G.J. E.D.T.,Vol.2(6):1-8 (November-December, 2013) ISSN: 2319 – 7293

GLOBAL JOURNAL OF ENGINEERING, DESIGN & TECHNOLOGY
(Published By: Global Institute for Research & Education)

www.gifre.org

Performance assessment of CLARANS: A Method for Clustering Objects for Spatial Data Mining

Vijaya Sagvekar[1] , Vidya Sagvekar[2], & Kalpana Deorukhkar[3]

[1]Assistant Professor, Dept. of Computer Engineering, ACE, Mumbai,
[2]Assistant Professor, Dept. of Electronics Engineering, KJSIEIT, Mumbai.
[3]Assistant Professor, Dept. Of Computer Engineering, Fr.CRCE, Mumbai.

Abstract

Spatial data mining is the discovery of interesting relationships and characteristics that may exist implicitly in spatial databases. To this end, this paper has three main contributions. First, a new clustering method called CLARANS, whose aim is to identify spatial structures that may be present in the data. Experimental results indicate that, when compared with existing clustering methods, CLARANS is very efficient and effective. Second, investigate how CLARANS can handle not only points objects, but also polygon objects efficiently. One of the methods considered, called the IR-approximation, is very efficient in clustering convex and nonconvex polygon objects. Third, building on top of CLARANS, develop two spatial data mining algorithms that aim to discover relationships between spatial and nonspatial attributes. Both algorithms can discover knowledge that is difficult to find with existing spatial data mining algorithms.

Index Terms:-Spatial data mining, clustering algorithms, randomized search, computational geometry.

1. Introduction

Data mining in general is the search for hidden patterns that may exist in large databases. Spatial data mining in particular is the discovery of interesting relationships and characteristics that may exist implicitly in spatial databases. Because of the huge amounts (usually, terabytes) of spatial data that may be obtained from satellite images, medical equipments, video cameras, etc., it is costly and often unrealistic for users to examine spatial data in detail. Spatial data mining aims to automate such aknowledge discovery process. Thus, it plays an important role in
1. extracting interesting spatial patterns and features,
2. capturing intrinsic relationships between spatial and nonspatial data,
3. presenting data regularity concisely and at higher conceptual levels, and
4. helping to reorganize spatial databases to accommodate data semantics, as well as to achieve better performance.

Cluster Analysis is a branch of statistics that, in the past three decades, has been intensely studied and successfully applied to many applications. To the spatial data mining task at hand, the attractiveness of cluster analysis is its ability to find structures or clusters directly from the given data, without relying on any hierarchies. However, cluster analysis has been applied rather unsuccessfully in the past to general data mining and machine learning. The complaints are that cluster analysis algorithms are ineffective and inefficient. Indeed, for cluster analysis to work effectively, there are the following key issues:

Whether there exists a natural notion of similarities among the “objects” to be clustered. For spatial data mining, our approach here is to apply cluster analysis only on the spatial attributes. If these attributes correspond to point objects, natura lnotions of similarities exist (e.g., Euclidean or Manhattan distances). However, if the attributes correspond to polygon objects, the situation is more complicated. More specifically, the similarity (or distance) between two polygon objects may be defined in many ways, some better than others. But, more accurate distance measurements may require more effort to compute. The main question then is for the kind of spatial clustering under consideration, which measurement achieves the best balance.

Whether clustering a large number of objects can be efficiently carried out. Traditional cluster analysis algorithms are not designed for large data sets, with say more than 1,000 objects.

In addressing these issues, the authors reported in this paper:

The development of CLARANS, which aims to use randomized search to facilitate the clustering of a large number of objects and

A study on the efficiency and effectiveness of three different approaches to calculate the similarities between polygon objects. They are the approach that calculates the exact separation distance between two polygons, the approach that overestimates the exact distance by using the minimum distance between vertices, and the approach that underestimates the exact distance by using the separation istance between the isothetic rectangles of the polygons.

CLARANS is more efficient than the existing algorithms PAM and CLARA, both of whichmotivate the development of CLARANS; Calculating the similarity between two polygons by using the separation distance between the isothetic rectangles of the polygons is the most efficient and effective approach.

2. Clustering Algorithms Based on Partitioning

2.1 Overview

Cluster analysis has been widely applied to many areas such as medicine (classification of diseases), Chemistry (grouping of compounds), social studies (classification of statistical findings), etc. Its main goal is to identify structures or clusters present in the data. Existing clustering algorithms can be classified into two main categories: hierarchical methods and partitioning methods.

Hierarchical methods are either agglomerative or divisive. Given n objects to be clustered, agglomerative methods begin with n clusters (i.e., all objects are apart). In each step, two clusters are chosen and merged. This process continues until all objects are clustered into one group. On the other hand, divisive methods begin by putting all objects in one cluster. In each step, a cluster is chosen and split up into two. This process continues until n clusters are produced. While hierarchical methods have been successfully applied to many biological applications (e.g., for producing taxonomies of animals and plants), they are known to suffer from the weakness that they can never undo what was done previously. Once an agglomerative method merges two objects, these objects will always be in one cluster. And once a divisive method separates two objects, these objects will never be regrouped into the same cluster.

In contrast, given the number k of partitions to be found, a partitioning method tries to find the best k partitions of the n objects. It is very often the case that the k clusters found by a partitioning method are of higher quality (i.e., more similar) than the k clusters produced by a hierarchical method. Because of this property, developing partitioning methods has been one of the main focuses of cluster analysis research. Indeed, many partitioning methods have been developed, some based on k-means, some on k-medoid, some on fuzzy analysis, etc. Among them, we have chosen the k-medoid m methods, the k-medoid methods are very robust to the existence of outliers (i.e., data points that are very far away from the rest of the data points). Second, clusters found by k-medoid methods do not depend on the order in which the objects are examined. Furthermore, they are invariant with respect to translations and orthogonal transformations of data points. Last but not least, experiments have shown thatthe k-medoid methods described below can handle very large data sets quite efficiently. For a more detailed comparison of k-medoid methods with other partitioning methods. In the remainder of this section, the authors present the two best-known k-medoid methods on which our algorithm is based.

2.2 PAM

PAM (Partitioning Around Medoids) was developed by Kaufman and Rousseeuw . To find k clusters, PAM’s approach is to determine a representative object for each cluster. This representative object, called a medoid, is meant to be the most centrally located object within the cluster.

Once the medoids have been selected, each nonselected object is grouped with the medoid to which it is the most similar. More precisely, if Oj is a nonselected object and Om is a (selected) medoid, we say that Oj belongs to the cluster represented by Om if d(Oj,Om) = minOed(Oj;Oe), where the notation minOe denotes the minimum over all medoids Oe and the notation d(O1,O2) denotes the dissimilarity or distance between objects O1 and O2. All the dissimilarity values are given as inputs to PAM. Finally, the quality of a clustering (i.e., the combined quality of the chosen medoids) is measured by the average dissimilarity between an object and the medoid of its cluster. To find the k medoids, PAM begins with an arbitrary selection of k objects. Then, in each step, a swap between a selected object Om and a nonselected object Op is made, as long as such a swap would result in an improvement of the quality of the clustering.

Algorithm PAM

1. Select k representative objects arbitrarily.
2. Compute TCmp for all pairs of objects Om,Op where Om is currently selected, and Op is not.
3. Select the pair Om,Op which corresponds to minOm,Op TCmp. If the minimum TCmp is negative, replace Om with Op, and go back to Step 2.
4. Otherwise, for each nonselected object, find the most similar representative object. Halt.

Experimental results show that PAM works satisfactorily for small data sets (e.g., 100 objects in 5 clusters). But, it is not efficient in dealing with medium and large data sets. This is not too surprising if we perform a complexity analysis on PAM. In Steps 2 and 3, there are altogether k(n - k) pairs of Om;Op. For each pair, computing TCmp requires the examination of (n-k) nonselected objects.

Thus, Steps 2 and 3 combined is of $O(k(n-k)^2)$. And this is the complexity of only one iteration. Thus, it is obvious that PAM becomes too costly for large values of n and k. This analysis motivates the development of CLARA.

2.3 CLARA

Designed by Kaufman and Rousseeuw to handle large data sets, CLARA (Clustering LARge Applications) relies on sampling . Instead of finding representative objects for the entire data set, CLARA draws a sample of the data set, applies PAM on the sample, and finds the medoids of the sample. The point is that, if the sample is drawn in a sufficiently random way, the medoids of the sample would approximate the medoids of the entire data set. To come up with better approximations, CLARA draws multiple samples and gives the best clustering as the output. Here, for accuracy, the quality of a clustering is measured based on the average dissimilarity of all objects in the entire data set, and not only of those objects in the samples. Experiments reported in indicate that five samples of size 40 þ 2k give satisfactory results.

Algorithm CLARA

1. For i = 1 to 5, repeat the following steps:
2. Draw a sample of 40 + 2k objects randomly from the entire data set and call Algorithm PAM to find k medoids of the sample.
3. For each object Oj in the entire data set, determine which of the k medoids is the most similar to Oj.
4. Calculate the average dissimilarity of the clustering obtained in the previous step. If this value is lessthan the current minimum, use this value as the current minimum, and retain the k medoids found inStep 2 as the best set of medoids obtained so far.
5. Return to Step 1 to start the next iteration.

Complementary to PAM, CLARA performs satisfactorily for large data sets (e.g., 1,000 objects in 10 clusters).Each iteration of PAM is of O(k(n _ k)2). But, for CLARA, by applying PAM just to the samples, each iteration is of $O(k(40 + k)^2 + k(n – k))$.This explains why CLARA is more efficient than PAM for large values of n.

3. Clustering Algorithm Based On Randomized Search

CLARANS (Clustering Large Applications based on RANdomized Search). first give a graph-theoretic framework within which we can compare PAM and CLARA, and motivate the development of CLARANS. Then, after describing the details of the algorithm, we will present experimental results showing how to fine tune CLARANS and that CLARANS outperforms CLARA and PAM in terms of both efficiency and effectiveness.

3.1 CLARANS

Many of the aforementioned techniques require some tree or grid structures to facilitate the clustering.Consequently, these techniques do not scale up well with increasing dimensionality of the datasets.While it is true that the material discussed in this paper is predominantly 2D, the CLARANS algorithm works the same way for higher dimensional datasets. Because CLARANS is based on randomized search and does not use any auxiliary structure, CLARANS is much less affected by increasing dimensionality.

Many of the aforementioned techniques assume that the distance function is Euclidean. CLARANS, being a local search technique, makes no requirement on the nature of the distance function.
Many of the aforementioned techniques deal with point objects; CLARANS is more general and supports polygonal objects. A considerable portionof this paper is dedicated to handling polygonal objects effectively.
CLARANS is a main-memory clustering technique, while many of the aforementioned techniques are designed for out-of-core clustering applications. We conclude that whenever extensive I/O operations are involved, CLARANS is not as efficient as the others. CLARANS still has considerable applicability.

Consider the 2D objects. Each object is represented by two real numbers, occupying a total of 16 bytes. Clustering 1,000,000 objects would require slightly more than 16 Mbytes of main memory. This is an amount easily affordable by a personal computer, let alone computers for data mining. The point here is that, given the very low cost of RAM, main-memory clustering algorithms, such as CLARANS, are not completely dominated by out-of-core algorithms for many applications. Finally, on a similar note, although some newly developed clustering methods may find clusters “natural” to the human eye and good for certain applications, there are still many applications, such as delivery services, to which partitioning-based clustering, such as CLARANS, is more appropriate.

3.2 Algorithm CLARANS

1. Input parameters numlocal and maxneighbor. Initialize i to 1, and mincost to a large number.
2. Set current to an arbitrary node in Gn;k
3. Set j to 1.
4. Consider a random neighbor S of current, and based on 5, calculate the cost differential of the two nodes.
5. If S has a lower cost, set current to S, and go to Step 3.
6. Otherwise, increment j by 1. If j maxneighbor, go to Step 4.
7. Otherwise, when j > maxneighbor, compare the cost of current with mincost. If the former is less than mincost, set mincost to the cost of current and set bestnode to current.
8. Increment i by 1. If i > numlocal, output bestnode and halt. Otherwise, go to Step 2.

Steps 3 to 6 above search for nodes with progressively lower costs. But, if the current node has already been compared with the maximum number of the neighbors of the node (specified by maxneighbor) and is still of the lowest cost, the current node is declared to be a “local” minimum. Then, in Step 7, the cost of this local minimum is compared with the lowest cost obtained so far. The lower of the two costs above is stored in mincost. Algorithm CLARANS then repeats to search for other local minima, until numlocal of them have been found.

CLARANS has two parameters: the maximum number of neighbours examined (maxneighbor) and the number of local minima obtained (numlocal). The higher the value of maxneighbor, the closer is CLARANS to PAM, and the longer is each search of a local minima. But,the quality of such a local minima is higher and fewer local minima needs to be obtained. Like many applications of randomized search , we rely on experiments to determine the appropriate values of these parameters.

3.3 CLARANS versus PAM

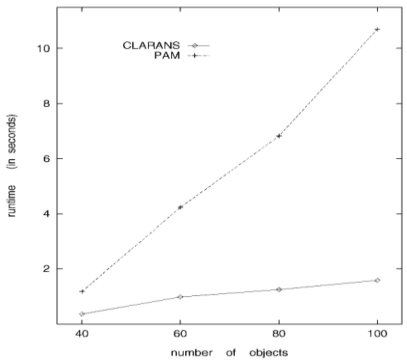

Fig.1 Efficiency: CLARANS versus PAM

For large and medium data sets, it is obvious that CLARANS, while producing clusterings of very comparable quality, is much more efficient than PAM.

For small datasets like here, both algorithms to data sets with 40, 60, 80, and 100 points in five clusters. Fig shows the runtime taken by both algorithms. Note that, for all those data sets, the clusterings produced by both algorithms are of the same quality (i.e. same average distance). Thus, the difference between the two algorithms is determined by their efficiency. It is evident from Fig. that, even for small data sets, CLARANS outperforms PAM significantly. As expected, the performance gap between the two algorithms grows, as the dataset increases in size.

3.4 CLARANS versus CLARA

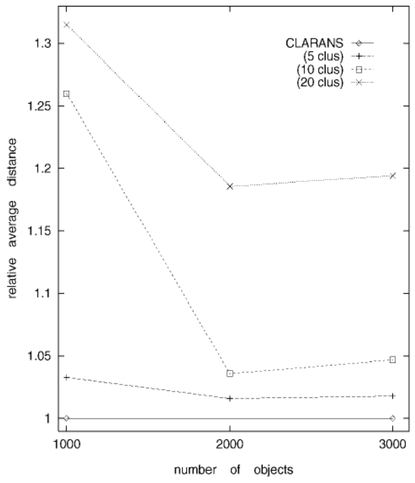

Fig.2. Relative Quality: Same time for CLARANS and CLARA.

CLARA is not designed for small data sets. Thus, ran this set of experiments on data sets whose number of objects exceeds 100. And the objects were organized in different number of clusters.When conducted this series of experiments running CLARA and CLARANS as presented earlier, CLARANS is always able to find clusterings of better quality than those found by CLARA.

In some cases, CLARA may take much less time than CLARANS. Thus, wondered whether CLARA would produce clusterings of the same quality if it was given the same amount of time. This leads to the next series of experiments in which we gave both CLARANS and CLARA the same amount of time.

Fig.2 shows the quality of the clusterings produced by CLARA, normalized by the corresponding value produced by CLARANS. Given the same amount of time, CLARANS clearly outperforms CLARA in all cases.

The gap between CLARANS and CLARA increases from 4 percent when k, the number of clusters, is five to 20 percent when k is 20.This widening of the gap as k increases can be best explained by looking at the complexity analyses of CLARA and CLARANS. Each iteration of CLARA is of $O(k^3+nk)$.

The cost of CLARANS is basically linearly proportional to the number of objects.3 Thus, an increase in k imposes a much larger cost on CLARA than on CLARANS. The above complexity comparison also explains why, for a fixed number of clusters, the higher the number of objects, the narrower the gap between CLARANS and CLARA .

For example, when the number of objects is 1,000, the gap is as high as 30 percent. The gap drops to around 20 percent as the number of object increases to 2,000. Since each iteration of CLARA is of $O(k^3 +nk)$, the first term k^3 dominates the second term. Thus, for a fixed k, CLARA is relatively less sensitive to an increase in n.

Since the cost of CLARANS is roughly linearly proportional to n, an increase in n imposes a larger cost on CLARANS than on CLARA. This explains why, for a fixed k, the gap narrows as the number of objects increases. Nonetheless, the bottom-line shown in Fig. is that CLARANS beats CLARA in all cases. Presented experimental evidence showing that CLARANS is more efficient than PAM and CLARA for small and large data sets. As authors stated, experimental results for medium data sets (not included here) lead to the same conclusion.

4. Clustering Convex Polygon Objects

4.1 Need

Each object is represented as a point, in which case standard distance metrics such as the Manhattan distance and the Euclidean distance can be used to calculate the distance between two objects/points. However, in practice, numerous spatial objects that we may want to cluster are polygonal in nature, e.g., shopping malls, parks. The central question then is how to calculate the distance between two polygon objects efficiently and effectively for clustering purposes. One obvious way to approximate polygon objects is to represent each object by a representative point, such as the centroid of the object.However, in general, the objects being clustered may have widely varied sizes and shapes. For instance, a typical house in Vancouver may have a lot size of 200 square meters and a rectangular shape, whereas Stanley Park in Vancouver has a size of about 500,000 square meters and an irregular shape that hugs the shoreline. Simply representing each of these objects by its centroid, or any single point, would easily produce clusterings of poor quality.

CLARANS (and for that matter, CLARA and PAM) can be augmented to allow convex polygon objects—in their entireties—to be clustered. The key question is how to efficiently compute the distance between two polygons. There are three different approaches.

The first one is based on computing the exact separation distance between two convex polygon objects.
The second approach uses the minimum distance between vertices to approximate the exact separation distance.
The third approach approximates by using the separation distance between isothetic rectangles. We analyze the pros and cons, and complexities of these approaches. Last but not least, we propose a performance optimization that is based on memoizing the computed distances. At the end of this section, we will give experimental results evaluating the usefulness of these ideas.

4.2 Calculating the Exact Separation Distance between point and line

In coordinate geometry, the distance between a point P and a line L is defined as the minimum distance, in this case the perpendicular distance, between the point and the line, i.e.,min {d(P,Q)|Q is a point on L}. Thus, given two polygons A;B, it is natural for us to define the distance between these two polygons to be the minimum distance between any pair of points in A,B, i.e., min {d(P;Q)| P,Q are points in A,B respectively}. This distance is exactly the same as the minimum distance between any pair of points on the boundaries of A,B. This is called the separation distance between the two polygons

Distance between two convex polygons

Determine whether A and B have any intersection. This can be done in O(log n + log m) time where n,m denotes the number of vertices A and B (A vertix of a polygon is the intersection point between two edges of the polygon.)
If the two polygons intersect, then the separation distance is zero. Otherwise, we compute the separation distance between the two boundaries. This again can be done in O(log n + log m) time.

4.3 MV-approximation

One way to approximate the exact separation distance between two polygons is to find the minimum distance between the vertices of the polygons, i.e., min{d(P,Q) |P, are vertices of A,B respectively}.this approximation is called as the MV-approximation. Obviously, the MV-approximation requires a time complexity of O(n*m).

4.4 Comparison

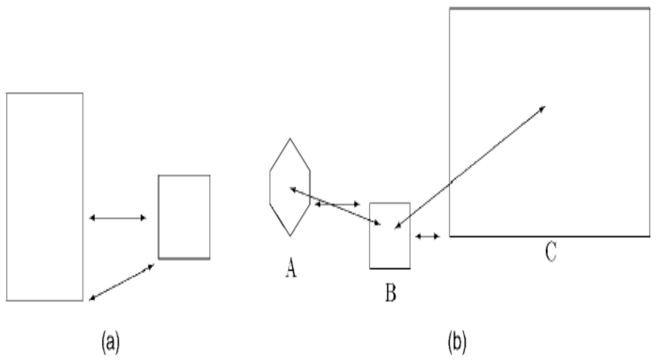

Fig3.(a) MV-approximation versus Seperation Distance versus Centroid Distance
(b) Centroid versus separation distance

Fig.3(a) shows a simple example demonstrating that the separation distance between two polygons need not be equal to the minimum distance between vertices. However, it is easy to see that the separation distance cannot exceed the minimum distance between vertices. Thus, the MV-approximation always overestimates the actual separation distance. From the point of view of clustering objects by their MV-approximations, the key question is whether such overestimations would affect the (quality of the) clusterings.

In Fig. b, B is closer to C than to A based on their exact separation distances. Whereas the centroid distance between A and B is fairly close to the between B and C is many times higher than the actual separation distance between B and C. Infact, by their centroid distances, B is closer to A than to C, thus reversing the ordering induced by their separation distances. In general, if a collection of objects has widely varied sizes and shapes, the centroid distance approximation would produce poor clusterings (relative to the clusterings produced by using the separation distances). On the other hand, for the example shown in Fig.b, the MV-approximation preserves the ordering that B is closer to C than to A. Certainly, an approximation is an approximation and it is not hard to construct situations where the MV-approximation can be inaccurate. However, by trying the approximation on numerous polygon objects that can be found in real maps, we have verified that the MV-approximation is reasonable and is much less susceptible to variations in sizes and shapes than the centroid distance approximation.

4.5 IR-approximation

Another way to approximate the exact separation distance between two polygons A,B is to
1) compute isothetic rectangles I_A, I_B and
2) Calculate the separation distance between I_A and I_B. Given a polygon A, the isothetic rectangle I_A is the smallest rectangle that contains A, and whose edges are parallel to either the x- or the
y-xes.Hereafter, refer to this approximation to the exact separation distance as the IR-approximation.

IR-approximation (and the MV-approximation) as a performance optimization to computing the exact separation distance. Actually, there is another advantage being offered by the IR-approximation
(and the MV-approximation). That is, it does not require the original polygon to be convex. As described above, the definition of the isothetic rectangle of a polygon applies equally well to convex and nonconvex polygons. Thus, when integrated with the IR-approximation, CLARANS can be used to cluster any polygons.

5. Performance Evaluation

5.1 Efficiency

Exact Distance versus IR-Approximation versus MV-Approximation

In this series of experiments done by authors, used polygons with different numbers of vertices and recorded the average amount of runtime needed to calculate the exact separation distance, its IR-approximation, and its MV-approximation for one pair of polygons. Fig. 6 shows the results with the runtimes recorded in milliseconds.

The IR-approximation approach is the clear winner in that it always outperforms the exact separation distance approach, that it beats the MV-approximation approach by a wide margin when the number of vertices is high, and that, even when the number of vertices is small, it delivers a performance competitive with that of the MV-approximation approach. Recall from Section 4.4 that two steps are needed to compute the IR-approximation. The first step is to compute the isothetic rectangles, which has a complexity of O(n). The second step is to compute the separation distance

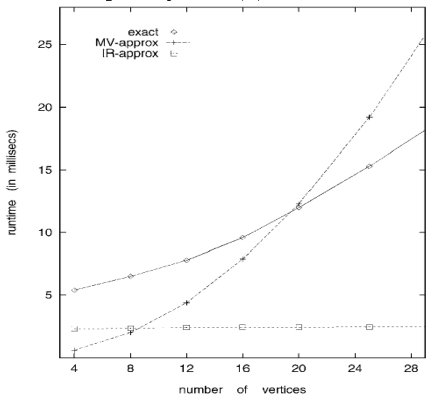

Fig. 4 Efficiency of the approximates

between the isothetic rectangles, which has a complexity of O(1). The flatness of the curve for the IR-approximation in Fig. 4 clearly shows that the second step dominates the first one. Thus, as a whole, the IR-approximation does not vary as the number of vertices increases. In contrast, both the exact separation distance and the MV-approximation require higher runtimes as the number of vertices increases. When the number of vertices is less than 20, calculating the exact separation distance takes more time than the MV-approximation does. But, the reverse is true when the number of vertices exceeds 20. In other words, the runtime for the MV-approximation grows faster than that for computing the exact separation distance.

5.2Clustering Efficiency

Exact Distance versus IR-approximation versus MV-approximation The two approximations relative to the exact distance approach.

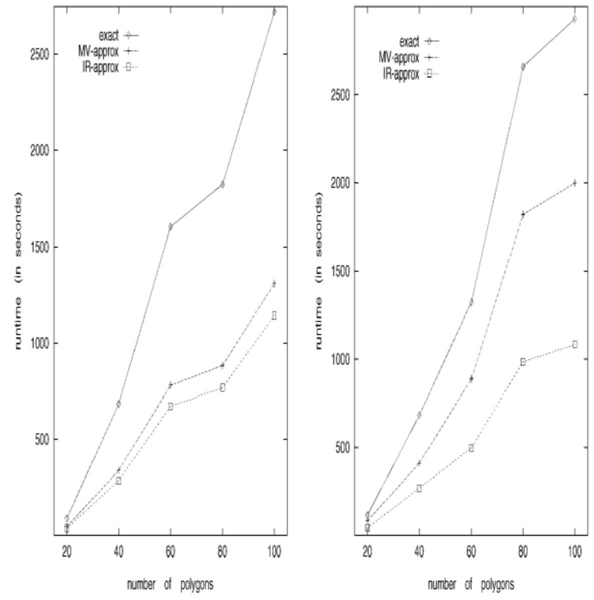

Fig.5 clustering efficiency of the approximations (a) 10-sided polygons (b) 4-to-20 Sided polygons.

Fig. 5 shows the times needed by CLARANS to cluster a varying number of polygons that have 10 edges and a varying number of polygons that have between 4 to 20 edges.

In both cases, the IR-approximation and the MV-approximation considerably outperform the exact separation distance approach. In particular, the IR-approximation is always the most efficient, requiring only about 30 percent to 40 percent the time needed by the exact distance approach. Other sets of polygons that we experimented with, including some that have varying densities, also give the same conclusion. Thus, given the fact that the IR-approximation is capable of delivering clusterings that are of almost identical quality to those produced by the exact approach, the IR-approximation is the definite choice for CLARANS. Other experiments that authors conducted indicate that same conclusion can be drawn if PAM and CLARA were to use to cluster polygon objects.

5.3 Effectiveness of Memoizing Computed Distances

While the graphs in Fig. 5 identify the IR-approximation approach as the clear winner, the performance results of the approximation is disappointing. For example, it takes about 1,000 seconds to cluster 100 polygons. Recall from Fig. 4 that the IR-approximation for one pair of polygons takes two to three milliseconds. For 100 polygons, there are 100*100/2 =5,000 pairs of polygons. These 5,000 distances take a total of 10 to 15 seconds to compute.

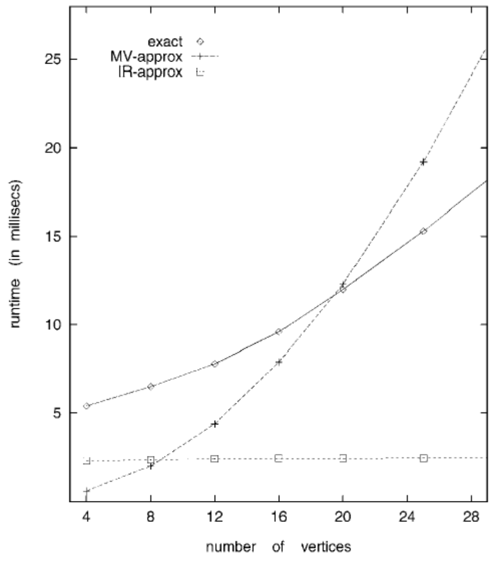

Fig.6 Clustering efficiency with distance memoization

Thus, what is happening is that the distance between each pair of polygons iscomputed on the average 60 to 100 times. This argues very strongly why computed distances should be memoized as a performance optimization. Fig. 6 shows the results of applying memoization to the same set of polygons used in Fig. 5b. Indeed, with memoization, the time taken to cluster 100 polygons with the IR-approximation drops to about 10 seconds, as estimated above. Similar performance gains are obtained if either the exact separation distance or the MV-approximation are used.

6. Conclusion

In this paper, presented a clustering algorithm called CLARANS which is based on randomized search. For small data sets, CLARANS is a few times faster than PAM. The performance gap for larger data sets is even larger. When compared with CLARA, CLARANS has the advantage that the search space is not localized to a specific subgraph chosen a priori, as in the case of CLARA.Consequently, when given the same amount of runtime,CLARANS can produce clusterings that are of much better quality than those generated by CLARA.Polygon objects can be clustered by CLARANS. The authors proposed three different ways to compute the distance between two polygons.Complexity and experimental results indicate that the IR-approximation is a few times faster than the method that computes the exact separation distance. Furthermore,experimental results show that despite the much smaller runtime, the IR-approximation is able to find clusterings that are of quality almost as good as those produced by using the exact separation distances.The IR-approximation can give ignificant efficiency gain ,but without loss of effectiveness. Spatial Dominant Clarans, assumes that items to be clustered contain both spatial and non-spatial components. It first clusters the spatial components using clarans and then examines the non-spatial attributes within each cluster to derive a description of that cluster.Non- Spatial Dominant Clarans , first looks at the non spatial attributes. By performing a generalization on these attributes, a set of representative tuples, one representing each cluster, can be found. Then the algorithm determines which spatial objects go with which representative tuple to finish the clustereing process.

7. References

[1] Margaret H. Dunham and S.Sridhar, “Data Mining- Introductory and Advanced Topics,”Peasrson Education.

[2] R. Ng and J. Han, “Efficient and Effective Clustering Methods for Spatial Data Mining,” Proc. 20th Conf. Very Large Databases,pp. 144–155, 1994.

[3] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan, “Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications,” Proc. 1998 ACM-SIGMOD, pp. 94–105,1998.

[4] R. Agrawal, S. Ghosh, T. Imielinski, B. Iyer, and A. Swami, “An Interval Classifier for
Database Mining Applications,” Proc. 18th Conf. Very Large Databases, pp. 560–573, 1992.

[5] R. Agrawal, T. Imielinski, and A. Swami, “Mining Association Rules between Sets of Items in Large Databases,” Proc. 1993 ACM Special Interest Group on Management of Data, pp. 207–216, 1993.

[6]M. Ankerst, M. Breunig, H.-P. Kriegel, and J. Sander, “OPTICS:Ordering Points To Identify the Clustering Structure,” Proc. 1999 ACM Special Interest Group on Management of Data, pp. 49–60, 1999.

Author’s Profile

Vijaya R. Sagvekar received B.E. degree in Computer Engineering from the University of Mumbai, and M.Tech in Information Technology from the NMIMS University, Mumbai, Maharashtra, in 2012. Currently, she is an assistant Professor of Computer Engineering at University of Mumbai. Her teaching and research areas include Computer Networking, Computer Security, and Network Simulator.

Vidya R. Sagvekar received B.E. degree in Electronics Engineering from the Pune University, in 2004, and M.E. in Electronics and Telecommunication Engineering from the Mumbai University, Mumbai, Maharashtra, in 2013. Currently, she is an assistant Professor of Electronics Engineering at University of Mumbai. Her teaching and research areas includes Computer Networking, Network Simulator.

Kalpana P. Deorukhkar received B.E. degree in Computer Engineering and M.E. in Computer Engineering from University of Mumbai, Maharashtra. Currently, she is an assistant Professor of Computer Engineering at University of Mumbai. Her teaching and research areas include image processing, data mining.