Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 2, Issue. 7, July 2013, pg.41 – 47

RESEARCH ARTICLE

ADCA: Advanced Density Based Clustering Algorithm for Spatial Database System

Abhaya Kumar Sahoo
Department of Information Technology, C.V. Raman College of Engineering, Bhubaneswar, India
kitsaks3@gmail.com

Abstract— Cluster detection in Spatial Databases is an important task for discovery of knowledge in spatial databases and in this domain density based clustering algorithms are very effective. Density Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm effectively manages to detect clusters of arbitrary shape with noise, but it fails in detecting local clusters as well as clusters of different density present in close proximity. Density Differentiated Spatial Clustering (DDSC) and Local-Density Based Spatial Clustering Algorithm with Noise (LDBSCAN) manage to detect clusters of different density as well as local clusters very effectively, but the number of input parameters is very high. Here I have proposed a new density based clustering algorithm with the introduction of a concept called Cluster Constant which basically represents the uniformity of distribution of points in a cluster. In order to find the density of a point I have used new measure called Reachability-Density. The proposed algorithm has minimized the input to be provided by the user down to one parameter (Minpts) and has made the other parameter (Eps) adaptive. Here I have also used some heuristics in order to improve the running time of the algorithm. Experimental results show that the proposed algorithm detects local clusters of arbitrary shape of different density present in close proximity very effectively and improves the running time when applied the heuristic.

Key Terms: - Spatial Data Mining; Clustering

I. INTRODUCTION

Spatial Database System (SDBS) is used for the management of spatial objects such as point, polygon or region representing a part of the surface of the earth [1]. Spatial database contains huge amount of spatial featured data [1].These data are gathered from various resources such as satellite images, geological survey, maps etc. As time passes more and more data are added which makes the size of the database really huge. Hence to extract knowledge from these huge databases there is a need for better organization of data. Clustering based framework has widely been used for the organization of spatial data. It is a process of grouping data such that, similar objects are put in the same group and dissimilar objects are put in different groups. Many clustering based framework exist for grouping data. For spatial objects, density based methods are effective frameworks. It groups the objects based on similar density region. It is more logical to use density based clustering for grouping spatial objects, because a cluster is basically a high density region as compared to its surrounding region and spatial objects are densely populated around a region. Following requirements are needed to be fulfilled when density based clustering methods are used in spatial databases [2]:

1) Minimizing the number of input parameter, because these parameter values are very difficult to gather in advance.

2) It should be able to detect cluster of arbitrary shape.

3) Good runtime complexity, because the algorithm operates on a database whose size is large.

In this paper, a new density based clustering algorithm is proposed with the introduction of a new concept, *"Cluster Constant"*, which basically represents the uniformity of distribution in a cluster. The algorithm requires only one input parameter (Minpts) and has made the other parameter (Eps) adaptive.

I have also used a heuristic in order to improve the running time of the algorithm. The paper is organized as follows. In section 2, the related work is briefly discussed. The basic definitions which are used in this algorithm are presented in section 3. In section 4, the algorithm called ADCA is explained. In section 5, implementation results are shown. In the end a conclusion is given along with some directions for future works.

II. RELATED WORKS

DBSCAN [3] algorithm is the base of the all density based clustering algorithms. The algorithm grows regions with sufficiently high density into clusters and discovers clusters of arbitrary shape in spatial databases with noise. It defines a cluster as a maximal set of density-connected points. It requires two inputs from user, i.e. Eps and Minpts, based on which it detects clusters. It can identify noise effectively, but it cannot identify local clusters if present very close to other clusters. So, the cluster which it detects has wide variation in its local density.

OPTICS [4] is an extension of DBSCAN algorithm which generates an order in which the objects needed to be processed. It then uses core-distance and reachability-distance in order to assign the each object a cluster membership. This order is generated through the reachability-distance and put in an ordered file. This ordered file is then used for assigning the cluster ID to each object. Here Eps parameter plays an important role. By changing the Eps value, different structure of cluster is detected.

IDBSCAN [5] is an improvement of DBSCAN algorithm in terms of execution time. DBSCAN algorithm spends its major time in each object's region query. So, instead of expanding every object inside the region of core objects, IDBSCAN proposed the expansion of only those objects which are at the boundary of the region. This is because the expansion of boundary objects would cover the objects which would have been covered by the objects, situated inside the region of core object, if had they been expanded. But it suffers from the limitations, similar to that of the DBSCAN.

LD-BSCA [6] is an improvement of DBSCAN algorithm in terms of reduction of number of input parameter and execution time. It requires only Eps as an input parameter. During the expansion of cluster, it considers density of only those objects which has not been assigned a cluster ID. Hence the neighborhood query is not performed for the Eps neighborhood objects of core object. In this way, it removes the neighborhood query of many objects and hence achieves an improvement over execution time. Here the limitations are same as that of the DBSCAN.

LDBSCAN [7] is another algorithm which can detect different density based cluster. It uses the concept of LOF [8] which represent the degree of outlierness and hence indicates whether the object is a core object or not. It then uses LRD [8] in order to assign an object to its corresponding cluster during the cluster expansion.

In [9] DDSC is proposed, which can identify cluster of different shape, size and density. It detects the change in density as the change in the number of objects inside a region. If the change in the density of region of an object is significant, then it indicates that the region query is entering into different density cluster and hence the cluster assignment proceeds with different cluster identification.

DENCLUE [10] is an algorithm which generalizes many clustering algorithm (DBSCAN, k-means, Hierarchical). It uses the concept of gradient in order to find the object which is density-attractor. All the objects which are density attracted to density-attractor are assigned to same cluster identity as that of density attractor.

LOF [8] is another density based algorithm that assigns a degree of outlierness to each object. Instead of assigning a cluster ID its assign how much outlier a point is in comparison to its surrounding region. It uses LRD to measure the local density of the objects. Through LRD, the LOF of each object is calculated which measures the outlierness of the object.

III. BASIC NOTIONS OF ADCA

A. Problems in Existing Approaches:

DBSCAN algorithm does manage to find cluster effectively, but it cannot identify local clusters which is present in close proximity. The constant value of Eps largely responsible for this problem. Hence the Eps value needs to be adaptive in order to remove this problem. DDSC and LDBSCAN algorithm does manages to find local clusters, but the number of parameters that are needed to be optimized has also increased in case of these two algorithms.

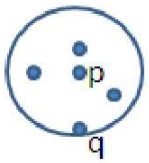

Fig. 1. 4-Distance of object p

Larger the number of input more will be user involvement and hence less accurate will be the cluster result. Hence there is a need to reduce the number of input parameters. There are two problems that can be identified from above discussion. First is to make the Eps value adaptive and second is to reduce the number of input parameter. The algorithm which is proposed here, does manages to solve the problems very effectively and efficiently.

B. Formal Definition of RD (Reachability Density):

RD of an object represents its surrounding density. The formal definition of RD will be presented shortly in the following which requires the explanation of following concepts:

- Eps: It is the radius of the circle.
- Minpts: It is the number of points in Eps-neighborhood of a point.

Definition 1 *(Minpts-Distance of object p)*: For any positive integer Minpts, the *Minpts-distance of object p*, denoted as Minpts-distance (p), is defined as the distance d (p, o) between p and an object oεD, where D is the entire dataset, such that

1) for at least Minpts objects o'εD it holds that d(p,o') ≤ d(p,o), and
2) for at most Minpts-1 objects o'εD it holds that d(p,o') < d(p,o).

Figure 1 shows the concept of definition 1. Suppose Minpts is 4. If p is the object under consideration and q is 4th nearest to point p, then the distance between p and q is the Minpts- Distance of p.

Definition 2 *(Minpts-Distance neighborhood of an object p)*: Given the Minpts-distance of object p, *the Minpts- Distance neighborhood of an object p* contains every object whose distance from p is not greater than *Minptsdistance*, i.e. *NMinpts−distance* (*p*) = *{qεD| {p} |d (p, q) ≤ Minpts-distance (p)}*. These objects are called Minpts-nearest neighbors of p.

From figure 1, all the objects present inside the circle of radius 4-Distance of object p are the Minpts-Distance neighborhood of object p.

Definition 3 *(Minpts Reachability distance of an object p)*: Let Minpts be a natural number. The *minpts reachability distance of an object p*, with respect to an object o is defined as *Minpts-Reach distMinpts(p,o)=max(Minptsdistance(o), d(p,o))*.

Definition 4 *(RD of an object p)*: The RD of p is defined as *RDMinpts*(*p*) = 1*/Minpts-Reach-dist(p)* TheRD of an object is inverse of *minpts-reachabiltydistance* which is the distance of Minpts-nearest neighbor of p. If the Minpts-nearest neighbor of object p are very close to object p then it will have very low Mipts-reachability-distance, thus will have a high RD, which will indicate a high density.

Density Based Notion of Cluster:

Definition 5 *(Cluster point)*: An unprocessed point p is a *cluster point* if its RD is largest and its Eps-neighborhood query does not overlap with the points of other cluster. A point become a cluster point only when the above two conditions are satisfied. For a database of huge size, there will be few points which will satisfy the two conditions. In order to select only those points following equation is used:

$$1.2^{*}\ RDp \geq RDo \qquad (1)$$

Here, R*Do* is the RD of previous cluster point. So even if a point has highest RD value amongst the unprocessed points, it must satisfy equation 1, after which the second condition is tested. It is done because the second condition takes more time to test than the first condition. A point has highest RD value amongst the unprocessed points; it must satisfy equation 1, after which the second condition is tested. It is done because the second condition takes more time to test than the first condition.

Definition 6 *(Directly density reachable)*: A point p is *directly density reachable* to point q w.r.t. Eps if pε*NEps* (*q*), where Eps is the radius of the circular region.

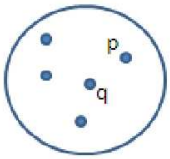

Fig. 2. Directly density reachable

Definition 7 *(Density reachable)*: A point p is *density reachable* from point q w.r.t. Eps if there is chain of points *p*1, *p*2, ..., *pn*, *p*1 = *q* and *pn* = *p* such that *pi*+1 is directly density reachable from *pi*.

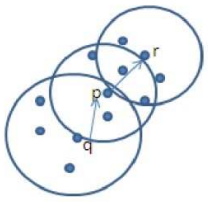

Fig. 3. Density reachable

Definition 8 *(Density connected)*: A point p is *density connected* to a point q from o if there is a point o such that both p and q are density reachable from o.

Definition 9 *(Cluster)*: Let D be a database of points. A cluster C w.r.t. Minpts is a non-empty subset of D satisfying following conditions:

1) For all p, p is density reachable from o w.r.t. Minpts, then pεC. (Maximality)
2) for all p,q C, p is density connected to q by o w.r.t Minpts. (Connectivity)

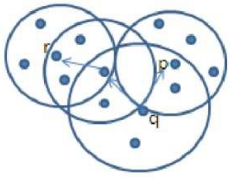

Fig. 4. Density connected

Definition 10 *(Noise)*: A point p is treated as noise if 1.2**RDp* < *RDo* or the Eps-neighborhood query overlaps with clustered points.

IV. ADCA

A. Proposed Approach of Making Eps Adaptive:

For a given Mints value, the points which are inside the cluster will have high RD value than the points which belongs to the edge of cluster. Hence, the RD value will give the position of the point within a cluster.

In this algorithm, the cluster expansion process starts from inside the cluster and proceeds towards the edge. As we move towards the edge, if the Eps values remains same then, its Eps neighborhood query might include points of another cluster which is present in close proximity. This problem is removed by reducing the Eps value as we move towards the edge of cluster. So, by the time the object which is present at the edge of cluster is expanded, its Eps will become so small that it will not include objects of another cluster present in close proximity.

So, it can be said that as we move towards the edge of cluster, Eps value is reduced, or it can be said that as we move towards the edge of cluster the RD value reduces and accordingly the Eps value is reduced i.e.

$$RD \propto Eps \quad (2)$$

$$RD = K*Eps \quad (3)$$

This K is called *"Cluster Constant"* and is used as guide for calculating the Eps value of points of a cluster during the cluster expansion i.e. $Eps = RD/K$ **(4)**

For the cluster point of a cluster its Eps value is equal to its Minpts-nearest neighbor distance. Having got the RD value of that object, the value of K is calculated. Now, this K is used for calculating the Eps values of the objects during the cluster expansion as shown in equation 4. In this way, the Eps value is made adaptive.

B. Heuristic for improving the running time of the algorithm:

During the expansion of a cluster, every point which is present in the Eps-neighborhood are expanded. In [5] it can be seen that, it is not necessary to expand each and every point present in the Eps-neighborhood. Instead, the points present at the boundary of Eps-neighborhood are expanded, which will cover the points, which would have been covered if the points present inside the Eps-neighborhood are expanded.

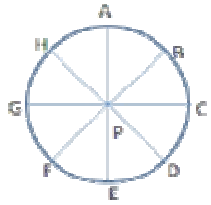

Fig. 5. Cluster region with 8 points at the boundary

Figure 5 shows the abstract view of the logic. If p is the point which is expanded to include points in its Eps-neighborhood, then only the points which are near to A,B,C,D,E,F,G,H are put in the seed for further expansion of cluster. As result, for example, if p includes 30 points in its Eps-neighborhood query then only 8 points at maximum would be put into seeds instead of 30 points which will cover approximately all the points which would have been covered had these 30 points expanded. This heuristic vastly improves the running time of the algorithm.

C. The algorithm:

The algorithm selects a point present at the inner portion of a cluster and proceeds towards the edge of cluster. To find a cluster, the algorithm selects a point which has the highest RD value. This point is treated as cluster-point and it yields a cluster. It then assigns the same cluster ID to all the points which are density reachable to the cluster point in accordance with definition 7. This process is repeated until there are no more points left which is density reachable to the clusterpoint; in that case the algorithm selects another cluster-point according to definition 5.

The following shows the pseudo code of ADCA:

```
ADCA(Set-Of-Points, Minpts)
Initialize the database by calculating RD and Eps (Minptsnearest neighbor distance) of each object.
Sort the objects in descending order of RD.
clusterID = 0;
FOR i FROM 1 TO Set-Of-Points.size DO
        Point = Set-Of-Points.get(i);
        IF the Point is not processed
           rd = Point.getRD();
           IF Point satisfy equation 1
                 eps = Point.getEPS();
                 const = rd/eps;
                 IF Set-Of-Points.regionQuery(Point,eps) does not overlap with other clusters
                        clusterID = clusterID+1;
                        ExpandCluster (Set-Of-Points, Point, clusterID, eps, const)
                 ELSE
                         Set-Of-Points.changeclusterID (Point, noise);
                         Point.processed = true;
            ELSE
                   Set-Of-Points.changeclusterID (Point, noise);
                    Point.processed = true;
           END IF
       END IF
END FOR
END ADCA
```

The Set-Of-Points is the entire database and Minpts is provided by the user. The algorithm starts with the calculation of RD of each point. The points are given Eps value which is equal to the Minpts-nearest neighbor distance. The points are then sorted in descending order of RD values. Initially all the points are unclassified and unprocessed. Set-Of-Points.get(i) returns the ith element of Set-Of-Points. If that point is a cluster-point then, its Eps is found through Point.getEPS() method. Then its *"Cluster Constant"* is calculated which used for calculation of Eps of points during cluster expansion.

```
ExpandCluster (Set-Of-Points, Point, clusterID, eps, const)
tempseeds = Set-Of-Points.regionQuery(Point, eps);
Set-Of-Points.changeclusterID(tempseeds, clusterID);
Put points in the seeds from tempseeds according to the heuristic of section 4.B
```



```
WHILE seeds <> Empty
      currentP = seeds.first();
      rd = current.getRD();
      eps = rd/const;
      temp-seed = Set-Of-Points.regionQuery(currentP, eps);
      Set-Of-Points.changeclusterID(temp-seed, clusterID);
      Delete the points from the seeds which are also member  of temp-seed
      Put points in seeds from temp-seed according to the heuristic of section 4.B
      seeds.delete(currentP);
END WHILE
END ExpandCluster
```

ExpandCluster procedure is called during the expansion of cluster once a cluster is identified. The procedure Set-Of- Points.regionQuery(Point, eps) returns the Eps-neighborhood of Point in Set-Of-Points. These points are assigned the same cluster ID as that of Point. After this, the heuristic is used as shown in section 4.B, in order the select points which are present at the boundary of Eps-region. These points are used as seeds for further expansion of cluster.

D. Parameter Minpts:

Our main objective here is initiating the cluster expansion process from the inside of a cluster and proceeds towards the edge of cluster. In order to satisfy this requirement we need select points having high RD value present at the inner portion of a cluster. For a given cluster, if the Minpts value is low(10-25), then its points will have high RD value at the edge of cluster. But, if the Minpts value is increased(35-50),

We will get more high RD values at the inside of cluster. Hence high Minpts value is recommended in order to begin cluster expansion process form the inside of cluster.

V. PERFORMANCE EVALUATION

Complexity Analysis:

The algorithm spends most of its time in executing the region query. The execution time of other functions can be ignored as compared to the execution time of region query. For a database of size n, the runtime complexity of region query is *O(n)* if no indexing is used. If indexing such as Rtree is used, then the complexity is *O(logmn)*, where m is the depth of tree. In order to calculate RD of each object, the object needs to perform a region query in order to find its Minpts-nearest neighbors. Hence this would take a runtime complexity of *O(logmn)*. For all the objects in the database, this would be *O(nlogmn)*. During the cluster identification and expansion process, for each object region query needs to be performed, hence its complexity would be *O(logmn)*. Hence for entire database the complexity would be *O(nlogmn)*. Hence the resultant complexity of the algorithm is *O(nlogmn)*. But because of the heuristic, the running time of the algorithm reduces significantly.

VI. CONCLUSION

It is logical to use density based clustering algorithms in order to detect clusters in spatial database. But, because of huge size of spatial database, the task of clustering becomes very time consuming. Also the problem of detecting clusters become difficult when there are local clusters present. The algorithm which is proposed here is capable of detecting clusters of arbitrary shape, size with local clusters of different density present in close proximity, with lesser number of input parameter than the exiting density based clustering algorithms. This is done with the use of new concept called *Cluster Constant* and we have used a new measure of calculating the density of a point through *Reachability-density*. We have also used a heuristic in order to improve the running time of the algorithm which is very useful for large dataset. The results of these experiments prove that the proposed algorithm is an improvement over DBSCAN, DDSC and LDBSCAN. The proposed algorithm presents several areas for future research. Further research can be done to see if Minpts parameter can be removed so that the algorithm is free from any user involvement.

REFERENCES

[1] R. H. Gueting, "An introduction to Spatial Database System," The VLDB journal, vol. 3, no. 4, pp. 357–399, Oct 1994.

[2] X. Xu and M. Ester and H.-P. Kriegel and J. Sander, "A distribution Based Clustering Algorithm for Mining in Large Spatial Databases," 14th Int. Conf. on Data Engineering, pp. 324–331, 1998.

[3] M. Ester and H. P. Kriegel and J. Snader and X. Xu , "A density Based Algorithm for Discovering clusters in Large Spatial Databases with Noise," Proc. 2nd Int. Conf. on Knowledge Discovery and Data

Mining, OR, AAAI Press, pp. 226–231, 1996.
[4] M. Ankerst and M. M. Breunig and H. P. Kriegel and J. Sander, "OPTICS:Ordering Points to Identify the Clustering Sructure," Proc. Of ACM SIGMOD Int. Conf. on Management of Data, Philadelphia, P.A.,ACM, New York, pp. 49–60, 1999.
[5] B. Borah and D. K. Bhattacharyya, "An Improved Sampling-Based DBSCAN for Large Spatial Databases," Int. Conf. on Intelligent Sensing, pp. 92–96, 2004.
[6] G. Wei and H. Wu, "LD-BSCA:A Local Density Based Spatial Clustering Algoithm," in IEEE Symposium on Computational Intelligence and Data Mining. IEEE Computer Scoiety, 1999, pp. 291–298.
[7] Duan, Lian and Xu, Lida and Guo, Feng and Lee, Jun and Yan, Baopin, "A local-density based spatial clustering algorithm with noise," Inf. Syst., vol. 32, pp. 978–986, November 2007.
[8] Breunig, Markus M. and Kriegel, Hans-Peter and Ng, Raymond T. and Sander, J, "LOF: Identifying Density-Based Local Outliers," in Proceedings of the 2000 ACM SIGMOD international conference on Management of data, vol. 29, no. 2. New York, NY, USA: ACM, June 2000, pp. 93–104.
[9] Borah, B. and D. K. Bhattacharyya, "DDSC :A Density Differentiated Spatial Clustering Technique," Journal of Computers, vol. 3, no. 2, pp. 72–79, 2008.
[10] Alexander Hinneburg and Er Hinneburg and Daniel A. Keim, "An Efficient Approach to Clustering in Large Multimedia Databases with Noise," AAAI Press, pp. 58–65, 1998.

