PUBLICATION OF Little Lion Scientific R&D, Islamabad PAKISTAN
Journal of Theoretical and Applied Information Technology
15th May 2011. Vol. 27 No.1
© 2005 - 2011 JATIT & LLS. All rights reserved.
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

A COMPARATIVE ANALYSIS BETWEEN K-MEDOIDS AND FUZZY C-MEANS CLUSTERING ALGORITHMS FOR STATISTICALLY DISTRIBUTED DATA POINTS

[1]T.VELMURUGAN AND [2]T.SANTHANAM
[1]Associate Professor, [2]Associate Professor & Head,
PG and Research Department of Computer Science, D.G.Vaishnav College,
Arumbakkam, Chennai-600106, India.

ABSTRACT

Data clustering is a process of putting similar data into groups. A clustering algorithm partitions a data set into several groups such that the similarity within a group is larger than among groups. In the field of data mining, various clustering algorithms are proved for their clustering quality. This research work deals with, two of the most representative clustering algorithms namely centroid based K-Medoids and representative object based Fuzzy C-Means are described and analyzed based on their basic approach using the distance between two data points. For both the algorithms, a set of n data points are given in a two-dimensional space and an integer K (the number of clusters) and the problem is to determine a set of n points in the given space called centers, so as to minimize the mean squared distance from each data point to its nearest center. The performance of the algorithms is investigated during different execution of the program for the given input data points. Based on experimental results the algorithms are compared regarding their clustering quality and their performance, which depends on the time complexity between the various numbers of clusters chosen by the end user. The total elapsed time to cluster all the data points and Clustering time for each cluster are also calculated in milliseconds and the results compared with one another.

Key Words: *K-Medoids Algorithm, Fuzzy C-Means Algorithm, Cluster Analysis, Data Analysis.*

1. INTRODUCTION

Clustering is an important area of application for a variety of fields including data mining, knowledge discovery, statistical data analysis, data compression and vector quantization. Clustering has been formulated in various ways in machine learning, pattern recognition, optimization and statistics literature. Clustering is the most common form of unsupervised learning. According to the rule of the unsupervised learning, clustering does not require supervision. No supervision means that there is no human expert who has assigned documents to classes. In clustering, it is the distribution and makeup of the data that will determine cluster membership. The notion of what constitutes a good cluster depends on the application and there are many methods for finding clusters subject to various criteria. These include approaches based on splitting and merging such as ISODATA, randomized approaches such as CLARA, CLARANS, and methods based on neural nets, and methods designed to scale to large databases, including DBSCAN, BIRCH and ScaleKM [2][4][10]. Among clustering formulations that are based on minimizing a formal objective function, perhaps the most widely used and studied is partition based algorithms like K-Means, K-Medoids and Fuzzy C-Means clustering. In a partitioned algorithm, given a set of n data points in real d-dimensional space, and an integer k, the problem is to determine a set of k points in R^d, called centers, so as to minimize the mean squared distance from each data point to its nearest center. This measure is often called the squared-error distortion and this type of clustering falls into the general category of variance based clustering [5][8][9].

A simple example of cluster is given in Fig. 1. It is visually clear that there are three distinct clusters of 50 data points. The key input to a clustering algorithm is the distance measure.

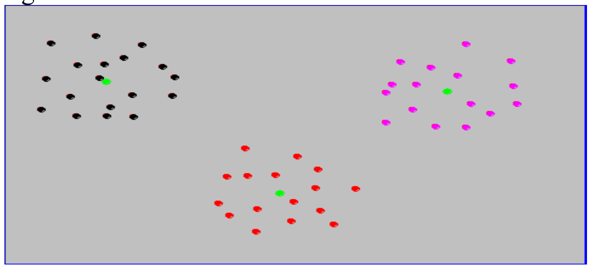


Fig. 1: Cluster Example

In Fig. 1, the distance measure is distance in the 2D plane. The cluster center is displayed in green color and the data points of each cluster are displayed in different colors. This measure suggests three different clusters in the figure. In most of the clustering approaches, the distance measure used is the Euclidean distance [6]. Different distance measures give rise to different clustering. Thus, the distance measure is an important means by which this research can influence the outcome of clustering.

2. METHODOLOGY

An important step in most clustering is to select a distance measure, which will determine how the similarity of two elements is calculated. This will influence the shape of the clusters, as some elements may be close to one another according to one distance and farther away according to another. For example, in a 2-dimensional space, the distance between the point ($x = 1$, $y = 0$) and the origin ($x = 0$, $y = 0$) is always 1 according to the usual norms, but the distance between the point ($x = 1$, $y = 1$) and the origin can be 2, $\sqrt{2}$ or 1 if you take respectively the 1-norm, 2-norm or infinity-norm distance. The variety of common distance functions are as follows:

- The Euclidean distance. This type of distance is also called as the distance as the crow flies or 2-norm distance).
- The Manhattan distance (aka taxicab norm or 1-norm)
- The maximum norm (aka infinity norm)
- The Mahalanobis distance corrects data for different scales and correlations in the variables.
- The angle between two vectors can be used as a distance measure when clustering high dimensional data.
- The Hamming distance measures the minimum number of substitutions required to change one member into another.

Another important distinction is whether the clustering uses symmetric or asymmetric distances. Many of the distance functions listed above have the property that distances are symmetric (the distance from object A to B is the same as the distance from B to A). In other applications, this is not the case. (A true metric gives symmetric measures of distance.) The symmetric and 2-norm distance measure is used in this research work. In the Euclidean space $\mathbf{R}^n$, the distance between two points is usually given by the Euclidean distance (2-norm distance). The formula for 2-norm distance is

$$\text{2-norm distance} = \left(\sum_{i=1}^{n} \left| x_i - y_i \right|^2 \right)^{1/2}$$

The 2-norm distance is the Euclidean distance, a generalization of the Pythagorean Theorem to more than two coordinates. It is what would be obtained if the distance between two points were measured with a ruler: the "intuitive" idea of distance. Based on this idea of finding the distance between data points, the clustering qualities of the proposed algorithms are analyzed in this work.

Determining the quality of a clustering algorithm involves evaluating and assessing the quality of the clusters produced and is an important task in data mining. There are three approaches to measuring cluster quality, based on external, relative and internal criteria. The term external validity criteria are used when the results of the clustering algorithm can be compared with some pre-specified clustering structures (Halkidi et al., 2002). Relative validity criteria measure the quality of clustering results by comparing them with others generated by other clustering algorithms, or by the same algorithm using different parameters [6][17][18]. An internal validity criterion involve the development of functions that compute the distances between objects within each cluster, or the distance between the clusters themselves, and uses such distances to assess the clustering quality. To achieve a good clustering, these criteria are in the form of measures to assess the quality of a clustering. This work uses only the internal validity criteria in a random way. The Euclidean distance is used to find the distance between the input data points.

To create input data points in the applet window, the Box-Muller formula is used. For computer simulations, it is often useful to generate values that have a normal distribution. There are several methods and the most basic is to invert the standard normal cumulative distribution function. One of the efficient methods is the Box–Muller transform. A simple approximate approach that is easy to program is as follows. Simply sum 12 uniform (0, 1) deviates and subtract 6 (half of 12). This is quite usable in many applications. The sum over these 12 values has an Irwin–Hall distribution; 12 are chosen to give the sum a variance of exactly one. The resulting random deviates are limited to the range (−6, 6) and have a density which is a 12-section eleventh-order polynomial approximation to the normal distribution. Box–Muller method is used for normal distribution coding to calculate the data points X and Y. This method says that, if two independent random numbers U and V are uniformly distributed on (0, 1], (e.g. the output from a random number generator), then two independent standard normally distributed random variables are X and Y, where

$$X = \sqrt{-2\ln U}\cos(2\pi V) \tag{1}$$

$$Y = \sqrt{-2\ln U}\sin(2\pi V) \tag{2}$$

This formulation arises because the chi-square distribution with two degrees of freedom is an easily-generated exponential random variable (which corresponds to the quantity of natural logarithm of U in these equations). Thus an angle is chosen uniformly around the circle via the random variable V, a radius is chosen to be exponential and then transformed to (normally distributed) x and y coordinates. The basic form of the Box-Muller method is as follows. Suppose U_1 and U_2 are independent random variables that are uniformly distributed in the interval (0, 1]. Let

$$Z_0 = R\cos(\theta) = \sqrt{-2\ln U_1}\cos(2\pi U_2) \tag{3}$$

$$Z_1 = R\sin(\theta) = \sqrt{-2\ln U_1}\sin(2\pi U_2) \tag{4}$$

Then Z_0 and Z_1 are independent random variables with a normal distribution of standard deviation 1. The derivation is based on the fact that, in a two-dimensional cartesian system where X and Y coordinates are described by two independent and normally distributed random variables, the random variables for R^2 and Θ in the corresponding polar coordinates are also independent and can be expressed as

$$R^2 = -2.\ln U_1 \text{ and } \theta = 2\pi U_2 \tag{5}$$

Because R^2 is the square of the norm of the standard bivariate normal variable (X, Y), it has the chi-square distribution with two degrees of freedom. In the special case of two degrees of freedom, the chi-square distribution coincides with the exponential distribution, and the equation for R^2 above is a simple way of generating the required exponential variate. The next sections deal with the basic concepts of K-Medoids and Fuzzy C-Means algorithm followed by the experimental results.

3. K-MEDOIDS ALGORITHM

The very popular K-Means algorithm is sensitive to outliers since an object with an extremely large value may substantially distort the distribution of data. How might the algorithm be modified to diminish such sensitivity? Instead of taking the mean value of the objects in a cluster as a reference point, a Medoid can be used, which is the most centrally located object in a cluster [14][15]. Thus the partitioning method can still be performed based on the principle of minimizing the sum of the dissimilarities between each object and its corresponding reference point. This forms the basis of the K-Medoids method (Han and Kamber, 2006; Jain, et al., 1999; Kaufman and Rousseeuw, 1990). The basic strategy of K-Medoids clustering algorithms is to find k clusters in n objects by first arbitrarily finding a representative object (the Medoids) for each cluster. Each remaining object is clustered with the medoid to which it is the most similar. K-Medoids method uses representative objects as reference points instead of taking the mean value of the objects in each cluster. The algorithm takes the input parameter k, the number of clusters to be partitioned among a set of n objects (Park, et al., 2009; Dunham, 2003; Han and Kamber, 2006). A typical K-Mediods algorithm for partitioning based on Medoid or central objects is as follows:

```
Input:   k: The number of clusters
         D: A data set containing n objects
Output: A set of k clusters that minimizes the
          sum of the dissimilarities of all the
          objects to their nearest medoid.
Method: Arbitrarily choose k objects in D as the
          initial representative objects;
Repeat   assign each remaining object to the
          cluster with the nearest medoid;
         randomly select a non medoid object
          $O_{random}$;
         compute the total points S of swaping
          object $O_j$ with $O_{ramdom;}$
         if S < 0 then swap $O_j$ with $O_{random}$ to
          form the new set of k medoid;
Until no change;
```

Similar to the above algorithm, a Partitioning Around Medoids (PAM) was one of the first k-Medoids algorithms introduced by researchers. It attempts to determine k partitions for n objects. After an initial random selection of k medoids, the algorithm repeatedly tries to make a better choice of medoids [16][21]. Therefore, the algorithm is often called as representative object based algorithm.

4. FUZZY C-MEANS ALGORITHM

Traditional clustering approaches generate partitions; in a partition, each pattern belongs to one and only one cluster. Fuzzy clustering extends this notion to associate each pattern with every cluster using a membership function [1][7]. The output of such algorithms is a clustering, but not a partition some times. Fuzzy clustering is a widely applied method for obtaining fuzzy models from data. It has been applied successfully in various fields including geographical surveying, finance or marketing. This method is frequently used in pattern recognition. It is based on minimization of the following objective function:

$$J_m = \sum_{i-1}^{N} \sum_{j-1}^{C} u_{ij}^m \left\| x_i - c_j \right\|^2 \quad , 1 \le \text{m} < \infty$$

where m is any real number greater than 1, u_{ij} is the degree of membership of x_i in the cluster j, x_i is the i^{th} of d-dimensional measured data, c_j is the d-dimension center of the cluster, and ||*|| is any norm expressing the similarity between any measured data and the center (Al-Zoubi et al., 2007; Yong et al., 2004). Fuzzy partitioning is carried out through an iterative optimization of the objective function shown above, with the update of membership u_{ij} and the cluster centers c_j by:

$$u_{ij} = \frac{1}{\sum_{k=1}^{c} \left[\frac{\left\| x_i - c_j \right\|}{\left\| x_i - c_k \right\|} \right]^{\frac{2}{m-1}}} , \qquad c_j = \frac{\sum_{i=1}^{N} u_{ij}^m . x_i}{\sum_{i=1}^{N} u_{ij}^m}$$

This iteration will stop when $\max_{ij} \left\{ \left| u_{ij}^{(k+1)} - u_{ij}^{(k)} \right| \right\} < \xi$, where ξ is a termination criterion between 0 and 1, whereas k are the iteration steps. This procedure converges to a local minimum or a saddle point of J_m. The algorithm is composed of the following steps:

Step 1: Initialize U=[u_{ij}] matrix, $U^{(0)}$
Step 2: At k-step: calculate the centers vectors $C^{(k)}$=[c_j] with $U^{(k)}$
Step 3: Update $U^{(k)}$, $U^{(k+1)}$
Step 4: If || $U^{(k+1)}$ - $U^{(k)}$||< ξ then STOP; otherwise return to step 2.

In this algorithm, data are bound to each cluster by means of a Membership Function, which represents the fuzzy behavior of the algorithm. To do that, the algorithm has to build an appropriate matrix named U whose factors are numbers between 0 and 1, and represent the degree of membership between data and centers of clusters (Moh'd Belal Al- Zoubi, ed al., 2007). FCM clustering techniques are based on fuzzy behavior and provide a natural technique for producing a clustering where membership weights have a natural (but not probabilistic) interpretation. This algorithm is similar in structure to the K-Means algorithm and also behaves in a similar way [12][23[24]. For the creation of input data points for both the algorithms, the Box-Muller formula is used [3]. With this discussion, the next part of this paper analyses the experimental results and discussion about the results.

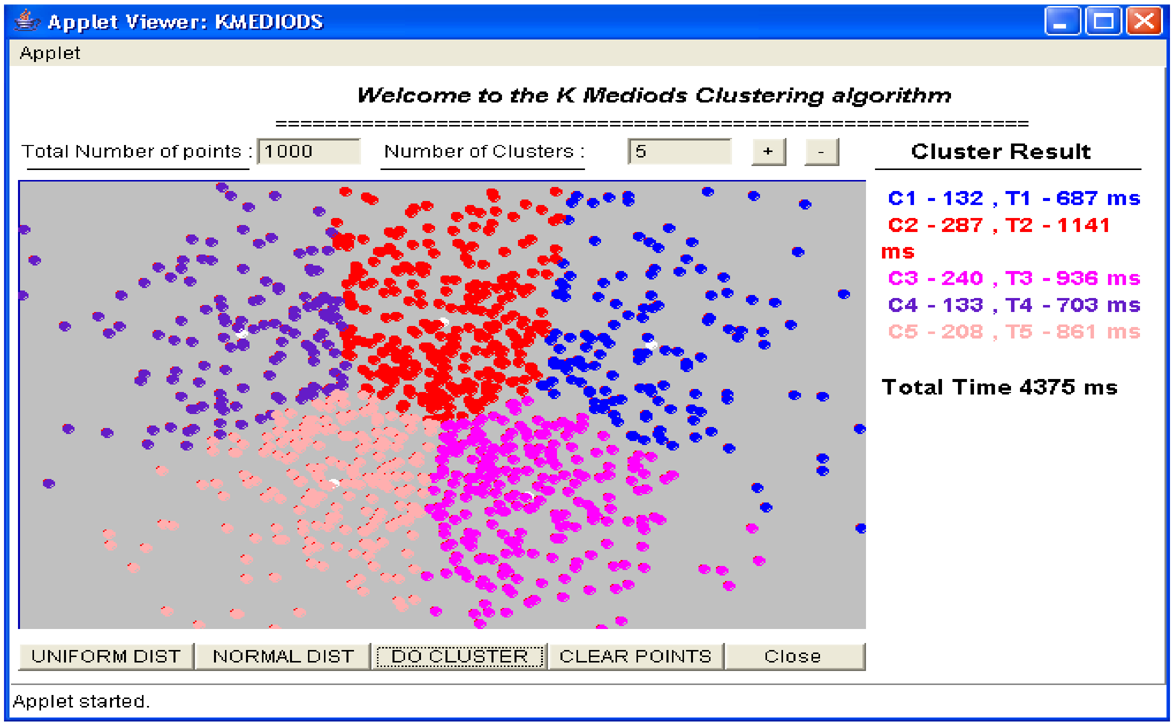
5. EXPERIMENTAL RESULTS

Having introduced the two clustering techniques and their basic mathematical foundations, now turn to the discussion of these techniques on the basis of a practical approach. This approach involves the implementation of the two techniques introduced previously, and performance of the algorithms is tested on statistical distributions of input data points via Normal and Uniform distributions. Therefore, the implementation plan consists of two parts; one for Normal distribution of input data points and the other for Uniform distribution. For both the types of distributions, the data points are created using Box-Muller formula as discussed [3]. The program is written in JAVA language. After the creation of data points, the user has to specify the number of clusters. In the output window, the data points in each cluster are displayed by different colors and the execution time is calculated in milliseconds [19][20][21]. Now, the discussion starts with the Normal distribution of input data points first followed by Uniform distribution. The resulting clusters of the Normal distribution for K-Medoids algorithm is given in Fig. 2. The number of data points and clusters given by end user as input during the execution of the program is 1000 and 5 (k = 5) respectively. The algorithm is repeated 1000 times (one iteration for each data point) to get efficient output. The cluster centers (centroids) are calculated for each cluster by its mean value and clusters are formed depending upon the distance between data points.

Different approaches of clustering yields different types of outcomes. In this work, for different input data points, the algorithm gives different types of outputs. The input data points are

PUBLICATION OF Little Lion Scientific R&D, Islamabad PAKISTAN
Journal of Theoretical and Applied Information Technology
15th May 2011. Vol. 27 No.1
© 2005 - 2011 JATIT & LLS. All rights reserved
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

generated in red color and the output of the algorithm is displayed in different colors as shown in Fig. 2. The center point of each cluster is displayed in white color. The total elapsed time and the execution time for each cluster to each run are calculated in milliseconds. The time taken for execution of the algorithm varies from one run to another run and also it differs from one computer to another computer. The number of data points is the size of the cluster. If the number of data points are 1000 then the algorithm is repeated the same one thousand times. For each data point, the algorithm executes once. The algorithm takes 4375 milliseconds for 1000 data points and 5 clusters. The same algorithm is executed 5 times and results are tabulated in table 1 (against the rows 'Normal'). The total elapsed time for all clusters is given at the end of the row 'Size'. The sum of the execution time for each cluster (individual execution time for each cluster) is given at the end of the row ICT (Individual Cluster Time). The difference between these two is available in the last column. For the same one thousand uniformly distributed data points, the algorithm is executed 5 times and the results are listed in table 1 (against the rows 'Uniform'). Next, one of the outcomes for 1000 uniformly distributed data points and 10 clusters is shown in Fig. 3. In this case, the algorithm takes 7344 milliseconds for 10 clusters. But the sum of the individual clustering time is 7313 milliseconds. The difference time between these two is 31 milliseconds.



Welcome to the K Mediods Clustering algorithm

Total Number of points : 1000 Number of Clusters : 5

Cluster Result

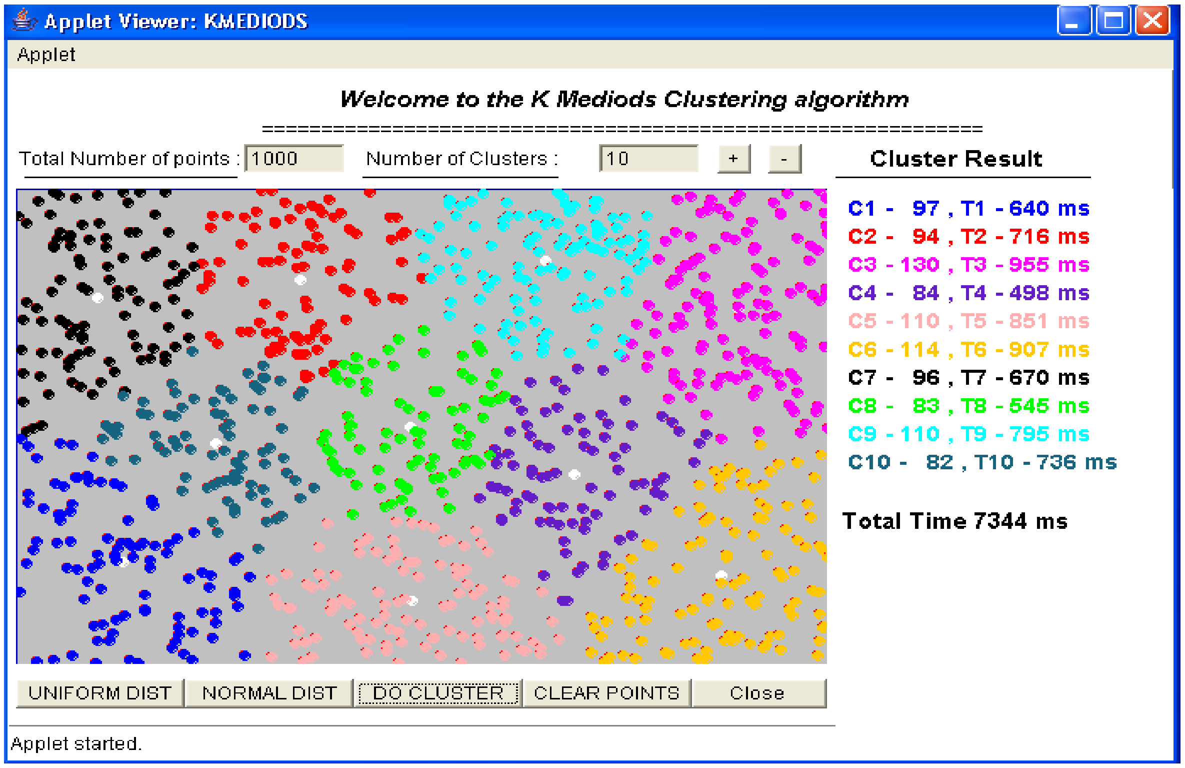
C1 - 132 , T1 - 687 ms
C2 - 287 , T2 - 1141 ms
C3 - 240 , T3 - 936 ms
C4 - 133 , T4 - 703 ms
C5 - 208 , T5 - 861 ms

Total Time 4375 ms

Applet started.

Fig. 2: Normal Distribution Output- K-Medoids

PUBLICATION OF Little Lion Scientific R&D, Islamabad PAKISTAN
Journal of Theoretical and Applied Information Technology
15th May 2011. Vol. 27 No.1
© 2005 - 2011 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195



Welcome to the K Mediods Clustering algorithm

Total Number of points : 1000 Number of Clusters : 10

Cluster Result

C1 - 97 , T1 - 640 ms
C2 - 94 , T2 - 716 ms
C3 - 130 , T3 - 955 ms
C4 - 84 , T4 - 498 ms
C5 - 110 , T5 - 851 ms
C6 - 114 , T6 - 907 ms
C7 - 96 , T7 - 670 ms
C8 - 83 , T8 - 545 ms
C9 - 110 , T9 - 795 ms
C10 - 82 , T10 - 736 ms

Total Time 7344 ms

Applet started.

Fig. 3: Uniform Distribution Output-K-Medoids

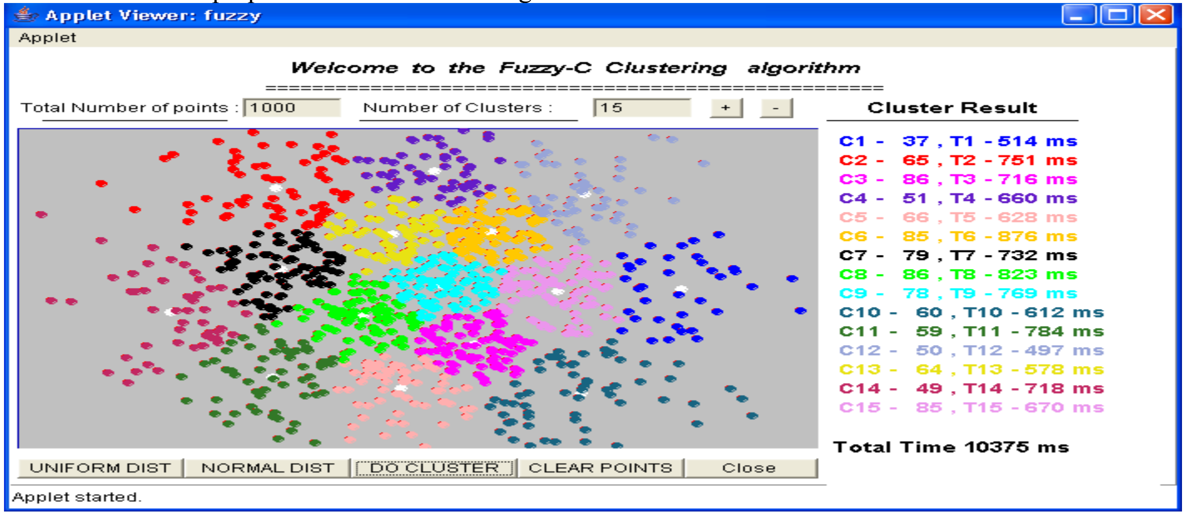
Next, the FCM algorithm is executed by taking 1000 Normal distribution of data points as input. The number of clusters chosen by the user is 5. The execution time taken by the system is 3953 milliseconds. Like, the algorithm is executed for one thousand uniformly distributed data points. Now the algorithm takes 3797 milliseconds for execution. Five times the algorithm is executed and the results are given in table 2 for both the types of inputs. Here also the difference time between total elapsed time and the sum of the execution time for all clusters is given in the last row. Next, both the algorithms are executed 5 times by considering the number of clusters is 10 and the results are given in the table 3. This table contains only the execution time for each cluster and the total elapsed time. The third column is the type of the distribution. Clustering time for Normal distribution is listed against the rows 'N' and for the Uniform distribution is listed against the rows 'U'. The execution time for each cluster is given in the table from the columns 1 to 10 for ten clusters. The sum of each cluster individual time is available in the column ICT (Individual Cluster Time). The total elapsed time is indicated in the column TET (Total Elapsed Time). The difference time (DT) between ICT and TET is calculated and is given in the last column. Fig. 4 is the output for 1000 normally distributed data points with 15 clusters of FCM algorithm. For the same algorithm, the number of clusters chosen by the user is 20 and the result of 1000 uniformly distributed data points is shown in Fig. 5. Table 4 shows that the performance results comparison for all the types of data distributed for the experiments for 5 and 10 clusters. Also the table contains the best and worst execution times for all the categories. From the table, it is clear that the best times (BT) are derived from the result of uniform distribution and the worst times (WT) are from normal distribution. The average times of ICT and TET are given in the same table for both the algorithms for the normal and uniform distributions of input data points. The difference time between ICT and TET are calculated and is given in the columns DT1 and DT2 for 5 and 10 clusters respectively. Next, both the algorithms are executed by choosing the number of clusters 15, 20 and the average times are listed in the table 5. Here ICT and TET times alone taken for discussion.

PUBLICATION OF Little Lion Scientific R&D, Islamabad PAKISTAN
Journal of Theoretical and Applied Information Technology
15th May 2011. Vol. 27 No.1
© 2005 - 2011 JATIT & LLS. All rights reserved
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 1: Cluster Results for K-Medoids

Run	Distribution	Cluster	1	2	3	4	5	T.Time	Diff.Time
1	Normal	Size	132	287	240	133	208	4375	47
		ICT	687	1141	936	703	861	4328	
	Uniform	Size	198	230	199	200	173	4062	24
		ICT	889	759	835	815	740	4038	
2	Normal	Size	237	217	163	204	179	4203	30
		ICT	902	870	750	865	786	4173	
	Uniform	Size	162	158	254	189	237	3891	16
		ICT	687	656	866	737	929	3875	
3	Normal	Size	167	248	241	127	217	4090	42
		ICT	753	887	998	649	761	4048	
	Uniform	Size	216	165	171	190	258	4172	15
		ICT	841	654	717	861	1084	4157	
4	Normal	Size	200	252	195	219	134	4059	44
		ICT	896	943	700	787	689	4015	
	Uniform	Size	187	179	201	212	221	3994	24
		ICT	803	710	790	856	811	3970	
5	Normal	Size	133	202	194	217	254	4210	33
		ICT	720	856	798	813	990	4177	
	Uniform	Size	172	174	239	215	200	3944	27
		ICT	785	781	811	790	750	3917	

The programs of K-Medoids and FCM are executed many times and the results are analyzed based on the number of data points and the number of clusters. The behavior of the algorithms is analyzed by observations. The performance of the algorithms have also been analyzed for several executions by considering different data points (for which the results are not shown) as input (1500 data points, 2000 data points etc.) and the number of clusters are from 15 and 20 (for which also the results are not shown), the outcomes are found to be highly satisfactory [20][21]. According to the results obtained from the proposed two data clustering algorithms, the results are compared based on their performance and clustering quality. A table of the achieved results by giving average times for each of the two techniques is presented in Table 5 and the graphical format is shown in Fig. 6. By comparing the average times of both the algorithms, it can easily observe that the FCM algorithm takes less time than the K-Medoids algorithm. For example, the average time for 15 clusters for normal distribution of data points is 10415.6 milliseconds, but for the FCM algorithm it is found to be 10305.6 milliseconds. This same type of results repeats for uniform distribution also.



Cluster Result

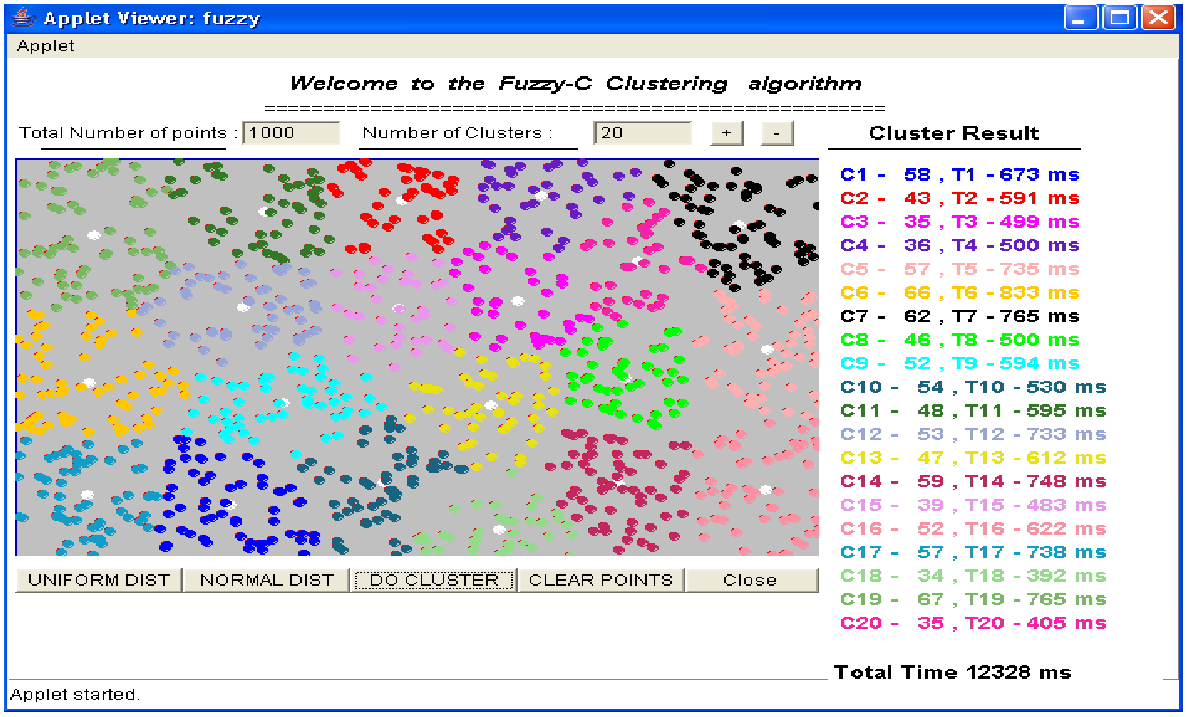
```
C1 -  37 , T1 - 514 ms
C2 -  65 , T2 - 751 ms
C3 -  86 , T3 - 716 ms
C4 -  51 , T4 - 660 ms
C5 -  66 , T5 - 628 ms
C6 -  85 , T6 - 876 ms
C7 -  79 , T7 - 732 ms
C8 -  86 , T8 - 823 ms
C9 -  78 , T9 - 769 ms
C10 -  60 , T10 - 612 ms
C11 -  59 , T11 - 784 ms
C12 -  50 , T12 - 497 ms
C13 -  64 , T13 - 578 ms
C14 -  49 , T14 - 718 ms
C15 -  85 , T15 - 670 ms

Total Time 10375 ms
```

Fig. 4: Normal Distribution Output- Fuzzy C-Means

PUBLICATION OF Little Lion Scientific R&D, Islamabad PAKISTAN
Journal of Theoretical and Applied Information Technology
15th May 2011. Vol. 27 No.1
© 2005 - 2011 JATIT & LLS. All rights reserved
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 2: Cluster results for FCM

Run	Distribution	Cluster	1	2	3	4	5	T.Time	Diff.Time
1	Normal	Size	269	207	152	235	137	3953	16
		ICT	951	858	768	769	591	3937	
	Uniform	Size	169	213	252	179	187	3797	16
		ICT	767	737	905	653	719	3781	
2	Normal	Size	160	162	201	211	266	4015	28
		ICT	723	705	781	845	933	3987	
	Uniform	Size	218	189	199	192	202	3818	15
		ICT	840	689	810	702	762	3803	
3	Normal	Size	197	223	216	186	178	3943	31
		ICT	742	842	748	791	789	3912	
	Uniform	Size	216	236	206	169	173	3931	18
		ICT	821	872	755	655	810	3913	
4	Normal	Size	214	237	162	176	211	4244	25
		ICT	838	970	813	704	894	4219	
	Uniform	Size	228	193	178	172	229	4024	23
		ICT	852	775	750	740	884	4001	
5	Normal	Size	213	157	178	200	252	4125	26
		ICT	889	712	788	809	901	4099	
	Uniform	Size	199	163	244	194	200	4022	24
		ICT	815	775	885	710	813	3998	



Welcome to the Fuzzy-C Clustering algorithm

Total Number of points : 1000 Number of Clusters : 20

Cluster Result

C1 - 58 , T1 - 673 ms
C2 - 43 , T2 - 591 ms
C3 - 35 , T3 - 499 ms
C4 - 36 , T4 - 500 ms
C5 - 57 , T5 - 735 ms
C6 - 66 , T6 - 833 ms
C7 - 62 , T7 - 765 ms
C8 - 46 , T8 - 500 ms
C9 - 52 , T9 - 594 ms
C10 - 54 , T10 - 530 ms
C11 - 48 , T11 - 595 ms
C12 - 53 , T12 - 733 ms
C13 - 47 , T13 - 612 ms
C14 - 59 , T14 - 748 ms
C15 - 39 , T15 - 483 ms
C16 - 52 , T16 - 622 ms
C17 - 57 , T17 - 738 ms
C18 - 34 , T18 - 392 ms
C19 - 67 , T19 - 765 ms
C20 - 35 , T20 - 405 ms

Total Time 12328 ms

Applet started.

Fig. 5: Uniform Distribution Output- Fuzzy C-Means

Table 3: Cluster Results for 10 Clusters

Algorithm	R	Dist\Clr	1	2	3	4	5	6	7	8	9	10	ICT	TET	DT
K-Medoids	1	N	735	865	655	636	670	640	531	892	952	891	7467	7501	34
		U	640	716	955	498	851	907	670	545	795	736	7313	7344	31
	2	N	693	625	876	655	889	843	703	645	655	691	7275	7321	46
		U	930	718	788	815	656	768	682	548	795	565	7265	7289	24
	3	N	699	955	598	687	749	602	718	891	689	755	7343	7378	35
		U	844	589	676	562	1004	579	954	613	698	626	7145	7177	32
	4	N	817	656	748	731	861	734	735	889	656	563	7390	7422	32
		U	691	862	704	782	578	684	899	733	592	569	7094	7132	38
	5	N	723	624	717	855	718	807	579	619	627	1071	7340	7368	28
		U	674	705	832	857	685	668	935	723	532	676	7287	7313	26
FCM	1	N	718	576	580	802	466	766	950	750	1000	846	7454	7500	46
		U	906	545	656	672	924	844	577	685	656	722	7187	7208	21
	2	N	754	721	545	808	566	778	732	767	656	860	7187	7202	15
		U	595	797	842	674	907	565	561	700	595	858	7094	7125	31
	3	N	671	652	714	565	794	975	867	722	621	661	7242	7265	23
		U	787	643	778	761	689	688	643	783	747	562	7081	7101	20
	4	N	640	805	513	630	857	548	644	1030	653	893	7213	7243	30
		U	789	596	714	705	760	676	611	673	699	887	7110	7135	25
	5	N	648	700	704	733	608	626	688	876	657	816	7056	7077	21
		U	514	764	706	639	828	672	623	610	724	982	7062	7085	23

PUBLICATION OF Little Lion Scientific R&D, Islamabad PAKISTAN
Journal of Theoretical and Applied Information Technology
15th May 2011. Vol. 27 No.1
© 2005 - 2011 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 4: Results Comparison (5 and 10 Clusters)

Algorithm	Distribution	5 Clusters					10 Clusters				
		BT	WT	ICT	TET	DT 1	BT	WT	ICT	TET	DT 2
K-Medoids	Normal	4059	4375	4148.2	4187.4	39.2	7145	7501	7363	7398	35
	Uniform	3891	4172	3991.4	4012.6	21.2	7015	7342	7220.8	7251	30.2
FCM	Normal	3943	4244	4030.8	4056	25.2	7195	7500	7230.4	7257.4	27
	Uniform	3797	4024	3899.2	3918.4	19.2	7085	7250	7106.8	7130.8	24

Table 5: Performance Results Comparison

Algorithm	Dist.	5 Clusters		10 Clusters		15 Clusters		20 Clusters	
		ICT	TET	ICT	TET	ICT	TET	ICT	TET
K-Medoids	Normal	4148.2	4187.4	7363	7398	10385.6	10415.6	12770.4	12795.6
	Uniform	3991.4	4012.6	7220.8	7251	10289.3	10314.2	12635.5	12656.6
FCM	Normal	4030.8	4056	7230.4	7257.4	10276.2	10305.6	12625.2	12652.8
	Uniform	3899.2	3918.4	7106.8	7130.8	10184.1	10208.2	12510	12538.8

From Fig. 6, it is easy to identify the difference between the execution times for both the algorithms. Here, the total execution time alone taken for comparison. The individual clustering time is not taken. Since the partitioning based clustering algorithms behave in a similar way, the difference between the execution time is less significant.

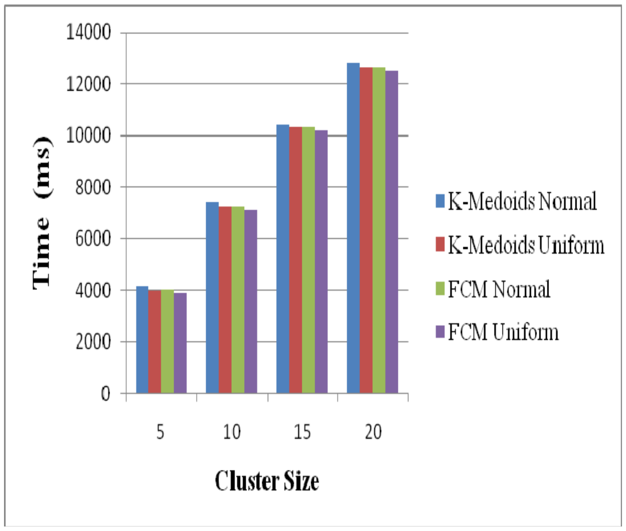


Fig. 6: Cluster Results

6. CONCLUSION

Cluster analysis is sensitive to both the distance metric selected and the criterion for determining the order of clustering. Different approaches may yield different results. The choice of clustering algorithm depends on both the type of data available and on the particular purpose and application. It is very evident from table 5 that the performance of the FCM algorithm is relatively better than that of K-Medoids algorithm. Thus, for the data points generated using statistical distributions (via Normal and Uniform distributions), the FCM algorithm seems to be superior to K-Medoids. However, between the distributions Uniform distribution yields the best results. Generally, when the number of clusters increases, in turn will increase the execution time for both the algorithms. Hence, the time taken for execution depends on the number of clusters and the number of data points chosen by the user.

REFERENCES

[1] Al-Zoubi, M.B., A. Hudaib and B. Al-Shboul, A fast fuzzy clustering algorithm. Proceedings of the 6th WSEAS Int. Conf. on Artificial Intelligence, Knowledge Engineering and Data Bases, February 2007, pp. 28-32.

[2] Berkhin, P., 2002, Survey of Clustering Data Mining Techniques; Accrue Software, Inc, USA, 2002. www.ee.ucr.edu/~barth/EE242/clustering_survey.pdf

[3] Box G. E. P. and Mervin E. Muller, A Note on the Generation of Random Normal Deviates, The Annals of Mathematical Statistics (1958), Vol. 29, No. 2, pp. 610–611.

[4] Davies.D.L and D.W. Bouldin, A cluster separation measure, IEEE Trans. Pattern Anal. Machine Intell. Vol. 1, 2009, pp. 224-227, ISSN: 0162-8828. DOI: 10.1109/TPAMI.1979.4766909

[5] Han J. and M. Kamber, Data Mining: Concepts and Techniques, Morgan Kaufmann Publishers, Second Edition, New Delhi, 2006. ISBN: 978-81-312-0535-8

[6] Holmes Finch, Comparison of Distance Measures in Cluster Analysis with Dichotomous Data, Journal of Data Science 3(2005), pp. 85-100, ISSN:1680-743X, online: ISSN 1683-8602. http://www.jds-online.com/file_download/66/JDS-192.pdf

[7] Indrajit Saha, and Anirban Mukhopadhyay, An Improved Crisp and Fuzzy based Clustering Technique for Categorical Data, International Journal of Computer Science and Engineering, 2008, pp 184-193. http://www.waset.ac.nz/journals/ijcse/v2/v2-4-33.pdf.

[8] Jain, A.K. and R.C. Dubes, Algorithms for Clustering Data. Prentice Hall Inc., Englewood Cliffs, New Jerssy, 1988, ISBN: 0-13-022278-X

[9] Jain, A.K., M.N. Murty and P.J. Flynn, Data Clustering: A Review, ACM Computing Surveys, Vol. 31, No. 3, Sep. 1999, pp. 264-323, DOI:10.1.1.18.2720&rep=rep1&type=pdf

[10] Kaufman, L. and P.J. Rousseeuw, Finding Groups in Data: an Introduction to Cluster Analysis, John Wiley and Sons, 1990.

[11] Maria Camila N. Barioni, Humberto L. Razente, Agma J. M. Traina, Caetano Traina Jr, An efficient approach to scale up k-medoid based algorithms in large databases, 2006. http://www.lbd.dcc.ufmg.br:8080/colecoes/sbbd/2006/018.pdf

[12] Marta V. Modenesi, Myrian C. A. Costa, Alexandre G. Evsukoff,,and Nelson F. F.Ebecken, Parallel Fuzzy C-Means Cluster Analysis, High Performance Computing for Computational Science - VECPAR 2006, Lecture Notes in Computer Science, 2007, Volume 4395/2007, pp. 52-65, DOI: 10.1007/978-3-540-71351-7_5

[13] Olivier Ferret, Brigitte grau and Mich`ele Jardino, A cross-comparison of two clustering methods, 2001. http://www.ldc.upenn.edu/acl/W/W01/W01-0909.pdf.

[14] Park, H.S., J.S. Lee and C.H. Jun, A K-Means-Like Algorithm for K-Medoids Clustering and Its Performance, Department of Industrial and Management Engineering, POSTECH, South Korea, 2009. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.90.7981&rep=rep1&type=pdf

[15] Raghu Krishnapuram, Anupam Joshi and Liyu Yi, A Fuzzy Relative of the k-Medoids Algorithm with Application to Web Document and Snippet Clustering, The IEEE International Fuzzy Systems Conference, FUZZ-IEEE'99; Seoul; South Korea; 22 Aug.-25 Aug. 1999, pp. III-1281-III-1286.

[16] Reynolds. A.P, G. Richards, and V. J. Rayward-Smith, The Application of K-medoids and PAM to the Clustering of Rules, Lecture notes in computer science, 2004, ISSN: 0302-9743. http://www.macs.hw.ac.uk/~ar136/publications/clusteringConference.pdf

[17] Romesburg, H. Clarles, Cluster Analysis for Researchers, 2004, pp. 340, ISBN 1-4116-0617-5.www.lulu.com/items/volume_1/46000/46479/.../CLUSPreview.pdf

[18] Tibshirani, R. Walther. G. Hastie.T, Estimating the number of clusters in a data set via the gap statistic, Journal of Royal Statistical SocICTy Series B Statistical Methodalogy, 2001, Vol 63; Part 2, pp. 411-423, ISSN:1369-7412.

[19] Velmurugan. T and T.Santhanam, A Practical Approach of K-Medoids Clustering Algorithm for Artificial data points, Proceedings of the Int. Conf. on Semantics, E-business and E-Commerce, Trichirappalli, India, Nov. 2009, pp. 45-50, ISBN: 978-81-907337-0-0.

[20] Velmurugan. T and T. Santhanam, Computational complexity between K-means and K-medoids clustering algorithms for normal and uniform distributions of data points. Journal of Computer Science, 6 (3): 2010, pp. 363-368, ISSN:1549-3636.

[21] Velmurugan.T and T.Santhanam, A Survey of Partition Based Clustering Algorithms in Data Mining: An Experimental Approach, Information Technology Journal, 10 (3): 2011, pp. 478-484, ISSN: 1812-5638/DOI: 10.3923/itj-2011.478-484.

[22] Weiguo Sheng and Xiaohui Liu, A genetic *k*-medoids clustering algorithm, J Heuristics, Vol. 12, No. 6, 2006, pp. 447–466, DOI: 10.1007/s10732-006-7284-z

[23] Yong, Y., Z. Chongxun and L. Pan, A novel fuzzy c-means clustering algorithm for image thresholding. Measurement Sci. Rev., Volume 4: 9(1), 2004.

[24] Yong, Y., Z. Chongxun and L. Pan, Fuzzy C-means clustering algorithm with a novel penalty term for image segmentation, Opto-Electronics Review 13(4), 2005, pp. 309-315.