ON FUZZY NEIGHBORHOOD BASED CLUSTERING ALGORITHM WITH LOW COMPLEXITY

G. ULUTAGAY AND E. NASIBOV

Abstract. The main purpose of this paper is to achieve improvement in the speed of Fuzzy Joint Points (FJP) algorithm. Since FJP approach is a basis for fuzzy neighborhood based clustering algorithms such as Noise-Robust FJP (NRFJP) and Fuzzy Neighborhood DBSCAN (FN-DBSCAN), improving FJP algorithm would an important achievement in terms of these FJP-based methods. Although FJP has many advantages such as robustness, auto detection of the optimal number of clusters by using cluster validity, independency from scale, etc., it is a little bit slow. In order to eliminate this disadvantage, by improving the FJP algorithm, we propose a novel Modified FJP algorithm, which theoretically runs approximately $n/\log_2 n$ times faster and which is less complex than the FJP algorithm. We evaluated the performance of the Modified FJP algorithm both analytically and experimentally.

1. Introduction

Clustering, the art of finding the natural grouping of data vectors based on their similarity, has been one of the most crucial methods in the statistical learning community for many decades [27, 30]. Traditional clustering techniques are broadly divided into five categories such as hierarchical, model-based, grid-based, partitioning and density-based clustering [12]. Actually, the different classes of algorithms can overlap.

Hierarchical clustering algorithms create a hierarchical decomposition of a database. The hierarchical decomposition is represented by a dendogram, a tree that iteratively splits the database into smaller subsets until each subset consist of only one object. The well-known algorithms for hierarchical clustering are average linkage, single-linkage (SLINK), complete-linkage (CLINK), BIRCH, CURE and CHAMELEON [11, 12, 17, 36]. Hierarchical clustering methods can also be divided into bottom-up or top-down. There are two popular methods for the two procedures, AGNES and DIANA, respectively. To improve the quality of the clustering methods, hierarchical clustering is integrated with other clustering techniques. For instance, BIRCH begins by partitioning objects hierarchically using tree structures, and then applies other clustering algorithms to refine the clusters [12].

Model-based methods hypothesize a model for each of the clusters and find the best fit of the data to the given model. A model-based algorithm such as EM

Received: April 2011; Accepted: August 2012
Key words and phrases: Clustering, Fuzzy neighborhood relation, Complexity, Modified FJP.
This work was supported from TUBITAK with Grant No.111T273 .

or COBWEB may locate clusters by constructing a density function that reflects the spatial distribution of the data points [7, 10]. CLASSIT is an extension of COBWEB for incremental clustering of continuous data. AutoClass is a popular clustering method that uses Bayesian statistical analysis to estimate the number of clusters in industry. SOMs is a neural network approach [12].

Grid-based methods are fast and they handle outliers well. The grid-based methodology can also be used as an intermediate step in many other algorithms. The most important methods for this category are STING, CLIQUE, and WaveCluster [1, 29, 31]. Mountain method is another example which can simultaneously be in the model-based clustering class [34].

Clustering algorithms construct a partition of a database X of n objects into a set of k clusters. The partitioning algorithms typically start with an initial partition of X and then use an iterative control strategy to optimize an objective function. Partitioning methods concentrate on how well points fit into their clusters' prototypes and tend to build clusters of proper convex shapes. The algorithms k-means, k-medoids, PAM, CLARA, CLARANS, and their extensions are the examples of the partitioning clustering methods [18, 24].

k-means is the most popular and well-known partitioning method. It is a centroid based technique. Some improved algorithms based on k-means are also available. For instance, k-modes is used to cluster categorical data, or if mixed numerical and categorical data are of concern, k-prototoypes method can be used. Another variant of k-means is the expectation-maximization algorithm which computes the means based on the weighted measures. If the medoid is computed instead of the mean as a reference point, the k-medoids method is used. PAM is a typical k-medoids method that works effectively for small datasets. In order to deal with larger data sets, a sampling-based method, CLARA, can be used. Another k-medoid type algorithm called CLARANS that combines the sampling technique with PAM, was proposed to improve the quality and the scalability of CLARA.

Fuzzy c-means (FCM) is the most popular fuzzy partitioning algorithm based on the fuzzy membership concept [4, 8]. The FCM approach is integrated with deterministic annealing which takes advantage of conventional noise clustering and deterministic annealing algorithms in terms of better performance for unbalanced data, and robustness against noise and outliers [35]. In order to obtain good performance in large databases, a hybrid method is proposed by combining fuzzy particle swarm optimization (FPSO) with FCM [19].

Density-based clustering methods attempt to discover dense connected components of data, which are flexible in terms of their shape, *i.e.* nonconvex, spherical, linear, elongated, *etc.* Density-based connectivity is used in the algorithms DBSCAN, DBCLASD, and OPTICS while the algorithm DENCLUE exploits space density functions [2, 9, 13, 32]. These algorithms are less sensitive to outliers and can discover clusters of irregular shape. They usually work with spatial data. Spatial objects may include not only points, but also geometrically extended objects as in the algorithm GDBSCAN [28]. An improving 3-phase density-based clustering method for clustering of web pages is proposed in study [5]. The main advantage of the method is its low time complexity. The state-of-the-art of density based

clustering algorithms is handled and advantages of the fuzzy neighborhood relation used in FJP approach is established in the paper [23].

FJP (Fuzzy Joint Points) and its noise robust extension NRFJP (Noise-Robust FJP) handle the concept of neighborhood from the fuzzy point of view and they are density-based methods as well as hierarchical [20, 21, 22, 23, 30]. The main distinction of the FJP-based methods from other fuzzy clustering methods is that the concept of fuzziness of clustering is considered from the hierarchical point of view. Thus, the fuzziness of clustering is determined by the detailedness of elements in forming a set of similar elements. It is obvious that a more detailed consideration of properties implies the disclosure of more distinctions between elements. If properties are considered fuzzier,*i.e.* without going into details, then elements are more similar. In this case, fuzziness of clustering is determined by the detailedness of taking into account the properties being investigated. Then, in the case of a minimal degree of fuzziness, all elements differ from one another to some extent and each element must be considered as an individual cluster. For some degree of fuzziness between 0 and 1, some elements with close properties are similar to one another and belong to the same cluster and more widely spaced elements belong to different clusters.

From the density-based viewpoint, the FJP algorithm is similar to DBSCAN [30]. In the DBSCAN method, it is not necessary to specify in advance the number of clusters. Instead, however, parameters that determine the concept of neighbor and of noise points must be specified. Subsequently, the number of clusters depends on the values of these parameters and is factually determined. In contrast to the DBSCAN method, one feature of the FJP method lies in the existence of a mechanism for auto-determination of the optimal number of clusters without specification of any predetermining parameters whatsoever; this is also the principal advantage of the FJP method.

Scalable FN-DBSCAN combines the ideas of Online FCM and Scalable Density Based Distributed Clustering [25]. In Online FCM, the fuzzy centroids obtained from clustering each of r subsets effectively represent the density of the data set up to that point of the processing [14]. Retention of the weighted centroids allows the algorithm to zero in the centroids of the entire data set. Scalable Density Based Distributed Clustering's (SDBDC) strategy is similar; it stores the covered points and covering radius of each retained point, and combines them at the end [15, 16]. SDBDC only retains points that are not within epsilon distance of each other in order to evenly sample the data subset. Scalable FN-DBSCAN method follows a similar but not identical approach to Online FCM [25]. Scalable FN-DBSCAN combines these ideas in the sense that the data is broken into subsets and points sampling the density from each subset are retained with weight. The retained points from each subset are combined and a weighted version of FN-DBSCAN is run on these points to create clusters. The last step is assigning points from the entire data set to the defined clusters.

Fuzzy neighborhood based clustering algorithms like NRFJP and FN-DBSCAN are based on FJP approach. Besides many advantages, the FJP algorithm runs slower than the DBSCAN algorithm. It is obvious that an improvement in FJP

algorithm, will also directly affect related algorithms. So, in this study, we mainly propose the Modified FJP (MFJP) algorithm as an improvement for the computational performance of FJP.

The rest of the paper is organized as follows: Section 2 contains some theoretical background about the clustering structure and neighborhood relations. Section 3 and Section 4 describe the FJP and the MFJP algorithms, respectively and each section investigates the algorithms' worst time complexities analytically. Section 5 presents an experimental comparison for computation performances of the above-mentioned algorithms on ten datasets. Section 6 concludes with a summary of the study.

2. Clustering Structure in Fuzzy Neighborhood Relation

Let a data set $X = \{x_1, x_2, ..., x_n\}$, $x_i = (x_{i1}, x_{i2}, \ldots, x_{ip})$, $i = 1, \ldots, n$ be given. It is required to divide the set X into homogenous groups, *i.e.* to partition its elements.

The clusters $X_1, X_2, \ldots, X_t$ satisfying the following conditions are called the partition of the set X into clusters:

$$X = \bigcup_{i=1}^{t} X_i \tag{1}$$

$$X_i \bigcap X_j = \emptyset, \forall i, j : i \neq j. \tag{2}$$

It is obvious that, any clustering result can be defined by an appropriate $n \times n$ relation matrix, $T = \|t_{ij}\|_{i,j=1,...,n}$. Then,

$$t_{ij} = \begin{cases} 1, & if\ x_i\ and\ x_j\ are\ from\ the\ same\ class \\ 0, & otherwise \end{cases} \tag{3}$$

In clustering, it is not important in which cluster an element falls, but whether the elements are in the same cluster or not is important. Clustering matrix T has the following properties:

i. $\forall i, j : \ t_{ij} \in \{0, 1\}\ and\ t_{ii} = 1.$ (4a)

ii. $\forall i, j : \ t_{ij} = t_{ji}$ (4b)

iii. $\forall i, j, k : \ t_{ij} \geq \min\{t_{ik}, t_{kj}\}$ (4c)

Proposition 2.1. *The rows (or columns) appropriate to the elements from the same cluster are the same in the clustering matrix T. In other words, if $t_{ij} = 1$ for any x_i and x_j, then $t_{ik} = t_{jk}, \ k = 1, ..., n$ (or $t_{ki} = t_{kj}, \ k = 1, ..., n$, for columns).*

Proof. Let's take the rows into account and assume the opposite of the proposition. Let the elements x_i and x_j be in the same cluster, i.e. $t_{ij} = t_{ji} = 1$, but let's assume

$$\exists k : t_{ik} \neq t_{jk}. \tag{5}$$

Since $t_{ik}, t_{jk} \in \{0, 1\}$, the condition equation(5) should be

$$t_{ik} = 1, \ t_{jk} = 0 \tag{6}$$

or

$$t_{ik} = 0,\ t_{jk} = 1. \tag{7}$$

Assume that equation(6)holds without the loss of generality. Then, due to the transitivity property iii), it should be $t_{jk} \geq \min\{t_{ji}, t_{ik}\}$. Since

$$0 = t_{jk} \geq \min\{t_{ji}, t_{ik}\} = \min\{1, 1\} = 1, \tag{8}$$

this contradiction means that the conditions equation(6) and equation(8) can not hold, thus the condition equation(6) can not be true.

The proof for columns is similar to that of rows. □

Definition 2.2. If the clustering matrices of any two clustering results are the same, then it can be concluded that these clustering results are the same.

Definition 2.3. A function $N : X \times X \to [0, 1]$ that satisfies the following properties is called a fuzzy neighborhood relation:

$$0 \leq N(x, y) \leq 1 \forall x, y \in X, \tag{9a}$$

$$N(x, y) = 1 \Leftrightarrow x = y, \forall x, y \in X \tag{9b}$$

$$N(x, y) = N(y, x) \forall x, y \in X. \tag{9c}$$

Clustering the set $X = \{x_1, \ldots, x_n\}$ can be formed when the fuzzy neighborhood relation N between elements are given. In such a situation, clustering matrix, T, is formed as the transitive closure of the neighborhood relation matrix [26]. Thus, the matrix T computed as $T = \hat{N}$, where $\hat{N}$ is the transitive closure of the relation N, is a clustering matrix that reflects a certain clustering result. It is obvious that, if the basic neighborhood relation N is a fuzzy relation, then its transitive closure $\hat{N}$ will also be a fuzzy relation. Definition and investigations about the transitive closure will be handled in more detail in Section 4.

Let $T = \hat{N}$, and $\alpha \in (0, 1]$ be a fixed degree. The α-level set, T^α of the fuzzy relation T is

$$t_{ij}^\alpha = \begin{cases} 1, & if \quad t_{ij} \geq \alpha \\ 0, & if \quad t_{ij} < \alpha \end{cases}. \tag{10}$$

Since the relation N holds the conditions (9a)-(9c) and a transitive closure matrix is transitive, the matrix $T = \hat{N}$ also holds the conditions (4a)-(4c). So, due to the resolution principle, T^α-level matrix for $\forall \alpha \in (0, 1]$ also holds the conditions(4a)-(4c). Consequently, the matrix T^α is a crisp relation matrix and forms a certain clustering structure of the elements of X. Hence, for a given α-level, a T^α clustering matrix, consequently a clustering structure can be constructed. Thus, the elements in the same cluster are in relation T with membership degree more than or equal α, while the degree of the relation T is less than α among elements which are from different clusters.

Let's form an array α_i, $i = 0, 1, \ldots, t$, by sorting the elements of the clustering matrix T in decreasing order and taking only one of the equal elements.

Lemma 2.4. *i) The clustering structures corresponding to the different* α_i, $i = 0, 1, 2, ...$ *values of the matrix* $T = \hat{N}$ *are different.*

ii) For each fixed $i \in \{0, 1, 2, ...\}$, *the clustering structures convenient to the* α*-degree in the interval* $[\alpha_{i+1}, \alpha_i)$ *are the same.*

Proof. i) Let two different elements of the clustering matrix T be $t_{i_1 j_1} = \alpha_1$, $t_{i_2 j_2} = \alpha_2$. Assume that $\alpha_1 > \alpha_2$. So, $t_{i_1 j_1} = 1$, $t_{i_2 j_2} = 0$ hold in the T^{α_1}-level matrix while $t_{i_1 j_1} = 1$, $t_{i_2 j_2} = 1$ hold in the T^{α_2}-level matrix. Hence $T^{\alpha_1} \neq T^{\alpha_2}$. According to Definition 2.2, the clustering structures will also be different.

ii) Since $\forall \alpha : \alpha \in [\alpha_{i+1}, \alpha_i)$ holds $\alpha \geq \alpha_{i+1}$, the following can be written:

$$T^{\alpha} \subseteq T^{\alpha_{i+1}} \tag{11}$$

The crisp set $T^{\alpha_{i+1}} \backslash T^{\alpha}$ must contain elements with membership degrees α' such that $\alpha_{i+1} < \alpha' < \alpha < \alpha_i$. On the other hand, since the T matrix does not have any element getting values from the interval (α_{i+1}, α_i), we can write

$$T^{\alpha_{i+1}} \backslash T^{\alpha} = \emptyset \tag{12}$$

From equation(11) and equation(12) entails $T^{\alpha} = T^{\alpha_{i+1}}$, which completes the proof. □

Let us denote

$$\Delta_i = \alpha_i - \alpha_{i+1}, i = 0,\ 1, \ldots, t-1, \tag{13}$$

where α_i, $i = 0,\ 1, \ldots, t$ are different elements of the T matrix, and

$$\Delta^* = \max_{i=0,\ldots,t-1} \Delta_i. \tag{14}$$

Lemma 2.5. *The worst time complexity of* Δ^* *calculated by formula equation(14) is*

$$\mathrm{O}\left(\frac{n\,(n-1)}{2}\ \log_2 \frac{n(n-1)}{2}\right) = \mathrm{O}(n^2 \log_2 n). \tag{15}$$

Proof. There are $n \times n$ elements in the matrix T, this matrix is symmetric and has a diagonal with value 1. The maximum number of remaining different elements can be $\frac{n(n-1)}{2}$ and the worst time complexity of sorting of such number of elements is

$$\mathrm{O}\left(\frac{n\,(n-1)}{2}\ \log_2 \frac{n(n-1)}{2}\right) = \mathrm{O}(n^2 \log_2 n^2) = \mathrm{O}(2n^2 \log_2 n) = \mathrm{O}(n^2 \log_2 n). \tag{16}$$

The proof is completed. □

3. FJP Algorithm and Its Complexity

In this section, the complexity of the FJP algorithm will be analyzed. Let's deal again with the solution of the problems (1)-(2). The FJP algorithm was suggested in studies [5, 20, 21, 22, 23, 30] in order to solve the mentioned problem. In case of fuzzy relations, the FJP algorithm uses the fact that the elements are in transitive relation with each other when it constructs homogenous clusters based on fuzzy logic. Of course, fuzzy clustering problem can be solved according to different α-levels, and at the end different clustering structures can be formed. The main

property of the FJP algorithm is that it does not only form a clustering structure based on a certain α degree, but also it determines the optimal clustering structure which is the fundamental question in clustering problem. FJP algorithm uses V_{FJP} cluster validity index in order to determine the optimal clustering structure, *i.e.* the optimal number of clusters.

Let $X(\alpha) = \{X^k, k = 1...t\}$ be any clustering structure of the set $X = \{x_1, ..., x_n\}$ convenient to the α membership degree. The cluster validity criteria V_{FJP} is based on the largest α-level change interval that does not affect the structure of clusters. In other words, the cluster structure, $X(\alpha) = \{X^k, k = 1...t\}$, which gives maximum value to the functional V_{FJP} given below is considered as optimal [22]:

$$V_{FJP}(X(\alpha)) = \max_k \left[\min_{i\neq j}\{d(x_i, x_j)|\, x_i \in X^k, x_j \in X^k\}\right] - \min_{i\neq j} d(X^i, X^j), \quad (17)$$

where

$$d(x_i, x_j) = \left(\sum_{k=1}^{p}(x_{ik} - x_{jk})^2\right)^{1/2} \quad (18)$$

is the Euclidean distance between the elements, x_i and x_j. However, in order to eliminate the scale dependency, first of all, all data are normalized by using the following transformation:

$$x_{ik} := \frac{x_{ik}}{(\max_i x_{ik} - \min_i x_{ik})\sqrt{p}} \quad (19)$$

where $\frac{1}{\sqrt{p}}$ is the multiplier that guarantees to take all data into a sphere with diameter 1, so $\max_{i,j} d(x_i, x_j) \leq 1$ holds for normalized data. In formula (17), the distance $d(X^i, X^j)$ between the sets X^i and X^j is computed as follows:

$$d(X^i, X^j) = \min\{d(x_i, x_j)|x_i \in X^i, x_j \in X^j\}. \quad (20)$$

The pseudocode of the FJP algorithm is given below:

FJP1. Calculate: $d_{ij} := d(x_i, x_j)$, $i, j = 1, .., n$; $d_{\max} := \max_{i,j=1,..,n} d_{ij}$. Set $\alpha_0 := 1$;

FJP2. Calculate the fuzzy neighborhood relation $N_{ij} := 1 - \frac{d_{ij}}{d_{\max}}$, $i, j = 1, .., n$;

FJP3. Calculate the transitive closure $T \equiv \hat{N}$ of the relation N;

FJP4. Denote $y_i := x_i$, $i = 1, .., n$; $t := 1$; $k := n$;

FJP5. Calculate: $d(y_i, y_j) = \min\{d(x', x'')|x' \in y_i, x'' \in y_j\}$, $i, j = 1, .., k$;

$$d_t := \min_{i\neq j} d(y_i, y_j); \alpha_t := 1 - \frac{d_t}{d_{\max}};$$

FJP6. Call the procedure *Clusters* (α_t) to calculate convenient clustering partition $X^1, X^2, ..., X^k$, and to constitute number k of these sets;

FJP7. If $k > 1$, then denote $y_i := X^i$, $i = 1, .., k$; set $t = t + 1$ and go to Step5; If $k = 1$, then go to Step 8.

FJP8. Calculate: $\Delta\alpha_i := \alpha_i - \alpha_{i+1}$, $i = 0, .., t - 1$; $z := \arg\max_{i=0,..,t-1} \Delta\alpha_i$**;**

FJP9. Call the procedure $Clusters(\alpha_z)$ with parameterα_z which completes the optimal partition of the set X with convenient clusters $X^1, ..., X^{k_z}$.
End.

Procedure *Clusters* (α):

Input: α

Output: α-fuzzy joint sets $X^1, X^2, ..., X^k$; k - number of these sets;

Cl1. $S := X = \{x_1, x_2, ..., x_n\}$; $k := 1$;

Cl2. Get the first element $x' \in S$ of the set S;

Create sets: $X^k := \{x'' \in S | T(x', x'') \geq \alpha\}$; $S := S \backslash X^k$;

Cl3. If $S \neq$, then let $k := k + 1$ and go to Step 2;

Otherwise go to Step 4;

Cl4. Return the sets $X^1, X^2, ..., X^k$ and number k of these sets.

End.

The procedure *Clusters* (α), used in the realization of the FJP algorithm, calculates a partition of the set $X = \{x_1, x_2, ..., x_n\}$ into α-fuzzy joint sets whose elements are in relation T with degree more than or equal α with a fixed membership degree α.

Theorem 3.1. *The worst time complexity of the FJP algorithm is*

$$\mathrm{O}\left(\frac{5n^4 - 8n^3 + 7n^2}{4}\right) = \mathrm{O}(n^4). \tag{21}$$

Proof. Let's investigate the complexity of the main steps of the FJP algorithm in detail.

The complexity of FJP1 step is $\mathrm{O}(n^2)$;

The complexity of FJP2 step is $\mathrm{O}(n^2)$;

FJP3 step is the most complex step. If the transitive closure is directly computed as $T = N \circ N \circ \cdots \circ N = N^{n-1}$, it is necessary to multiply N for $n-2$ times, so the complexity is $\mathrm{O}(n^3(n-2)) = \mathrm{O}(n^4 - 2n^3)$.

The complexity of FJP5 step: We need to compare the pairwise distances between all n elements in order to determine the minimum distance between clusters. It makes $n(n-1)/2$ comparisons.

The complexity of FJP6 step: In this step *Clusters* (α) procedure is called. The complexity of this procedure is approximately $\mathrm{O}(n)$.

The steps FJP5 and FJP6 should be applied for each α-level in which the number of clusters is affected. According to Lemma 2.4, the clustering structure changes only at levels which are equal to the different elements of the clustering matrix $T \equiv \hat{N}$. As we mentioned in the proof of the Lemma 2.5, there are maximum $\frac{n(n-1)}{2}$ different elements in matrix T. So, the steps FJP5 and FJP6 will be repeated maximum $\frac{n(n-1)}{2}$ times.

Consequently, the full complexity of the FJP algorithm will be:

$$O(n^2) + O(n^2) + O(n^4 - 2n^3) + \frac{n(n-1)}{2}\left(O\left(\frac{n(n-1)}{2}\right) + O(n)\right)$$

$$= \mathrm{O}\left(\frac{5n^4 - 8n^3 + 7n^2}{4}\right) = \mathrm{O}(n^4) \tag{22}$$

which completes the proof. □

4. **Modified FJP Algorithm and Its Complexity**

As it is mentioned above, the difference of the FJP algorithm from classical neighborhood-based clustering algorithms (*e.g.* DBSCAN) is that it uses fuzzy relation between points [5, 20, 21, 22]. Such a fuzzy relation is based on the distance between points. In general, any fuzzy relation between points can be of interest.

One of the main steps in the FJP and the NRFJP algorithms is the computation of the transitive closure matrix. In these algorithms, relation matrix is multiplied by itself for $n-1$ times in order to reach transitive closure matrix. However, in the Modified FJP algorithm, in order to reach the transitive closure matrix, multiplication procedure is realized for $\log_2 n$ times. By the way, since the computation task is lightened the result is reached faster and the computational complexity is reduced.

Moreover, the procedure *Clusters* (α) is applied, *i.e.* clustering is realized, for all α-levels which affect the number of clusters in the FJP algorithm while the procedure *Clusters*(α) is applied only once for the optimal level in the Modified FJP (MFJP) algorithm. So, we suggest the Modified FJP algorithm given below which is easier to understand and faster than the FJP algorithm.

The pseudocode of the Modified FJP Algorithm is given below:

MFJP1: Compute pairwise distances between elements: $d_{ij} := d(x_i, x_j), \quad i, j = 1, ..., n$ and denote $d_{\max} := \max\limits_{i,j=1,...,n} d_{ij}$.

MFJP 2: Compute the fuzzy neighborhood relation

$$N_{ij} := 1 - \frac{d_{ij}}{d_{\max}}, \quad i, j = 1, ..., n.$$

Note that, instead of the formulation given above, we can construct the neighborhood relation by using any neighborhood function satisfying the conditions equation(9a)-(9c).

MFJP 3: Compute the transitive closure matrix $T \equiv \hat{N}$.

MFJP 4: Sort the elements of the transitive closure matrix in decreasing order and form an array $\alpha_i, i = 0, 1, 2, ...$ by getting only one of the equal elements.

Determine the α_z-level appropriate to the max change interval:

$$z := \arg \max_i \quad \Delta\alpha_i,$$

where $\Delta\alpha_i := \alpha_i - \alpha_{i+1}, \quad i = 0, ..., t-1$,

MFJP 5: Call the procedure $Clusters(\alpha_z)$ in order to construct the clusters convenient to the α_z-level.

End.

As mentioned previously, the main factor that affects the complexity in the FJP algorithm is the computation of the transitive closure matrix. But, as we will see later in this section, since the transitive closure matrix is calculated in a more effective scheme, the worst time complexity of the MFJP algorithm is remarkably less than that of the FJP algorithm.

Definition 4.1. Let $N : X \times X \to [0,1]$ be a fuzzy neighborhood relation. A relation (matrix) $\bar{N} = (\bar{n}_{ij})_{n\times n}$, calculated as

$$\overline{n}_{ij} = \max_k \ \min \{n_{ik}, n_{kj}\}, i, j = 1, .., n, \tag{23}$$

is called a *max-min* composition of the relation N, and is denoted as $N \circ N$.

So, based on Definition 4.1, we can say that the relation N is transitive if and only if $N \circ N \subseteq N$ is satisfied.

Definition 4.2. Let $N : X \times X \to [0,1]$ be a fuzzy neighborhood relation. A relation (matrix) T, calculated as

$$T = N \bigcup N^2 \bigcup ... \bigcup N^n \bigcup N^{n+1} \bigcup ... \tag{24}$$

is called a transitive closure of the relation N where

$$N^1 = N$$

and

$$N^k = N^{k-1} \circ N, k \geq 2.$$

Note that, the transitive closure of any relation is its nearest covering transitive relation [6, 26].

Proposition 4.3. *Let X be an n-element set and $N : X \times X \to [0,1]$ be a reflexive relation. Then the successive powers of N hold relations below:*

$$I \subset N \subset N^2 \subset \cdots \subset N^{n-1} = N^n = N^{n+1} = ... \tag{25}$$

Consequently, $T = N^{n-1}$ holds.

Since the transitive closure, T of any relation N, is a transitive relation, any fuzzy relation, N, that has properties of reflexivity and symmetry can be transformed into the nearest fuzzy transitive relation by at most $n-2$ compositions. That is,

$$N^{n-1} = N \circ N \circ \cdots \circ N = T. \tag{26}$$

Proposition 4.4. [33] *Let $N : X \times X \to [0,1]$ be a neighborhood relation matrix, then after finite times of compositions:*

$$N \to N^2 \to N^4 \to \cdots \to N^{2^k} \to \cdots \tag{27}$$

there must exist a positive integer k such that

$$N^{2^k} = N^{2^{(k+1)}} = N^{2^{(k+2)}} = \cdots . \tag{28}$$

Theorem 4.5. *Let X be an n element set and $N : X \times X \to [0, 1]$ is a reflexive relation. Then after $k = \lceil \log_2(n-1) \rceil$ compositions, the transitive closure, $T \equiv \hat{N}$ of the matrix N is reached:*

$$N \to N^2 \to N^4 \to \cdots \to N^{2^k} = T \tag{29}$$

Proof. According to Proposition 4.3, if X contains n elements and the relation N is reflexive, then

$$T = N^{n-1} = N^n = N^{n+1} = \ldots \tag{30}$$

On the other hand, for k which satisfies the condition below

$$2^k \geq n - 1, \tag{31}$$

the equality $N^{2^k} = N^{n-1} = T$ will be satisfied due to the Proposition 4.3. So, the minimum integer value to hold the condition (31) is determined as follows:

$$k = \lceil \log_2(n-1) \rceil . \tag{32}$$

which completes the proof. □

Theorem 4.6. *The worst time complexity to determine the transitive closure matrix, T is as follows:*

$$\mathrm{O}(n^3 \log_2(n-1)). \tag{33}$$

Proof. Multiplication of two $n \times n$ matrices needs n^3 times computations. If we take Theorem 2 into account, we can say that the worst time complexity to determine T is $\mathrm{O}(n^3 \log_2(n-1))$. The proof is completed. □

Theorem 4.7. *The worst time complexity of the Modified FJP algorithm is:*

$$\mathrm{O}\left(n^3 \log_2(n-1) + n^2(2 + \log_2 n) + n\right) = \mathrm{O}(n^3 \log_2 n). \tag{34}$$

Proof. Let us investigate the complexity of the main steps of the MFJP algorithm in detail.

The complexity of MFJP1 step is $\mathrm{O}(n^2)$;

The complexity of MFJP2 step is $\mathrm{O}(n^2)$;

MFJP3 step is the most complex step. If we take into account the scheme proposed in the Modified FJP algorithm in order to calculate T matrix, according to Theorem 4.6, we need $\mathrm{O}(n^3 \log_2(n-1))$ computations.

The complexity of MFJP4 step: Since there are maximum $\frac{n(n-1)}{2}$ different elements in symmetrical and reflexive $n \times n$ matrix T, according to the Lemma 2.5, we need maximum $\mathrm{O}\left(\frac{n(n-1)}{2} \log_2 \frac{n(n-1)}{2}\right) = \mathrm{O}(n^2 \log_2 n)$ computations in order to sort the elements of T matrix in decreasing order, and then to find the maximal change interval.

In MFJP5 step, the procedure *Clusters*(α) with complexity $\mathrm{O}(n)$ is called only once.

Consequently, the full worst time complexity of the MFJP algorithm will be:

$$\begin{aligned} &O(n^2) + O(n^2) + O(n^3 \log_2(n-1)) + O(n^2 \log_2 n) + O(n) \\ &= \mathrm{O}\left(n^3 \log_2(n-1) + n^2(2 + \log_2 n) + n\right) = \mathrm{O}(n^3 \log_2 n) \end{aligned} \tag{35}$$

which completes the proof.

□

Eventually, as it is seen from the Theorems 3.1 and 4.7, the Modified FJP algorithm has approximately $\frac{n}{\log_2 n}$ times less computational complexity than the FJP algorithm has.

5. **Experimental Results**

In this section we will make a comparison among the run-times of the above-mentioned DBSCAN, FJP and the Modified FJP algorithms.

In order to compare the effectiveness of the algorithms, we use ten sample databases which are depicted in Figure 1. DB3, DB4, DB8 and DB9 are the datasets which were used by the authors in [22]. DB3 is the famous IRIS data set of Anderson with two-dimensions, petal width and petal length. DB4 is Bensaid's data set with some noise added [3]. DB8 is a nonconvex simulated data set which looks like a horseshoe with a ball inside it. DB6 is a data set which was used by Ester et.al. to implement their algorithms, DBSCAN and GDBSCAN [9, 28]. DB10 is again Bensaid's data set but with more noise added. So, we are able to compare run times of the algorithms more clearly. The other data sets are a mixture of clusters with arbitrary shapes like nonconvex, spherical, linear or elongated.

We have implemented the DBSCAN, FJP and the Modified FJP algorithms in algorithmic language C++. All experiments were run on a Pentium IV, 2.66 Mhz CPU, 2 Gb RAM PC.

Since the basic FJP algorithm does not perform with noisy datasets, we run all three algorithms after applying NoiseFilter procedure in order to filter out noise points [22]. The NoiseFilter procedure is used to partition the dataset X into two disjunctive core X_{core} and noise X_{noise} sets, *i.e*

$$X = X_{core} \bigcup X_{noise} \tag{36}$$

and

$$X_{core} \bigcap X_{noise} = \emptyset. \tag{37}$$

The NoiseFilter procedure runs with two parameters such as ε_1 and ε_2. ε_1 parameter determines the neighborhood radius while ε_2 parameter is the minimum neighborhood cardinality around a point in order to be a core point. The points which can not satisfy this condition are handled as noise, and the algorithms run with discarding noise points.

Let $N(x)$ denote a fuzzy neighborhood set of given point $x \in X$ on the base of the fuzzy relation T, *i.e.*

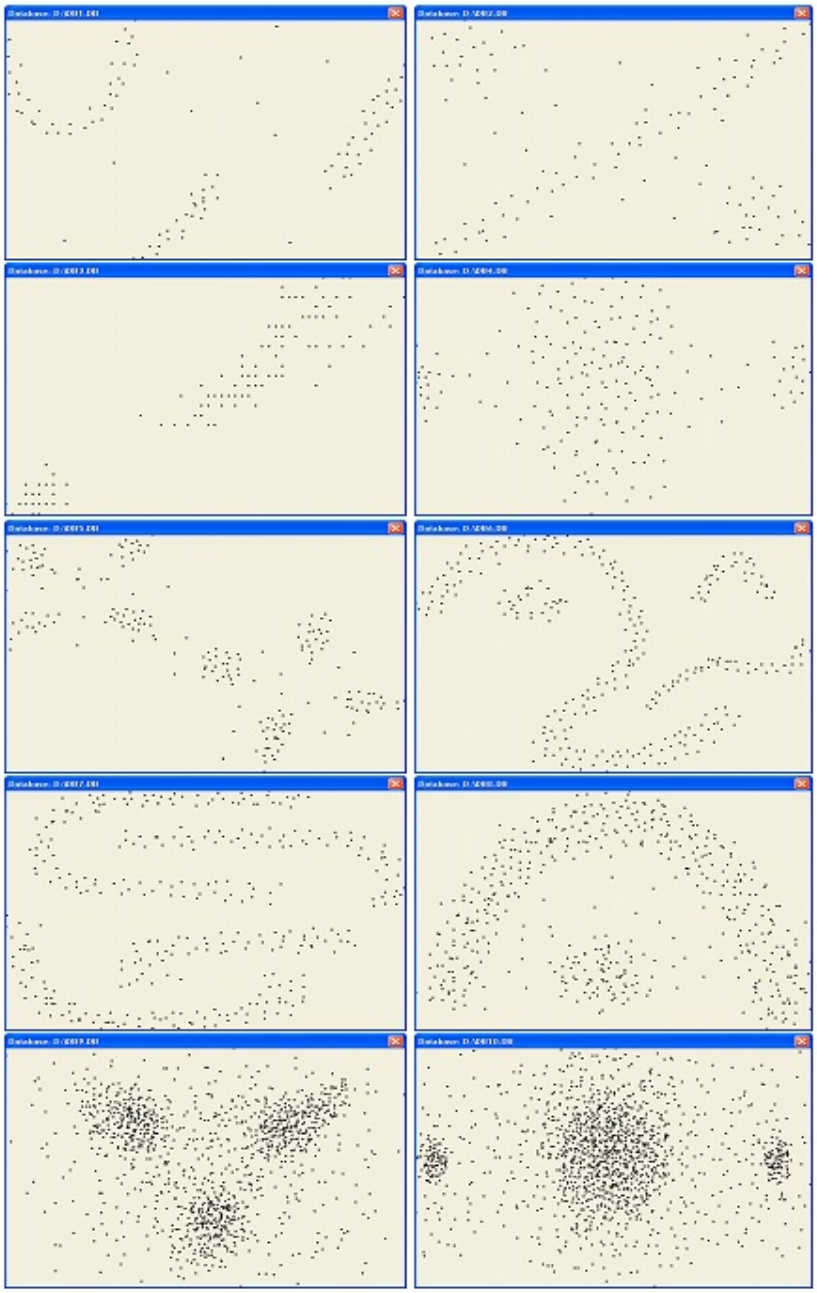

FIGURE 1. Data Sets Used in Experiments

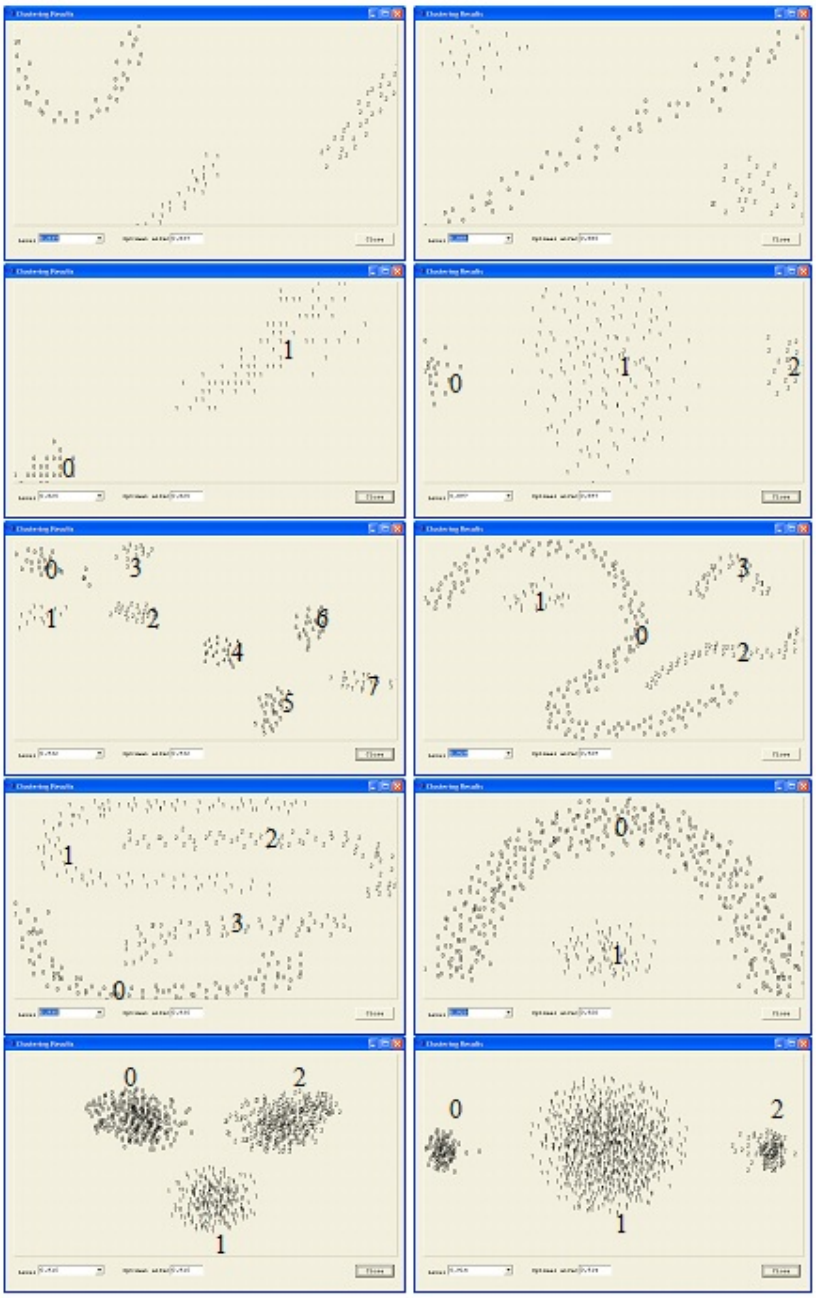

FIGURE 2. Clustering Results for Filtered Data Sets

$$N(x) = \{(y,\, T(x,y)) \,|\, y \in X\}. \tag{38}$$

Let

$$N(x, \varepsilon_1) = \{y \in X \,|\, T(x,y) \geq \varepsilon_1\} \tag{39}$$

be the ε_1-level set of $N(x)$, *i.e.* fuzzy ε_1-neighborhood set of the point $x \in X$.

Suppose that

$$card\ N(x, \varepsilon_1) = \sum_{y \in N(x,\varepsilon_1)} T(x,y) \tag{40}$$

is the fuzzy cardinality of the set $N(x, \varepsilon_1)$. If

$$card\ N(x, \varepsilon_1) < \varepsilon_2 \tag{41}$$

holds, then $x \in X$ is called a noise point with parameters ε_1, ε_2 for given $\varepsilon_1 > 0$, $\varepsilon_2 > 0$, and it is discarded from the clustering process. Note that the algorithms run with discarding noise points, *i.e.* all the algorithms are applied to the sets X_{core}. The NoiseFilter procedure is given below:

Procedure $NoiseFilter(\varepsilon_1, \varepsilon_2)$:
Input: ε_1 and ε_2;
Output: The sets X_{core} and X_{noise};
Step 1. Let $X \equiv \{x_1, x_2, ..., x_n\}$ is the set of initial points, $X_{noise} =$;
Step 2. For each element $x \in X$ repeat the steps 3 and 4:
Step 3. Calculate $card\ N(x, \varepsilon_1) = \sum_{y \in N(x,\varepsilon_1)} T(x,y)$;
Step 4. If $card\, N(x, \varepsilon_1) < \varepsilon_2$, then mark x as noise point:$X_{noise} = X_{noise} \bigcup \{x\}$;
Step 5. Let $X_{core} = X \backslash X_{noise}$;
Step 6. Return the sets X_{core} and X_{noise}.
End.

For a more detailed information about the procedure NoiseFilter see [22]. The number of points of the datasets after applying the procedure NoiseFilter,*i.e.* the number of points in X_{core} sets, and convenient parameters are shown in Table 1.

The filtered data sets, X_{core}, and the clustering results found by three of the algorithms (FJP, MFJP and DBSCAN) are given in Figure 2. Different clusters are marked by different digits. Note that these results are determined directly by FJP and MFJP algorithms whereas many experiments with different values of the parameters ε and MinPts were conducted for DBSCAN algorithm in order to find the correct clustering results.

After applying each algorithm to all data sets ten times, the average runtime of the DBSCAN, FJP, and the Modified FJP algorithms for given databases are shown in Table 2.

In general, time complexity of the DBSCAN algorithm is circa $O(n \log n)$ [9]. On the other hand, as it was mentioned in Theorems 3.1 and 4.7, the worst time complexities of FJP and MFJP algorithms are approximately $O(n^4)$ and $O(n^3 \log_2 n)$, respectively. Thus, the MFJP is theoretically $\frac{n^3 \log n}{n \log n} = n^2$ times weaker than the

Database	**Number of points**	**Number of points after filtering**	ε_1	ε_2
DB1	104	94	0.90	0.20
DB2	127	110	0.95	0.20
DB3	150	143	0.90	0.20
DB4	200	186	0.90	0.20
DB5	200	170	0.96	0.20
DB6	294	281	0.96	0.30
DB7	334	330	0.95	0.20
DB8	486	460	0.90	0.30
DB9	1130	767	0.90	0.45
DB10	1336	1053	0.90	0.20

TABLE 1. NoiseFilter Parameters Implemented to the Databases

Database	**Number of points**	**DBSCAN**	**FJP**	**Modified FJP**
DB1	104	7.08	0.83	0.14
DB2	127	7.71	1.53	0.21
DB3	150	9.03	4.51	0.44
DB4	200	12.18	11.56	1.10
DB5	200	10.66	7.52	0.76
DB6	294	19.42	60.77	5.11
DB7	334	25.48	115.11	9.58
DB8	486	44.97	440.97	31.26
DB9	1130	111.62	3873.36	218.42
DB10	1336	220.37	13673.60	765.14

TABLE 2. Comparison of Run Times of the DBSCAN, FJP and Modified FJP Algorithms in Seconds

DBSCAN algorithm is. On the other hand, FJP and MFJP algorithms have an advantage of using an integrated cluster validity mechanism to find optimal clustering results directly. So, by running these algorithms only once (for each cycle), it is possible to reach optimal clustering results. Naturally, these algorithms need much more time. So, it would not be that much correct to compare FJP and MFJP algorithms directly with DBSCAN algorithm, since they are different qualitatively (the former ones include an integrated cluster validity mechanism, but the latter does not!). However, in order to have an idea, we compare DBSCAN algorithm's results not by using direct computational time, but by changing the values of ε_1 and ε_2 parameters by increasing 0.01 and 0.1, respectively, within the interval $[0, 1]$. Moreover, in order to eradicate the scale-dependency problem of DBSCAN algorithm, a

modified version of this algorithm is used. Instead of ε and $MinPts$ parameters, both of which are scale-dependent, the modified algorithm uses scale-independent ε_1 and ε_2 parameters to determine neighborhood radius and neighborhood density, respectively.

As it is seen from Table 2, for the Modified FJP algorithm it takes a few seconds to reach the results while the runtime of the FJP algorithm increases dramatically with respect to the number of points in the datasets (Figure 3).

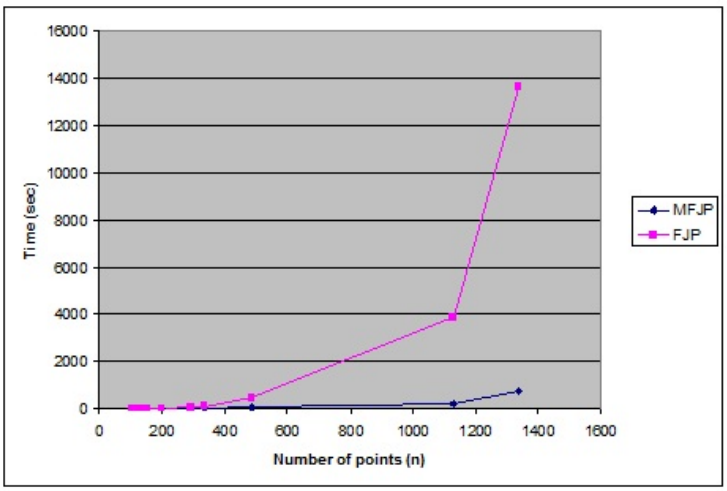

FIGURE 3. Computation Times of FJP and MFJP Algorithms

The Modified FJP algorithm is also faster than the DBSCAN algorithm for small data sets ($n < 600$). However, the larger the data set, the more advantageous is DBSCAN algorithm with respect to computational speed (Figure 4).

Generally, for large data sets, both FJP and Modified FJP algorithms have a disadvantage of using more computational time in comparison with DBSCAN algorithm. On the other hand, the FJP and Modified FJP algorithms have an integrated mechanism to determine the optimal cluster structure which recovers this disadvantage exceedingly.

6. **Conclusion**

The FJP-based clustering algorithms resemble the classical crisp neighborhood-based algorithms such as DBSCAN, but differ from them in such a way that they use the concept of fuzzy neighborhood in order to compute the similarity relations. Although FJP has many advantages such as robustness, auto detection of the optimal number of clusters by using integrated cluster validity mechanism, *etc.*, it looks slow when the worst time complexity is of concern. In order to eliminate this disadvantage, we improved FJP algorithm's time complexity and proposed a novel

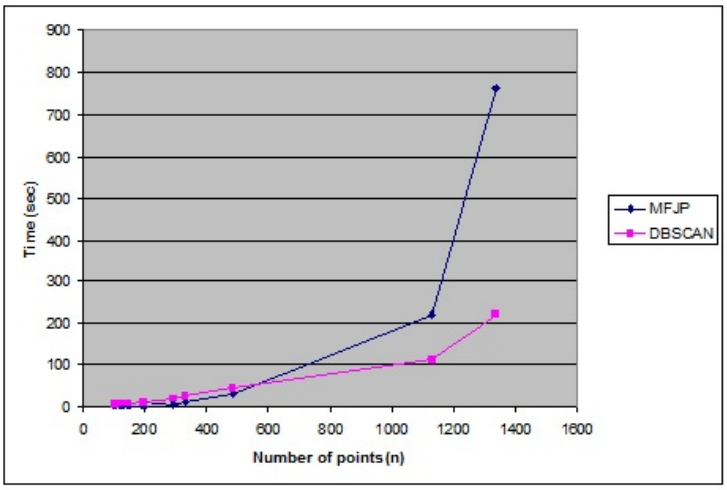

FIGURE 4. Computation Times of DBSCAN and MFJP Algorithms

Modified FJP algorithm, which theoretically runs approximately $\frac{n}{\log_2 n}$ times faster than the known FJP algorithm and which is easier to understand.

In order to compare the effectiveness of the algorithms, we use ten sample datasets with various sizes and shapes. Since the basic FJP algorithm does not perform with noisy datasets, we run all three algorithms after applying NoiseFilter procedure in order to filter out noise points.

The computational experiments show that besides the theoretical superiority, the practical performance of the Modified FJP algorithm is also very high in comparison with the FJP algorithm, especially in large databases.

Nevertheless, when dealing with large databases (e.g. $n > 10000$), processing $n \times n$ dimensional matrices arises some overloading problems related with the virtual memory of the computer, which could be the subject of the future research.

Acknowledgements. Authors would like to express their sincere thanks to anonymous reviewers for their valuable comments. This study was partially supported by the Scientific and Technological Research Council of Turkey (TUBITAK) under grant No. 111T273.

REFERENCES

[1] R. Agrawal, J. Gehrke, D. Gunopulos and P. Raghavan, *Automatic subspace clustering of high dimensional data for data mining applications*, Proceedings of the 1998 ACM-SIGMOD Int. Conference on Management of Data, Seattle, Washington, June 1998.

[2] M. Ankerst, M. M. Breunig, H. P, Kriegel and J. Sander, *OPTICS: ordering points to identify the clustering structure*, In: Proceedings of ACM SIGMOD International Conference on Management of Data, Philadelphia, PA, (1999), 49–60.

[3] A. M. Bensaid, L. O. Hall, J. C. Bezdek, L. P. Clarke, M. L. Silbiger, J. A. Arrington and R. F. Murtagh, *Validity-guided (re)clustering with applications to image segmentation*, IEEE Transactions on Fuzzy Systems, **4** (1996), 112–123.

[4] J. C. Bezdek, *Fuzzy mathematics in pattern classification*, PhD Thesis, Cornell Univ, NY, 1973.

[5] M. H. Chehreghani, H. Abolhassani and M. H. Chehreghani, *Improving density-based methods for hierarchical clustering of web pages*, Data & Knowledge Engineering, **67** (2008), 30–50.

[6] T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, *Introduction to algorithms*, The MIT Press, 2001.

[7] A. P. Dempster, N. M. Laird and D. B. Rubin, *Maximum likelihood from incomplete data via the EM algorithm*, Journal of Royal Statistical Society, Series B, **39** (1977), 1–38.

[8] J. C. Dunn, *A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters*, Journal of Cybernetics, **3(3)** (1973), 32–57.

[9] M. Ester, H. P. Kriegel, J. Sander and X. Xu, *A density-based algorithm for discovering clusters in large spatial databases with noise*, In Proc. 2^{nd} International Conference on Knowledge Discovery and Data Mining, (1996), 226–231.

[10] D. Fisher, *Knowledge acquisition via conceptual clustering*, Machine Learning, **2** (1987), 139–172.

[11] S. Guha, R. Rastogi and K. Shim, *CURE: an efficient clustering algorithms for large databases*, In: Proceeding ACM SIGMOD International Conference on Management of Data, Seattle, WA, (1998), 73–84.

[12] J. Han and M. Kamber, *Data mining concepts and techniques*, Morgan Kaufmann Publishers, San Francisco, CA, 2001.

[13] A. Hinneburg and A. K. Daniel, *An efficient approach to clustering in large multimedia databases with noise*, Proceedings of the 4^{th} Int. Conference on Knowledge Discovery and Data Mining (KDD98), New York, (1998), 58–65.

[14] P. Hore, L. O. Hall, D. B. Goldgof, Y. GU, A. A. Maudsley and A. Darkazanli, *A scalable framework for segmenting magnetic resonance images*, Journal of Signal Processing Systems, **54** (2009), 183–203.

[15] E. Januzaj, H. P. Kriegel and M. Pfeifle, *DBDC: density based distributed clustering*, 5^{th} International Conference on Extending Database Technology (EDBT), Heraklion, Greece, (2004a), 88–105.

[16] E. Januzaj, H. P. Kriegel and M. Pfeifle, *Scalable density based distributed clustering*, 15^{th} International Conference on Machine Learning (ECML) and the 8^{th} European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD), Pisa, Italy, 2004b.

[17] G. Karypis, E. H. Han and V. Kumar, *CHAMELEON: a hierarchical clustering algorithm using dynamic modeling*, IEEE Computer, **32(8)** (1999), 68–75.

[18] L. Kaufman and P. J. Rousseuw, *Finding groups in data: an introduction to cluster analysis*, John Wiley&Sons, Inc, 1990.

[19] E. Mehdizadeh, S. Sadi-Nezhad and R. Tavakkoli-Moghaddam, *Optimization of fuzzy clustering criteria by a hybrid PSO and fuzzy c-means clustering algorithm*, Iranian Journal of Fuzzy Systems, **5(3)** (2008), 1–14.

[20] E. N. Nasibov and G. Ulutagay, *A new approach to clustering problem using the fuzzy joint points method*, Automatic Control and Computer Sciences, **39(6)** (2005), 8–17.

[21] E. N. Nasibov and G. Ulutagay, *On the fuzzy joint points method for fuzzy clustering problem*, Automatic Control and Computer Sciences, **40(5)** (2006), 33–44.

[22] E. N. Nasibov and G. Ulutagay, *A new unsupervised approach for fuzzy clustering*, Fuzzy Sets and Systems, **158(19)** (2007), 2118–2133.

[23] E. N. Nasibov and G. Ulutagay, *Robustness of density-based clustering methods with various neighborhood relations*, Fuzzy Sets and Systems, **160(24)** (2009), 3601–3615.

[24] T. R. Ng and J. Han, *Efficient and effective clustering methods for spatial data mining*, Proceedings of the 20^{th} Very Large Databases Conference (VLDB94), Santiago, Chile, (1994), 144–155.

[25] J. K. Parker, L. O. Hall and A. Kandel, *Scalable fuzzy neighborhood DBSCAN*, IEEE International Conference on Fuzzy Systems, Barcelona, Spain, doi: 10.1109/FUZZY.2010.5584527, (2010), 1–8.

[26] W. Pedrycz and F. Gomide, *An introduction to fuzzy sets*, MIT Press, MA, 1998.

[27] W. Pedrycz, *Distributed and collaborative fuzzy modeling*, Iranian Journal of Fuzzy Systems, **4(1)** (2007), 1–19.
[28] J. Sander, M. Ester, H. P. Kriegel and X. Xu,*Density-based clustering in spatial databases: the algorithm GDBSCAN and its applications*, Data Mining and Knowledge Discovery, **2** (1998), 169–194.
[29] G. Sheikholeslami, S. Chatterjee and A. Zhang, *WaveCluster: a multi-resolution clustering approach for very large spatial databases*, Proceedings of the 24^{th} Very Large Databases Conference (VLDB 98), New York, 1998.
[30] G. Ulutagay and E. Nasibov, *Fuzzy and crisp clustering methods based on the neighborhood concept: a comprehensive review*, Journal of Intelligent and Fuzzy Systems, **23** (2012), 271–281.
[31] W. Wang, Y. Jiong and R. Muntz, *STING: a statistical information grid approach to spatial data mining*, Proceedings of the 23^{rd} Very Large Databases Conference (VLDB 1997), Athens, Greece, 1997.
[32] X. Xiaowei, E. Martin, H. P. Kriegel and J. Sander, *A distribution-based clustering algorithm for mining in large spatial databases*, Proceedings of the 14^{th} Int. Conference on Data Engineering (ICDE98), Orlando, Florida, (1998), 324–331.
[33] A. Z. Xu, J. Chen and J. Wu, *Clustering algorithm for intuitionistic fuzzy sets*, Information Sciences, **178** (2008), 3775–3790.
[34] R. R. Yager and D. P. Filev, *Approximate clustering via the mountain method*, IEEE Transactions on Systems, Man and Cybernetics, **24(8)** (1994) , 1279–1284.
[35] X. L. Yang, Q. Song and Y. L. Wu, *A robust deterministic algorithm for data clustering*, Data & Knowledge Engineering, **62** (2007), 84–100.
[36] T. Zhang, R. Ramakrishnan and M. Livny, *BIRCH: an efficient data clustering method for very large databases*, Proceedings of the 1996 ACM SIGMOD Int. Conference on Management of Data, Montreal, Canada, (1996), 103–113.

Gözde Ulutagay*, Department of Industrial Engineering, Izmir University, Gursel Aksel Blv 14, Uckuyular, Izmir, Turkey
E-mail address: gozde.ulutagay@izmir.edu.tr

Efendi Nasibov, Department of Computer Science, Dokuz Eylul University, Izmir, 35160, Turkey, Institute of Cybernetics, Azerbaijan National Academy of Sciences, Azerbaijan
E-mail address: efendi_nasibov@yahoo.com

*Corresponding author