Journal of Computer Science 6 (3): 363-368, 2010
ISSN 1549-3636

Computational Complexity between K-Means and K-Medoids Clustering Algorithms for Normal and Uniform Distributions of Data Points

T. Velmurugan and T. Santhanam
Department of Computer Science, DG Vaishnav College, Chennai, India

Abstract: Problem statement: Clustering is one of the most important research areas in the field of data mining. Clustering means creating groups of objects based on their features in such a way that the objects belonging to the same groups are similar and those belonging to different groups are dissimilar. Clustering is an unsupervised learning technique. The main advantage of clustering is that interesting patterns and structures can be found directly from very large data sets with little or none of the background knowledge. Clustering algorithms can be applied in many domains. **Approach:** In this research, the most representative algorithms K-Means and K-Medoids were examined and analyzed based on their basic approach. The best algorithm in each category was found out based on their performance. The input data points are generated by two ways, one by using normal distribution and another by applying uniform distribution. **Results:** The randomly distributed data points were taken as input to these algorithms and clusters are found out for each algorithm. The algorithms were implemented using JAVA language and the performance was analyzed based on their clustering quality. The execution time for the algorithms in each category was compared for different runs. The accuracy of the algorithm was investigated during different execution of the program on the input data points. **Conclusion:** The average time taken by K-Means algorithm is greater than the time taken by K-Medoids algorithm for both the case of normal and uniform distributions. The results proved to be satisfactory.

Key words: K-Means clustering, K-Medoids clustering, data clustering, cluster analysis

INTRODUCTION

Clustering can be considered the most important unsupervised learning problem; so, as every other problem of this kind, it deals with finding a structure in a collection of unlabeled data (Jain and Dubes, 1988; Jain *et al*., 1999). A loose definition of clustering could be "the process of organizing objects into groups whose members are similar in some way". A cluster is therefore a collection of objects which are "similar" between them and are "dissimilar" to the objects belonging to other clusters. Unlike classification, in which objects are assigned to predefined classes, clustering does not have any predefined classes. The main advantage of clustering is that interesting patterns and structures can be found directly from very large data sets with little or none of the background knowledge. The cluster results are subjective and implementation dependent. The quality of a clustering method depends on:

- The similarity measure used by the method and its implementation
- Its ability to discover some or all of the hidden patterns
- The definition and representation of cluster chosen

A number of algorithms for clustering have been proposed by researchers, of which this study establishes with a comparative study of K-Means and K-Medoids clustering algorithms (Berkhin, 2002; Dunham, 2002; Han and Kamber, 2006; Xiong *et al*., 2009; Park *et al*., 2006; Khan and Ahmad, 2004; Borah and Ghose, 2009; Rakhlin and Caponnetto, 2007).

MATERIALS AND METHODS

Problem specification: In this research, two unsupervised clustering methods, namely K-Means and K-Medoids are examined to analyze based on the distance between the various input data points. The clusters are formed according to the distance between data points and cluster centers are formed for each cluster. The implementation plan will be in two parts, one in normal distribution and another in uniform distribution of input data points. Both the algorithms

Corresponding Author: T. Santhanam, Department of Computer Science, DG Vaishnav College, Chennai, India
Tel: +91-9444169090

are implemented by using JAVA language. The number of clusters is chosen by the user. The data points in each cluster are displayed by different colors and the execution time is calculated in milliseconds.

K-Means algorithm: K-Means is one of the simplest unsupervised learning algorithms that solve the well known clustering problem. The procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters) fixed a priori. The main idea is to define k centroids, one for each cluster. These centroids should be placed in a cunning way because of different location causes different result. So, the better choice is to place them as much as possible far away from each other. The next step is to take each point belonging to a given data set and associate it to the nearest centroid. When no point is pending, the first step is completed and an early group is done. At this point, it is need to re-calculate k new centroids as centers of the clusters resulting from the previous step. After these k new centroids, a new binding has to be done between the same data points and the nearest new centroid. A loop has been generated. As a result of this loop it may notice that the k centroids change their location step by step until no more changes are done. In other words centroids do not move any more. Finally, this algorithm aims at minimizing an objective function, in this case a squared error function. The objective function:

$$J = \sum_{j=1}^{k}\sum_{i=1}^{n}\left\|x_i^{(j)} - c_j\right\|^2$$

where, $\left\|x_i^{(j)} - c_j\right\|^2$ is a chosen distance measure between a data point $x_i^{(j)}$ and the cluster centre c_j, is an indicator of the distance of the n data points from their respective cluster centers. The algorithm is composed of the following steps:

1. Place k points into the space represented by the objects that are being clustered. These points represent initial group centroids
2. Assign each object to the group that has the closest centroid
3. When all objects have been assigned, recalculate the positions of the k centroids
4. Repeat Step 2 and 3 until the centroids no longer move

This produces a separation of the objects into groups from which the metric to be minimized can be calculated. Although it can be proved that the procedure will always terminate, the K-Means algorithm does not necessarily find the most optimal configuration, corresponding to the global objective function minimum. The algorithm is also significantly sensitive to the initial randomly selected cluster centers. The K-Means algorithm can be run multiple times to reduce this effect. K-Means is a simple algorithm that has been adapted to many problem domains. First, the algorithm randomly selects k of the objects. Each selected object represents a single cluster and because in this case only one object is in the cluster, this object represents the mean or center of the cluster.

K-Medoids algorithm: The 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. 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. The basic strategy of K-Mediods 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. 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 swap point 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;

Like this algorithm, a Partitioning Around Medoids (PAM) was one of the first k-Medoids algorithms introduced. 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.

RESULTS

In this study, the k-Means algorithm is explained with an example first, followed by k-Medoids algorithm. The two types of experimental results are discussed for the K-Means algorithm. One is Normal and another is Uniform distributions of input data points. For both the types of distributions, the data points are created by using Box-Muller formula. The resulting clusters of the normal distribution of K-Means algorithm is presented in Fig. 1. The number of clusters and data points is given by the user during the execution of the program. The number of data points is 1000 and the number of clusters given by the user is 10 (k = 10). The algorithm is repeated 1000 times 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.

For different input data points, the algorithm gives different types of outputs. The input data points are generated in red color and the output of the algorithm is displayed in different colors as shown in Fig. 1. The center point of each cluster is displayed in green color. The execution time of each run is 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 result shows the normal distribution of 1000 data points around ten cluster centers. The number of data points is the size of the cluster. The size of the cluster is high for some clusters due to increase in the number of data points around a particular cluster center. For example, size of cluster 10 is 139, because of the distance between the data points of that particular cluster and the cluster centers is very less. Since the data points are normally distributed, clusters vary in size with maximum data points in cluster 10 and minimum data points in cluster 8. For the same normal distribution of the input data points and for the uniform distribution, the algorithm is executed 10 times and the results are tabulated in the Table 1.

From Table 1 (against the rows indicated by 'N'), it is observed that the execution time for Run 6 is 7640 m sec and for Run 8 is 7297 msec. In a normal distribution of the data points, there is much difference between the sizes of the clusters. For example, in Run 7, cluster 2 has 167 points and cluster 5 has only 56 points. Next, the uniformly distributed one thousand data points are taken as input. One of the executions is shown in Fig. 2. The number of clusters chosen by user is 10. The result of the algorithm for 10 different execution of the program is given in Table 1 (against the rows indicated by 'U').

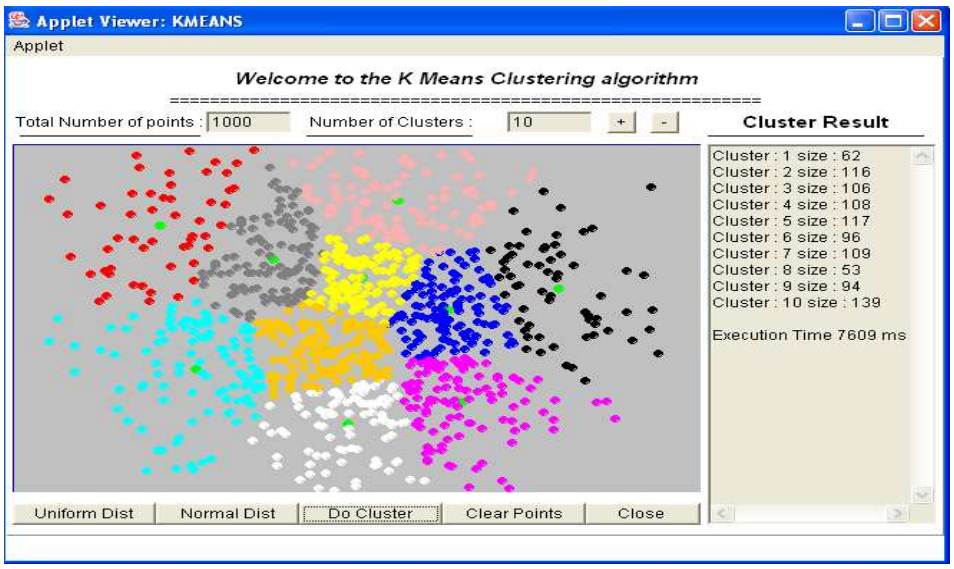

Fig. 1: Normal distribution output-K-Means

Table 1: Cluster results for K-Means algorithm

Cluster		1	2	3	4	5	6	7	8	9	10	Time (m sec)
Run 1	N	62	116	106	108	117	96	109	53	94	139	7609
	U	118	87	101	109	103	91	104	100	94	93	7281
Run 2	N	149	108	60	123	90	132	99	74	81	84	7500
	U	101	98	111	93	96	99	79	125	106	92	7500
Run 3	N	79	108	134	85	89	95	78	72	125	135	7578
	U	103	86	98	87	77	109	108	91	121	120	7469
Run 4	N	54	102	88	83	86	131	113	129	101	113	7391
	U	84	117	90	98	106	97	85	111	110	102	7375
Run 5	N	78	55	143	81	79	101	157	91	123	92	7360
	U	103	115	83	120	105	100	77	103	109	85	7297
Run 6	N	96	100	119	132	119	108	66	75	52	133	7640
	U	94	96	89	120	119	92	98	78	113	101	7437
Run 7	N	106	167	98	114	56	60	63	129	81	126	7515
	U	89	84	116	100	108	89	107	92	99	116	7422
Run 8	N	74	64	130	114	88	103	114	83	137	93	7297
	U	99	88	103	93	140	108	75	99	105	90	7469
Run 9	N	100	105	70	135	90	157	68	93	113	69	7453
	U	95	103	100	84	102	104	105	95	112	100	7391
Run 10	N	58	42	151	126	121	97	74	101	71	159	7422
	U	112	79	80	114	79	102	90	135	99	110	7407

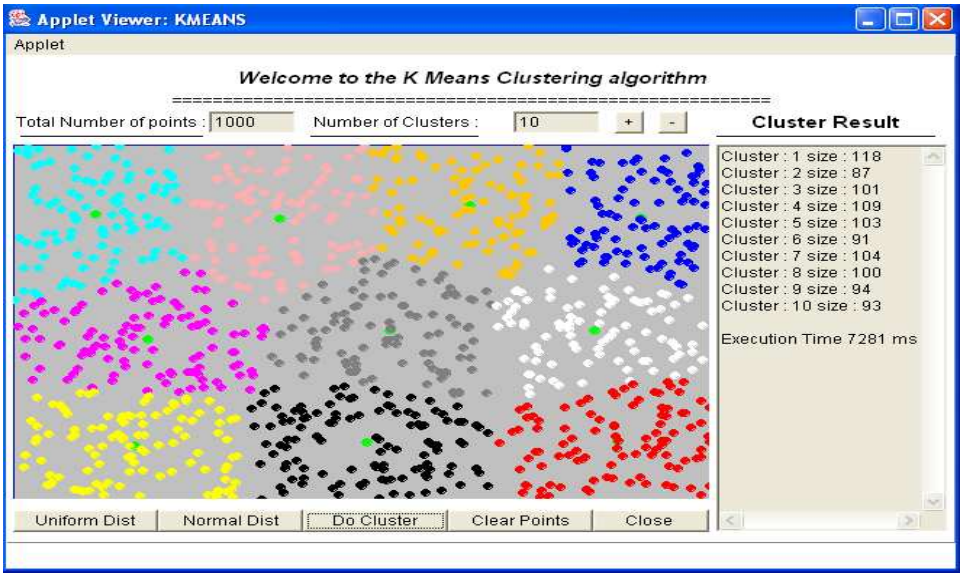

Fig. 2: Uniform distribution output-K-Means

From the result, it can be inferred that the size of the clusters are near to the average data points in the uniform distribution. This can easily be observed that for 1000 points and 10 clusters, the average is 100. But this is not true in the case of normal distribution. It is easy to identify from the Fig. 1 and 2, the size of the clusters are different. This shows that the behavior of the algorithm is different for both types of distributions.

Next, for the K-Medoids algorithm, the input data points and the experimental results are considered by the same method as discussed for K-Means algorithm. In the next example, one thousand normally distributed data points are taken as input. Number of clusters chosen by the user is 10. The output of one of the trials is shown in Fig. 3. Next, the uniformly distributed one thousand data points are taken as input. The number of clusters chosen by the user is 10. One of the executions is shown in Fig. 4. The result of the algorithm for ten different executions of the program is given in Table 2. For both the algorithms, K-Means and K-Medoids, the program is 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 based on observations.

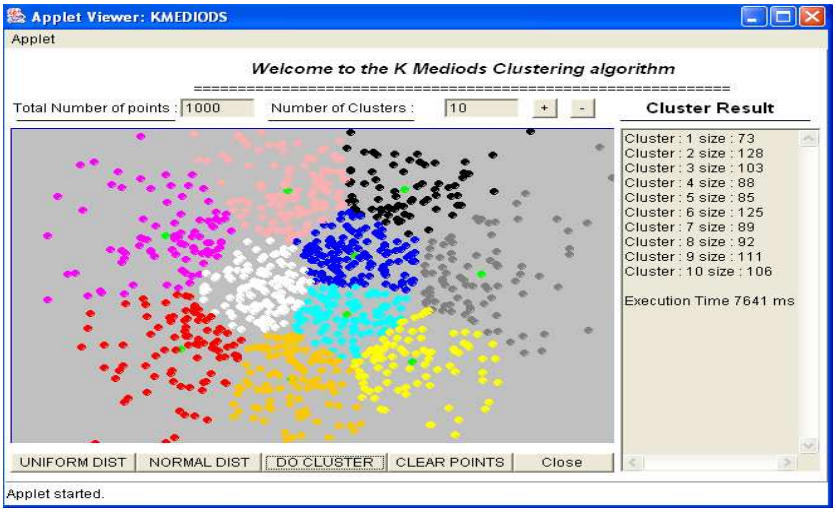

Fig. 3: Normal distribution output-K-Medoids

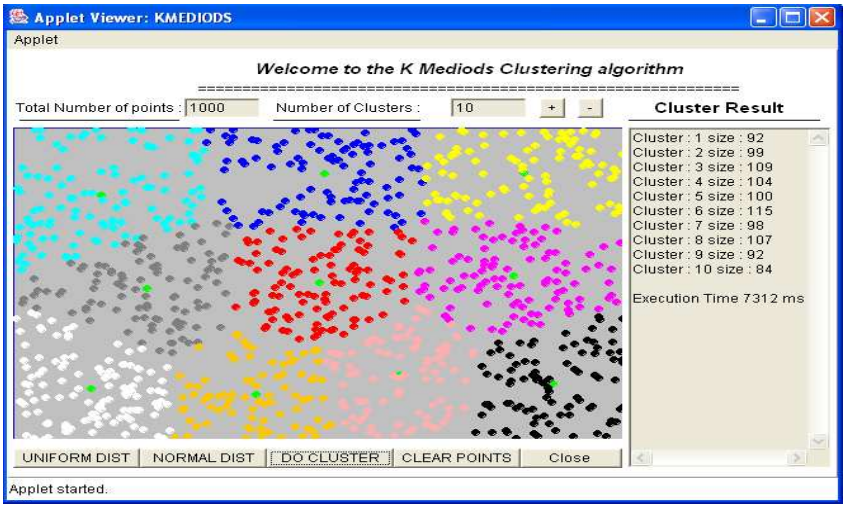

Fig. 4: Uniform distribution output-K-Medoids

Table 2: Cluster results for K-Medoids algorithm

Cluster		1	2	3	4	5	6	7	8	9	10	Time (m sec)
Run 1	N	73	128	103	88	85	125	89	92	111	106	7641
	U	92	99	109	104	100	115	98	107	92	84	7312
Run 2	N	73	71	138	186	86	101	124	83	64	76	7578
	U	119	73	80	99	93	89	141	94	105	107	7484
Run 3	N	83	53	149	75	110	123	61	93	130	123	7422
	U	98	91	109	107	89	89	118	114	93	92	7375
Run 4	N	66	80	80	65	95	170	81	97	148	118	7360
	U	111	90	95	89	96	117	103	112	186	81	7219
Run 5	N	78	91	73	165	82	104	154	74	60	119	7469
	U	115	88	101	114	96	97	111	96	87	95	7250
Run 6	N	95	142	76	80	79	147	131	73	89	88	7390
	U	116	118	110	100	99	83	79	97	99	99	7531
Run 7	N	99	81	137	98	153	63	76	73	82	138	7484
	U	100	104	105	97	91	82	111	98	96	116	7375
Run 8	N	129	83	120	111	112	95	68	123	80	79	7282
	U	104	112	97	80	70	82	136	95	131	93	7172
Run 9	N	98	84	54	116	116	60	110	60	156	146	7485
	U	100	64	97	94	88	112	101	117	117	110	7141
Run 10	N	81	139	81	83	150	133	83	94	85	71	7312
	U	102	115	113	77	68	86	91	143	105	100	7219

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 (250 data points, 500 data points etc), the outcomes are found to be highly satisfactory.

DISCUSSION

From the experimental results, for K-Means algorithm, the average time for clustering the normal distribution of data points is found to be 7476.5 m sec. The average time for clustering the uniform distribution of data points is found to be 7404.8 m sec. For the K-Medoids algorithm, the average time for clustering the normal distribution of data points is found to be 7442.3 m sec and for the average time for clustering the uniform distribution of data points are 7307.8 m sec. It is understood that the average time for normal distribution is greater than the average time for the uniform distribution. This is true for both the algorithms K-Means and K-Medoids. If the number of data points is less, then the K-Means algorithm takes lesser execution time. But when the data points are increased to maximum the K-Means algorithm takes maximum time and the K-Medoids algorithm performs reasonably better than the K-Means algorithm. The characteristic feature of this algorithm is that it requires the distance between every pair of objects only once and uses this distance at every stage of iteration.

CONCLUSION

The time taken for one execution of the program for the Uniform Distribution is less than the time taken for Normal Distribution. Usually the time complexity varies from one processor to another processor, which depends on the speed and the type of the system. The partition based algorithms work well for finding spherical-shaped clusters in small to medium-sized data points. The advantage of the K-Means algorithm is its favorable execution time. Its drawback is that the user has to know in advance how many clusters are searched for. From the experimental results (by many execution of the programs), it is observed that K-Means algorithm is efficient for smaller data sets and K-Medoids algorithm seems to perform better for large data sets.

REFERENCES

Berkhin, P., 2002. Survey of clustering data mining techniques. Technical Report, Accrue Software, Inc. http://www.ee.ucr.edu/~barth/EE242/clustering_survey.pdf

Borah, S. and M.K. Ghose, 2009. Performance analysis of AIM-K-Means and K-Means in quality cluster generation. J. Comput., 1: 175-178. http://arxiv.org/ftp/arxiv/papers/0912/0912.3983.pdf

Dunham, M., 2002. Data Mining: Introductory and Advanced Topics. 1st Edn., Prentice Hall, USA., ISBN: 10: 0130888923, pp: 315.

Han, J. and M. Kamber, 2006. Data Mining: Concepts and Techniques. Morgan Kaufmann Publishers, 2nd Edn., New Delhi, ISBN: 978-81-312-0535-8.

Jain, A.K. and R.C. Dubes, 1988. Algorithms for Clustering Data. Prentice Hall Inc., Englewood Cliffs, New Jersey, ISBN: 0-13-022278-X, pp: 320.

Jain, A.K., M.N. Murty and P.J. Flynn, 1999. Data clustering: A review. ACM Comput. Surveys, 31: 264-323. DOI: 10.1145/331499.331504

Khan, S.S. and A. Ahmad, 2004. Cluster center initialization algorithm for K-Means clustering. Patt. Recog. Lett., 25: 1293-1302. DOI: 10.1016/j.patrec.2004.04.007

Park, H.S., J.S. Lee and C.H. Jun, 2006. A K-means-like algorithm for K-medoids clustering and its performance. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.90.7981&rep=rep1&type=pdf

Rakhlin, A. and A. Caponnetto, 2007. Stability of k-Means clustering. Adv. Neural Inform. Process. Syst., 12: 216-222. http://cbcl.mit.edu/projects/cbcl/publications/ps/rakhlin-stability-clustering.pdf

Xiong, H., J. Wu and J. Chen, 2009. K-Means clustering versus validation measures: A data distribution perspective. IEEE Trans. Syst., Man, Cybernet. Part B, 39: 318-331. http://www.ncbi.nlm.nih.gov/pubmed/19095536