-ca[3]	**0.93**	0.70
				bre-cw[4]	0.93	**0.96**
				mam_mas[5]	0.75	**0.78**
				jap_cr[6]	**0.84**	0.79

The largest value of each row is marked in bold

1: wdbc is shorted for wdbc_breast_cancer. 2: balance is shorted for balance-scale

3: bre-ca is shorted for breast-cancer. 4: bre-cw is shorted for breast-cancer-wisconsin

5: mam-mas is shorted for mammographic_masses. 6: jap-cre is shorted for japanese_credit

Table 4 Comparison of MLS with other clustering algorithms based spectral and density

Data set	Metric	Ours	Spectral based			Density based				Geometric
		MLS	AFS	KNSC	GLCDC	ADBS	GDPC	FSDPC	ACSC	GBC
Iris	accu	**0.96**	–	0.90	–	–	0.94	0.94	0.89	–
	NMI	0.86	**0.90**	–	0.86	0.74	–	–	–	0.81
	F1	**0.96**	0.95	–	–	–	–	–	–	0.95
wdbc	accu	**0.93**	–	–	–	–	–	–	–	–
	NMI	**0.63**	0.48	–	–	–	–	–	–	–
	F1	**0.93**	0.78	–	–	–	y	–	–	–
Wine	accu	**0.92**	–	0.70	–	y	–	–	0.88	–
	NMI	0.76	0.76	–	**0.88**	0.63	–	–	–	–
	F1	**0.92**	0.87	–	–	y	–	–	–	–
Sonar	accu	**0.62**	–	–	–	–	–	–	–	–
	NMI	0.04	0.08	–	**0.30**	–	–	–	–	–
	F1	**0.61**	0.55	–	–	–	–	–	–	–
Iono[1]	accu	**0.71**	–	–	–	–	–	–	–	–
	NMI	0.13	–	–	**0.32**	–	–	–	–	–
	F1	**0.70**	–	–	–	–	–	–	–	–
Ecoli	accu	**0.80**	–	–	–	–	–	–	–	–
	NMI	0.68	–	0.58	–	–	–	–	–	**0.70**
	F1	0.54	**0.79**	–	–	–	–	–	–	0.79
Heart	accu	**0.79**	–	–	–	–	–	–	–	–
	NMI	0.26	–	–	**0.40**	–	–	–	–	–
	F1	**0.79**	–	–	–	–	–	–	–	–
Habe[2]	accu	**0.75**	–	–	–	–	–	–	–	–
	NMI	0.08	–	–	**0.27**	–	–	–	–	–
	F1	**0.65**	–	–	–	–	–	–	–	–
Bre-ca	accu	**0.70**	–	–	–	–	–	–	–	–
	NMI	0.03	–	–	0.79	–	–	–	–	**0.83**
	F1	0.60	–	–	–	–	–	–	–	**0.93**
Bre-wi	accu	**0.96**	–	–	–	–	–	–	–	–
	NMI	**0.76**	**0.76**	–	–	–	–	–	–	–
	F1	**0.96**	0.93	–	–	–	–	–	–	–
Seeds	accu	0.85	–	–	–	–	0.88	**0.90**	–	–
	NMI	0.64	**0.74**	–	–	0.66	–	–	–	0.69
	F1	0.53	0.84	–	–	–	–	–	–	**0.89**
Segm[3]	accu	0.62	–	**0.65**	–	–	–	0.6	–	–
	NMI	**0.70**	0.59	–	–	–	–	–	–	–
	F1	0.50	**0.57**	–	–	–	–	–	–	–
Waveform	accu	**0.78**	–	–	–	–	0.62	0.54	–	–
	NMI	**0.47**	–	–	–	–	–	–	–	–
	F1	**0.77**	–	–	–	–	–	–	–	–
Gamma	accu	**0.69**	–	0.66	–	–	0.51	0.65	–	–
	NMI	**0.08**	–	–	–	–	–	–	–	–
	F1	**0.66**	–	–	–	–	–	–	–	–
Pendigits	accu	0.74	–	**0.83**	–	–	0.37	0.65	–	–
	NMI	**0.68**	–	–	–	–	–	–	–	–
	F1	**0.71**	–	–	–	–	–	–	–	–

The largest value of each row is marked in bold

1: iono is shorted for ionosphere. 2: habe is shorted for haberman

3: segm is shorted for segmetation. Other abbreviations are the same as in Table 3 adjustment strategy. Second, the local search algorithm in this paper could be combined with reinforcement learning technique to enhance its efficiency. Third, given the merits of the new objective function of minimizing the number of adjacent pairs, other objective functions with respect to both inter- and intra-cluster measurements will be adopted to tackle more complex and irregular data distribution problems in future study.

Acknowledgements We are grateful to one of our team members Tuanyue Xiao, who has contributed substantially to the revisions, especially the additional experiments for validation, comparisons with other clustering algorithms, and analysis of computational results. This work was supported by the Special Project for Knowledge Innovation of Hubei Province under Grant No. 2022013301015175 and the National Natural Science Foundation of China under Grant No. 62202192.

Author Contributions Xiaolu Liu contributed to conceptualization, validation, writing and editing, supervision, project administration. Wenhan Shao contributed to data analysis, investigation, and writing-original draft preparation; Jiaming Chen contributed with conceptualization, validation, investigation, visualization, writing original draft and editing. Jiaming Chen contributed to revisions, additional experiments for validation, drawing flowchart of the main algorithm, and data analysis. Zhipeng Lü contributed to conceptualization, validation, writing-draft review and editing, supervision, project administration, funding acquisition. Fred Glover contributed to conceptualization, methodology, writing-review and editing, project administration. Junwen Ding contributed to conceptualization, methodology, validation, writing-review and editing, formal analysis, supervision, and funding acquisition.

Data Availability The datasets adopted in the current study are available at https://archive.ics.uci.edu/ml/datasets.php.

Declarations

Ethics approval and consent to participate This article does not contain any studies with human participants or animals performed by any of the authors.

Conflict of Interests The authors declare that they have no conflict of interest.

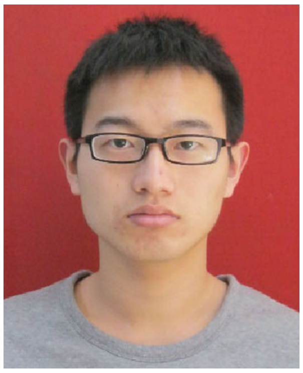
References

1. Abualigah LM, Khader AT, Hanandeh ES (2018) Hybrid clustering analysis using improved krill herd algorithm. Appl Intell 48(11):4047–4071
2. Agrawal R, Gehrke J, Gunopulos D et al (2005) Automatic subspace clustering of high dimensional data. Data Min Knowl Disc 11(1):5–33
3. Aljalbout E, Golkov V, Siddiqui Y et al (2018) Clustering with deep learning: taxonomy and new methods. arXiv:180107648
4. Aljarah I, Mafarja M, Heidari AA et al (2020) Clustering analysis using a novel locality-informed grey wolf-inspired clustering approach. Knowl Inf Syst 62(2):507–539
5. Aloise D, Deshpande A, Hansen P et al (2009) NP-Hardness of euclidean sum-of-squares clustering. Mach Learn 75(2):245–248
6. Alshammari M, Stavrakakis J, Takatsuka M (2021) Refining a *k*-nearest neighbor graph for a computationally efficient spectral clustering. Pattern Recog 114:107,869
7. Amini A, Wah TY, Saybani MR et al (2011) A study of density-grid based clustering algorithms on data streams. In: 2011 Eighth international conference on fuzzy systems and knowledge discovery (FSKD), IEEE, pp 1652–1656
8. Ankerst M, Breunig MM, Kriegel HP et al (1999) Optics: ordering points to identify the clustering structure. ACM Sigmod Record 28(2):49–60
9. Ashraf FB, Matin A, Shafi MSR et al (2021) An improved k-means clustering algorithm for multi-dimensional multi-cluster data using meta-heuristics. In: 2021 24th International conference on computer and information technology (ICCIT), IEEE, pp 1-6
10. Bateni MH, Behnezhad S, Derakhshan M et al (2017) Affinity clustering: hierarchical clustering at scale. In: Proceedings of the 31st International conference on neural information processing systems, pp 6867–6877
11. Bendoly E (2003) Theory and support for process frameworks of knowledge discovery and data mining from ERP systems. Inf Manag 40(7):639–647
12. Brown D, Japa A, Shi Y (2019) A fast density-grid based clustering method. In: 2019 IEEE 9Th annual computing and communication workshop and conference (CCWC), IEEE, pp 0048–0054
13. Cao B, Glover F, Rego C (2015) A tabu search algorithm for cohesive clustering problems. J Heuristics 21(4):457–477
14. Celebi ME, Kingravi HA, Vela PA (2013) A comparative study of efficient initialization methods for the K-means clustering algorithm. Expert Syst Appl 40(1):200–210
15. Davies DL, Bouldin DW (1979) A cluster separation measure. IEEE Trans Pattern Anal Mach Intell 3(2):224–227
16. Duwairi R, Abu-Rahmeh M (2015) A novel approach for initializing the spherical K-means clustering algorithm. Simul Model Pract Theory 54:49–63
17. Ester M, Kriegel HP, Sander J et al (1996) A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceedings of the AAAI conference on artificial intelligence, pp 226–231
18. Fahy C, Yang S, Gongora M (2018) Ant colony stream clustering: a fast density clustering algorithm for dynamic data streams. IEEE Trans Cybern 49(6):2215–2228
19. Fan J (2019) Ope-hca: an optimal probabilistic estimation approach for hierarchical clustering algorithm. Neural Comput Applic 31(7):2095–2105
20. Fränti P, Sieranoja S (2018) K-means properties on six clustering benchmark datasets. Appl Intell 48(12):4743–4759
21. Frey BJ, Dueck D (2007) Clustering by passing messages between data points. Science 315(5814):972–976
22. Gionis A, Mannila H, Tsaparas P (2007) Clustering aggregation. ACM Trans Knowl Discov Data 1(1):4–14
23. Glover F (2017) Pseudo-centroid clustering. Soft Comput 21(22):6571–6592
24. Guha S, Rastogi R, Shim K (1998) Cure: an efficient clustering algorithm for large databases. ACM Sigmod Record 27(2):73–84
25. Guo X, Gao L, Liu X et al (2017) Improved deep embedded clustering with local structure preservation. In: The 26th International joint conference on artificial intelligence (IJCAI), pp 1753–1759
26. Hartigan JA, Wong MA (1979) Algorithm as 136: a K-means clustering algorithm. J R Stat Soc Ser C (Appl Stat) 28(1):100–108
27. Huang D, Wang CD, Lai JH (2017) Locally weighted ensemble clustering. IEEE Trans Cybern 48(5):1460–1473
28. Huang D, Wang CD, Wu JS et al (2019) Ultra-scalable spectral clustering and ensemble clustering. IEEE Trans Knowl Data Eng 32(6):1212–1226

29. Jain AK (2010) Data clustering: 50 years beyond K-means. Pattern Recogn Lett 31(8):651–666
30. Kabsch W, Sander C (1983) Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. Biopolymers: Original Res Biomol 22(12):2577–2637
31. Kanungo T, Mount DM, Netanyahu NS et al (2002) An efficient K-means clustering algorithm: analysis and implementation. IEEE Trans Pattern Anal Mach Intell 24(7):881–892
32. Langan DA, Modestino JW, Zhang J (1998) Cluster validation for unsupervised stochastic model-based image segmentation. IEEE Trans Image Process 7(2):180–195
33. Li H, Liu X, Li T et al (2020) A novel density-based clustering algorithm using nearest neighbor graph. Pattern Recog 102:107,206
34. Likas A, Vlassis N, Verbeek JJ (2003) The global K-means clustering algorithm. Pattern Recogn 36(2):451–461
35. Liu R, Wang H, Yu X (2018) Shared-nearest-neighbor-based clustering by fast search and find of density peaks. Inf Sci 450:200–226
36. Lloyd S (1982) Least squares quantization in PCM. IEEE Trans Inf Theory 28(2):129–137
37. Meyerhenke H, Sanders P, Schulz C (2016) Partitioning (hierarchically clustered) complex networks via size-constrained graph clustering. J Heuristics 22(5):759–782
38. Peng X, Zhu H, Feng J et al (2019) Deep clustering with sample-assignment invariance prior. IEEE Trans Neural Netw Learn Syst 31(11):4857–4868
39. Pourbahrami S, Hashemzadeh M (2022) A geometric-based clustering method using natural neighbors. Inf Sci 610:694–706
40. Ran X, Zhou X, Lei M et al (2021) A novel K-means clustering algorithm with a noise algorithm for capturing urban hotspots. Appl Sci 11(23):11,202
41. Rappoport N, Shamir R (2018) Multi-omic and multi-view clustering algorithms: review and cancer benchmark. Nucleic Acids Res 46(20):10,546–10,562
42. Rezaee MJ, Eshkevari M, Saberi M et al (2021) GBK-Means clustering algorithm: an improvement to the K-means algorithm based on the bargaining game. Knowl-Based Syst 213:106,672
43. Rezaei M, Fränti P (2016) Set matching measures for external cluster validity. IEEE Trans Knowl Data Eng 28(8):2173–2186
44. Rodriguez A, Laio A (2014) Clustering by fast search and find of density peaks. Science 344(6191):1492–1496
45. Ruspini EH, Bezdek JC, Keller JM (2019) Fuzzy clustering: a historical perspective. IEEE Comput Intell Mag 14(1):45–55
46. Sabin M, Gray R (1986) Global convergence and empirical consistency of the generalized Lloyd algorithm. IEEE Trans Inf Theory 32(2):148–155
47. Sedluk MJ, Miller JW (2000) Cluster-based data compression system and method. US Patent 6,100:825
48. Sharma KK, Seal A (2020) Spectral embedded generalized mean based K-nearest neighbors clustering with S-distance. Expert Syst Appl 169(4):114,326
49. Sheikholeslami G, Chatterjee S, Zhang A (2000) Wavecluster: a wavelet-based clustering approach for spatial data in very large databases. VLDB J 8(3):289–304
50. Sheng Y, Wang M, Wu T et al (2019) Adaptive local learning regularized nonnegative matrix factorization for data clustering. Appl Intell 49(6):2151–2168
51. Shin KS, Jeong YS, Jeong MK (2012) A two-leveled symbiotic evolutionary algorithm for clustering problems. Appl Intell 36(4):788–799
52. Sieranoja S, Fränti P (2021) Adapting K-means for graph clustering. Knowl Inf Syst 8(11):33–47
53. Strehl A, Ghosh J (2002) Cluster ensembles—a knowledge reuse framework for combining multiple partitions. J Mach Learn Res 3(Dec):583–617
54. Sun G, Cong Y, Wang Q et al (2020) Lifelong spectral clustering. In: Proceedings of the AAAI conference on artificial intelligence, pp 5867–5874
55. Tao X, Wang R, Chang R et al (2019) Spectral clustering algorithm using density-sensitive distance measure with global and local consistencies. Knowl-Based Syst 170:26–42
56. Vassilvitskii S, Arthur D (2006) K-means++: the advantages of careful seeding. In: Proceedings of the eighteenth annual ACM-SIAM symposium on discrete algorithms, pp 1027–1035
57. Veenman CJ, Reinders MJT, Backer E (2002) A maximum variance cluster algorithm. IEEE Trans Pattern Anal Mach Intell 24(9):1273–1280
58. Virmajoki O, Franti P, Kaukoranta T (2002) Iterative shrinking method for generating clustering. In: Proceedings of international conference on image processing, IEEE, pp 2–10
59. Wang H, Yang Y, Liu B (2019a) Gmc: graph-based multi-view clustering. IEEE Trans Knowl Data Eng 32(6):1116–1129
60. Wang H, Yang Y, Liu B et al (2019b) A study of graph-based system for multi-view clustering. Knowl-Based Syst 163:1009–1019
61. Wang TS, Lin HT, Wang P (2017) Weighted-spectral clustering algorithm for detecting community structures in complex networks. Artif Intell Rev 47(4):463–483
62. Wang W, Yang J, Muntz R et al (1997) Sting: a statistical information grid approach to spatial data mining. In: The VLDB journal, pp 186–195
63. Wang Y, Duan X, Liu X et al (2018) A spectral clustering method with semantic interpretation based on axiomatic fuzzy set theory. Appl Soft Comput 64:59–74
64. Wu B, Wilamowski BM (2016) A fast density and grid based clustering method for data with arbitrary shapes and noise. IEEE Trans Ind Inf 13(4):1620–1628
65. Xie H, Zhang L, Lim CP et al (2019) Improving K-means clustering with enhanced firefly algorithms. Appl Soft Comput 84:105,763
66. Xie J, Girshick R, Farhadi A (2016) Unsupervised deep embedding for clustering analysis. In: International conference on machine learning, pp 478–487
67. Xu J, Wang G, Deng W (2016) Denpehc: density peak based efficient hierarchical clustering. Inf Sci 373:200–218
68. Xu Q, Zhang Q, Liu J et al (2020) Efficient synthetical clustering validity indexes for hierarchical clustering. Expert Syst Appl 151:113,367
69. Xu X, Ding S, Shi Z (2018) An improved density peaks clustering algorithm with fast finding cluster centers. Knowl-Based Syst 158:65–74
70. Xu X, Ding S, Wang Y et al (2021) A fast density peaks clustering algorithm with sparse search. Inf Sci 554:61–83
71. Yin L, Li M, Chen H et al (2022) An improved hierarchical clustering algorithm based on the idea of population reproduction and fusion. Electronics 11(17):2735
72. Yoo HW, Jung SH, Jang DS et al (2002) Extraction of major object features using VQ clustering for content-based image retrieval. Pattern Recogn 35(5):1115–1126
73. Zhang C, Fu H, Hu Q et al (2018a) Generalized latent multi-view subspace clustering. IEEE Trans Pattern Anal Mach Intell 42(1):86–99
74. Zhang T, Ramakrishnan R, Livny M (1996) Birch: an efficient data clustering method for very large databases. ACM Sigmod Record 25(2):103–114
75. Zhang T, Ramakrishnan R, Livny M (1997) Birch: a new data clustering algorithm and its applications. Data Min Knowl Disc 1(2):141–182
76. Zhang W, Zang W (2018) A fuzzy density peaks clustering algorithm based on improved DNA genetic algorithm and K-nearest neighbors. In: International conference on intelligent science and big data engineering, Springer, pp 476–487

77. Zhang X, Xu Z (2015) Hesitant fuzzy agglomerative hierarchical clustering algorithms. Int J Syst Sci 46(3):562–576
78. Zhang Z, Liu L, Shen F et al (2018b) Binary multi-view clustering. IEEE Trans Pattern Anal Mach Intell 41(7):1774–1782
79. Zhu X, Zhang S, Li Y et al (2018) Low-rank sparse subspace for spectral clustering. IEEE Trans Knowl Data Eng 31(8):1532–1543
80. Zhu X, Zhu Y, Zheng W (2020) Spectral rotation for deep one-step clustering. Pattern Recogn 105:107,175

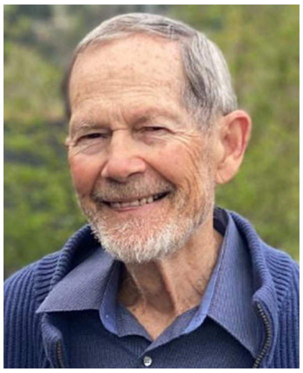
Xiaolu Liu received the B.E. degree in system engineering from the National University of Defense Technology, China, in 2006, and the Ph.D. degree in management science and engineering from the National University of Defense Technology, China, in 2011. She is an associate professor with College of system engineering, National University of Defense Technology. Her research mainly focuses on artificial intelligence and metaheuristics, solving distribution management and satellite scheduling problems.



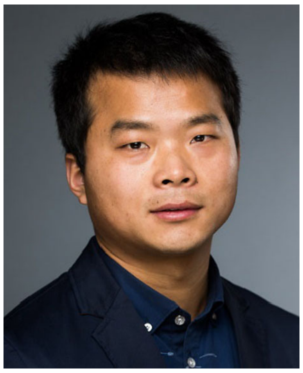
Wenhan Shao received the B.S. degree in software engineering from Huazhong University of Science and Technology, China, in 2017. He is currently working towards his Ph.D. degree in computer software and theory from Laboratory of Smart Computing and Optimization, the School of Computer Science and Technology, Huazhong University of Science and Technology. His research interest includes metaheuristic algorithms for solving the vehicle routing problem and its variants.

Jiaming Chen received the B.E. degree in system engineering from the National University of Defense Technology, China, in 2018, and the master's degree in management science and engineering from the National University of Defense Technology, China, in 2020. His research interests include computation intelligence, evolutionary computation, and adaptive meta-heuristic or hyperheuristic for solving large-scale combinatorial optimization and resource scheduling problems.

Zhipeng Lü received his B.S. degree in applied mathematics from Jilin University, China, in 2001, and the Ph.D. degree in computer software and theory from the Huazhong University of Science and Technology, China, in 2007. He was a Postdoctoral Research Fellow with the LERIA, Department of Computer Science, University of Angers, France, from 2007 to 2011. He is currently a Professor with the School of Computer Science and Technology, Huazhong University of Science and Technology, and the Director of the Institute of Artificial Intelligence and Optimization. His research interests include artificial intelligence, computational intelligence, operations research and adaptive metaheuristics for solving large-scale real-world and theoretical combinatorial optimization, and constrained satisfaction problems.



Fred Glover is Chief Scientific Officer of Entanglement, Inc., in charge of algorithmic design and strategic planning initiatives. He previously served as Chief Technology Officer for OptTek Systems, Inc., the leading provider of optimization software to the simulation industry, and holds the title of Distinguished University Professor, Emeritus, in the Schools of Engineering and Business at the University of Colorado, Boulder. A recipient of the von Neumann Theory Prize, the highest honor of the INFORMS Society, and an elected member of the U. S. National Academy of Engineering, he has authored or co-authored more than 500 published articles and eight books in the fields of mathematical optimization, computer science and artificial intelligence. He also serves on advisory boards for numerous journals and professional organizations and has co-founded several research and consulting companies in the areas of simulation optimization, logistics and analytics.



Junwen Ding received his B.S. degree in computer science from Wuhan Textile University, Wuhan, China, in 2011, and the M.Sc. and Ph.D. degrees in computer software and theory from the Huazhong University of Science and Technology, Wuhan, China, in 2014 and 2017, respectively. He is currently in the postdoctoral program with the Institute of Artificial Intelligence and Optimization, Huazhong University of Science and Technology. His current research interests include metaheuristic algorithms and evolutionary computation for solving classical NP-hard problems and real-world applications, such as production scheduling, facility location, graph partition, and multi-agent path finding.

Affiliations

Xiaolu Liu[1] · **Wenhan Shao**[2] · **Jiaming Chen**[1] · **Zhipeng Lü**[2] · **Fred Glover**[3] · **Junwen Ding**[2]

Xiaolu Liu
lxl_sunny_nudt@live.cn

Wenhan Shao
shaowenhan95@hust.edu.cn

Jiaming Chen
chen-jm@aliyun.com

Zhipeng Lü
zhipeng.lv@hust.edu.cn

Fred Glover
glover@colorado.edu

[1] College of System Engineering, National University of Defense Technology, Deya Road, Changsha, 410000, Hunan, China

[2] School of Computer Science and Technology, Huazhong University of Science and Technology, Luoyu Road, Wuhan, 430074, Hubei, China

[3] College of Engineering & Applied Science, University of Colorado, Boulder, Colorado, 80309-0425, Colorado, USA