ISSN: 2321-7782 (Online)
Impact Factor: 6.047

Volume 4, Issue 7, July 2016

International Journal of Advance Research in Computer Science and Management Studies

Research Article / Survey Paper / Case Study

Available online at: www.ijarcsms.com

Improving Efficiency for Spatial Data Mining Using SCLD Algorithm

Bharath Andugula[1]
Assistant Professor
Department of CSE, GNIT,
Hyderabad, India

Sindhu Boianapalli[2]
Assistant Professor
Department of CSE, KMIT,
Hyderabad, India

Kathi Venkataramana[3]
Assistant Professor
Department of CSE, GIST,
Nellore, India

Abstract: ***spatial databases contain spatial-related information such databases include geographic (map) databases. Spatial data mining is the discovery of interesting characteristics and patterns that may exist in large spatial databases. Clustering, in spatial data mining, aims at grouping a set of objects into classes or clusters such that objects within a cluster have high similarity among each other, but are dissimilar to objects in other clusters. Many clustering methods have been developed. Most of these algorithms, however, substance not allow users to specify real life constraints such as the existence of physical obstacles, like mountains and rivers. Existence of such obstacles could substantially affect the result of a clustering algorithm. The paper proposes an SCLD (Scheduled Clustering for Load Density) algorithm for spatial clustering of large spatial databases. The algorithm divides the spatial area into rectangular cells and labels each cell as dense or non-dense. The algorithm finds all the maximal, connected, and dense regions that form the clusters by a breadth-first search and determine a center for each region.***

Keywords: Data Mining, Density Based algorithms, Spatial Databases, Clustering Algorithms, Breadth-first search.

I. INTRODUCTION

The extensive and world-wide use of spatial datasets has led to the need for generating spatial knowledge. The complexity of dealing with spatial datasets has rendered the traditional methods redundant, which calls for specialized techniques in spatial data mining for extraction and discovering useful information. Coherent tools for extracting information from spatial datasets are very significant to organizations which make decisions on spatial datasets, including NASA, the National Imagery and Mapping Agency and many more.

II. RELATED WORK

2.1 Development Work for Spatial Data Mining Primitives and Algorithms

I have taken the related work of Martin ester and Alexander from melt which reveals study about spatial data mining for conventional database, algorithmic approach and efficient dbms support in following way.

Spatial data mining algorithms heavily depend on the efficient processing of neighbourhood relations since the neighbours of many objects have to be investigated in a single run of a distinctive algorithm. Therefore, it provides general concepts for nearby relations as well as an efficient implementation of these concepts will allow a tight integration of spatial data mining algorithms with a spatial database management system. In this paper, I have taken surrounding graphs and paths and a small set of conventional databases for their manipulations. I showed that distinctive spatial data mining algorithms are well supported by the proposed basic operations.

I discussed here those points which allow only those nearby paths that will significantly minimize the searching space for spatial data mining algorithms. For this I considered nearby indices to enhance the processing of our old-fashioned databases. The potency and efficiency of the discussed approach was evaluated by using an analytical cost model and an extensive experimental study on a geographic database.

Conclusion of this review, defines surrounding graphs and paths and a small set of database primitives for spatial data mining. Research and Studies states that in typical applications the exponential number of all nearby paths can be reduced to a linear number of related nearby paths. I also revealed that spatial data mining algorithms like spatial clustering, their characterization, and classification and behaviour detections are well supported by the proposed operations.

Finally, I have given support for nearby indices to speed-up the processing of our database primitives and can be easily created in a commercial DBMS by using standard functions.

2.3 Development Work for Mining Large Spatial Databases.

Third review I taken of "Xiaowei, xu, hans, peter kriegel, jorg sander ref.in [4]" in the title of A distribution based clustering algorithm for mining in large spatial data bases. In this, I undergone with the new clustering algorithm as Distribution Based Clustering of Large Spatial Databases to discover clusters of this type. Experimental based study reveals that this algorithm, as contrast to partition making algorithms such as algorithm of Clustering using large applications with randomized search, finds arbitrary shape like clusters. In addition, Distribution based algorithm does not require any input parameters, as consider for the clustering algorithm. Distribution based scanning "In [4]" requires two input parameters which may be difficult to be given for large databases. Hence, the productivity of "DBCLASD In ref [4]" on large spatial databases is very impressive when considering its nonparametric nature and its good quality for clusters of arbitrary shape.

In this paper, I consider the task of clustering in spatial databases, regarding the problem of detecting clusters of points which are distributed as belonging to same group also called uniform distribution or random distribution. The problem finds many areas e.g. seismology In [4] (collection of earthquakes clustered containing seismic faults), In 4 minefield detection (grouping mines in a minefield) and astronomy In [4] (grouping stars in galaxies).Content rich spatial database applications requires the following

Requirements for clustering algorithms:

1. Minimal number of input parameters.

2. Discovery of clusters with arbitrary shapes.

3. Good efficiency on huge databases.

None of the well-known clustering algorithms fulfils the combination of these requirements. In this review, I discussed on the new clustering algorithm DBCLASD In [4] which is based on the assumption that the points inside a cluster are uniformly distributed and is quite reasonable for many applications.

Applications of DBCLASD:-

- It works effectively on real databases where the data is not exactly uniformly distributed.
- It dynamically determines the appropriate number and shape of clusters for a database without requiring any input parameters.

III. SPATIAL DATA MINING TECHNIQUES

A. Clustering and Outlier Detection.

Spatial Clustering is a process of collecting spatial objects with similar attributes together. The attributes within the same clusters are similar and outside the clusters are dissimilar. Clustering algorithms are generally divided into four categories: partitioning method, hierarchical method, density-based method and grid-based method.

1. Partitioning Method

Partitioning method generally involves segregation of the cluster objects such that the total deviation of the cluster objects from the centroid is minimized. The most commonly used algorithm in this method is K-Means algorithm.

In K-means algorithm, there are basically 4 steps:

1. Group the data in k sets where k is defined.

2. Select k points at random as cluster centers and find their centroid co-ordinates.

3. Find the distance of each object to the centroids using Euclidean distance formula.

4. Cluster the objects based on minimum distance from centroid co-ordinates.

5. Repeat steps 2, 3 and 4 until same points are obtained in consecutive rounds.

2. Hierarchical Method.

Hierarchical method of clustering involves combining data sets into clusters and clusters into larger clusters. The outcome of hierarchical clustering is a tree, usually called a dendogram. This tree shows the merging process and intermediate clusters. The most commonly used algorithm in this method is Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH).The different types of hierarchical clustering algorithms are BIRCH, ROCK (RObust Clustering using links), Chameleon and CURE (Clustering Using Representatives).

BIRCH:

BIRCH algorithm mainly involves four phases:

1) Loading

2) Condensation

3) Global Clustering and

4) Cluster Refining.

Phase 1: Loading data into memory by building CF tree

The task of this phase is to scan all the data and build a CF-tree which reflects the clustering information of the dataset in a precise manner to the memory limits.

Phase2: Condensation of Initial CF-tree into a smaller tree

This phase is optional. In Phase 3, we use existing global or semi-global clustering algorithms which may create a gap between the results of Phase 1 and Phase 3.Hence, we sometimes require Phase 2 which cushions the results. The task of this phase is to scan the leaf nodes of the initial CF- tree and construct a smaller CF-tree by eliminating outliers and grouping denser sub clusters into larger ones.

Phase 3: Global Clustering

In this phase, existing global or semi-global clustering algorithms are used to obtain good clusters. All the leaf entries are clustered using existing clustering algorithms.

Phase 4: Cluster Refining

After Phase 3, we obtain a set of clusters that captures great distribution pattern in the data. Using the centroids as centers produced by Phase 3 as seeds, this phase redistributes the data points to its closest seed to obtain a new set of clusters. This phase helps to ensure that all copies of given data go to the same cluster and also helps to migrate the points belonging to a cluster.

ROCK (RObust Clustering using links):

ROCK is an algorithm which is based on the concept of links. In this algorithm, similarity of clusters is based on the number of points from different clusters which have neighbours in common. The steps involved in this algorithm are

1) Drawing random sample from dataset.

2) Clustering with links.

3) Label data in disk.

In the first step, random sample from data set is taken.

Now to this sample clustering algorithm is applied that employs links to sample data points. Lastly, the clusters involving the sample points are used to assign the remaining points on the disk appropriately.

CURE (Clustering Using REpresentatives):

CURE is an algorithm that uses partitioning of the datasets. In this process, a constant number of scattered points are chosen. These chosen points are then shrunk towards the centroid of the clusters. After shrinking, these points are used as representatives of the cluster. These clusters are partially clustered. After the clustering is done, instead of single centroid, multiple points from clusters are used as representatives to label the remainder of the data set.

IV. PROPOSED ALGORITHM

4.1. Algorithm *SCLD*

In this section, we describe the proposed algorithm, *SCLD*. The *SCLD* algorithm consists of two phases.

1. Phase 1

In this phase, the algorithm first divides the spatial area into *m*, which is an input, rectangular cells of equal areas by dividing each of the dimensions of the spatial area into the same number, W $\sqrt{m}$, of equal pieces. Then, the algorithm labels each cell as *dense* or *non-dense* (according to the number of points in that cell and an input threshold). The algorithm finds all axial, connected, regions of *dense* cells. Each region is a cluster. Then, the algorithm finds a center for each of the obtained clusters. To find a center for a cluster the algorithm determines the arithmetic mean of all the points in the cluster.

2. Phase 2

In this phase, for each *non-empty*, *non-dense* cell, the algorithm assigns the cell to its neighboring cluster that satisfies the following conditions. First, the addition of the cell to the cluster substance not violates the first condition in the definition of the cluster. That is the number of the points in cluster after adding this cell is greater than or equal to

$$\left(\frac{n}{m} * h\right) * (t+1)$$

Second, if there are more than one cluster that satisfy the first condition, the cell is added to the one whose center in the nearest (using the Euclidean distance) to the mean point of the cell. Finally the algorithm re-computes a center for each extended cluster and outputs the obtained centers as the centers of the clusters in the spatial area.

4.2. *Algorithm for SCLD.*

***Input*:**

1. A set of *N* objects in a spatial area *S*.

2. A square number, *m*, which represents the number of cells in the spatial area such that $m<<N$.

3. A percentage, *h*, used to determine the dense cells according to definition 2.

***Output*:** Clusters with their centers.

***Method*:**

1. W $\leftarrow \sqrt{m}$;

2. Divide the spatial area into *m* rectangular cells by dividing each of dimension of the Spatial area into *w* equal segments;

3. for (*i*=0; *i*<*m*; *i*++)

{

determine for the cell, *ci*, parameters:

- n_{Ci} : the number of the objects in that cell.

- m_{Ci}: the mean object of all objects in that cell.

}

4. $d \leftarrow round\left(\frac{N}{M} * h\right)$;

5. for (*i*=0; *i*<*m*; *i*++)

{

if ($n_{Ci} \geq$ d) then

ci is labeled as *dense;*

else

ci is labeled as *non-dense;*

}

6. *j* = 0;

7. for (*i*=0; *i*<*m*; *i*++)

{

if (*ci* is dense) then,

{

if (*ci* is not processed yet) then,

{

- Construct a new cluster, r*j*, and mark the cell *ci* as an element of r*j;*

- Put the *dense* neighboring cells of *ci* in a list, *Q;*

- While (*Q* is not empty)

{

- Take the first element, *c'*, from *Q*, mark *c'* as an element of the cluster *rj* and add to *Q* the *dense* neighboring cells of *c'*, that have not been processed yet;

}

- *j*++;

}

}

8. for(*i*=0; *i*<*j*; *i*++)

Compute the mean object of the cluster *rj;*

9. for (*i*=0; *i*<*j*; *i*++)

{

- List all neighboring non-empty cells of *rj* in a list , *QN;*

- Sort, *QN* in a descending order, according to the number of objects in the cells;

}

10. while (*QN* isn't empty)

{

- Take the first element, *e*, of *QN*;

- Find all neighboring clusters of *e;*

- Determine which of those clusters can be extended to contain *e* without becoming non-dense;

- Assign *e* to the nearest cluster obtained in previous step;

}

11. Re-compute a center for each extended cluster;

12. Output the clusters with their centers;

V. RESULT ANALYSIS

We describe our algorithm by an example. Suppose that we have a spatial area containing 5,000 objects. Figure 3.2 shows a sketch of the spatial area. We are asked to find the clusters in that spatial area given that m = 36 and h = 0.9. The algorithm

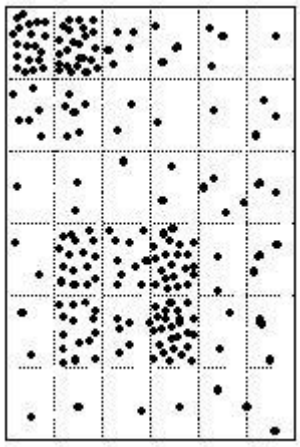

Fig5.1. A sketch of spatial area with objects represented by points

The algorithm works as follow:

1. $w = \sqrt{m} = 6$.

2. The spatial area is divided into 36 equal cells by dividing each of the dimensions into six equal segments. So we obtained the dotted grid structure in figure 5.1. The cells in figure.

5.1 are numbered from left to right and from top to down.

3. For each cell, *c*, the algorithm computes the two parameters *nc*, *mc*. table 5.1 shows values for the number of points for some cells in the grid. We assume that the remaining points in the spatial area are distributed among the remaining cells such that no cell contains more than 100 points.

4. $d = round\left(\frac{N}{M} * h\right) = 125$;

5. Each cell, c, with n_c s labeled as *dense*. So, cells 1, 2, 14, 16, 20 and 22 are the only *dense* cells.

Cell number	Number Of points
1	400
2	400
20	125
21	120
22	600
26	125
27	110
28	600

Table 5.1: The Nc's Parameters from some cells

6. The maximal, connected regions (clusters) of *dense* cells are determined by a breadth-first search. Figure 5.2 illustrates such clusters in the given spatial area with each cluster in a different color.

7. For each cluster obtained in step 6, the algorithm finds the arithmetic mean of the points in the cluster as its center. These centers are shown in Figure 5.2.

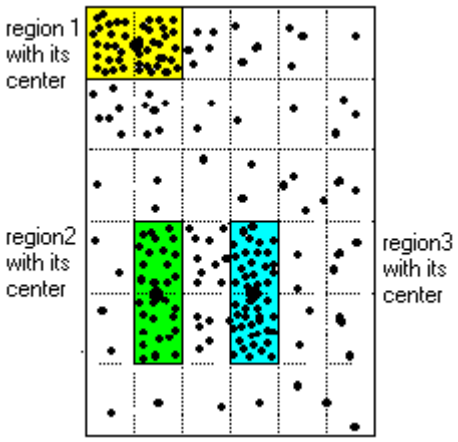

Figure 5.2: The regions are founded after Step6 with their contents

8. In this step, cells 3, 7, 8, 9, 13, 14, 15, 16, 17, 19, 21, 23, 25, 27, 29,30, 31, 32, 33, 34 and 35 that are neighbors to the obtained clusters are put in a list *QN* in a descending order according to the number of the points in each of them. So cell 21 is the first element of *QN* followed by cell 27.

9. The algorithm proceed as follows:

9.1 Cell 21 is eliminated from *QN* because this cell at the top of the list *QN*.

9.2 Regions 2, 3 are determined as neighboring clusters to cell 21.

9.3 Region 3 is determined. In this step region 2 is eliminated because the addition of cell

21 to region 2 violate the first condition in the definition of the cluster (Definition 6).

9.4 Cell 21 is assigned to region 3. Steps 9.1 to 9.4 are repeated to each element of *QN*.

10. Figure 5.3 shows the extended clusters with their new centers.

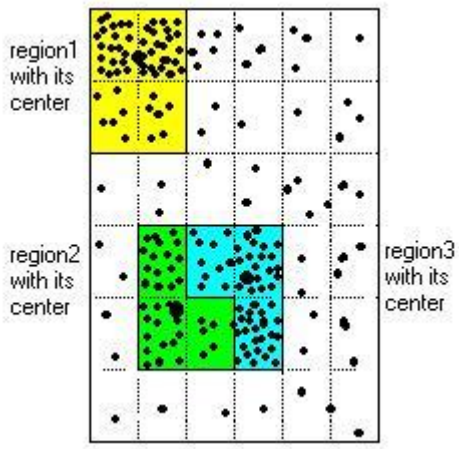

Figure 5.4: The resulted regions with their contents.

VI. CONCLUSION

The proposed an efficient algorithm SCLD for spatial clustering of large spatial databases. The algorithm divides the spatial area into rectangular cells and labels each cell as *dense* (contains relatively large number of points) or *non-dense*. The algorithm

finds all the maximal, connected, dense regions that form the clusters by a breadth-first search and determine a center for each region.

Clustering with obstacles is a novel and interesting problem in the fields of data mining. While the work presented here is sufficient for many applications of clustering with obstacles entities, there still a lot of future work to be considered. We belief that the future work may go in several directions. The first direction is to accommodate the existent algorithms like STING that querying spatial databases to consider the presence of obstacles. It will be useful to extend *SCLD* to solve the clustering problem in a database of *D* dimensions. It will also be useful to consider the clustering problem with obstacles when the obstacles are complex shapes rather than simple polygons

References

1. C. R. Valencio, G. P. Daniel, C. A. D. Medeiros, A. M. Cansian, L. C. Baida, and F. Ferrari, "Vdbscan+: Performance optimization based on gpu parallelism," in Parallel and Distributed Computing, Applications and Technologies (PDCAT), 2013 International Conference on, Dec 2013, pp. 23–28.
2. C. R. Valêncio, T. Kawabata, C. A. de Medeiros, R. C. G. de Souza, and J. M. Machado, "3d geovisualisation techniques applied in spatial data mining," in Machine Learning and Data Mining in Pattern Recognition. Springer, 2013, pp. 57–68.
3. W. Tang and W. Feng, "Parallel map projection of vector-based big spatial data: Coupling cloud computing with graphics processing units," Comput. Environ. Urban Syst., Feb. 2014.
4. A. A. Yıldırım and C. Özdo˘gan, "Parallel wavelet-based clustering algorithm on gpus using cuda," Procedia Computer Science, vol. 3, pp. 396–400, 2011.
5. C. R. Valêncio, C. de Medeiros, F. Ichiba, and R. C. G. de Souza, "Spatial clustering applied to health area," in Parallel and Distributed Computing, Applications and Technologies (PDCAT), 2011 12th International Conference on. IEEE, 2011, pp. 427–432.
6. R. Thapa, C. Trefftz, and G. Wolffe, "Memory-efficient implementation of a graphics processor-based cluster detection algorithm for large spatial databases," Electro/Information Technol. vol. 49401, 2010.
7. D. Luebke and G. Humphreys, "How gpus work," IEEE Comput., 2007.
8. D. Coleman and D. Feldman, "Porting existing radiation code for GPU acceleration," Sel. Top. Appl. Earth, vol. 6, no. 6, pp. 2486–2491, 2013.
9. [9] M. Hemalatha and N. Saranya, "A Recent Survey on Knowledge Discovery in Spatial Data Mining," Int. J. Comput. Sci, vol. 8, no. 3, pp. 473–479, 2011.
10. Y. Zhong, J. Han, T. Zhang, Z. Li, J. Fang, and G. Chen, "Towards Parallel Spatial Query Processing for Big Spatial Data," 2012 IEEE 26th Int. Parallel Distribution Process. Symp. Work. PhD Forum, pp. 2085–2094, May 2012.
11. W. Shuliang, D. Gangyi, and Z. Ming, "Big spatial data mining," 2013 IEEE Int. Conf. Big Data, pp. 13–21, Oct. 2013.
12. W. Jinlin, C. Xi, Z. Kefa, Z. Haibo, and W. Wei, "Research of GISbased Spatial Data Mining Model." WKDD, pp. 159–162, Jan. 2009.
13. Y. Kim, K. Shim, M.-S. Kim, and J. Sup Lee, "DBCURE-MR: An efficient density-based clustering algorithm for large data using MapReduce," Inf. Syst., vol. 42, pp. 15–35, Jun. 2014.
14. G. Sheikholeslami, "Wavecluster: A multi-resolution clustering approach for very large spatial databases," VLDB,1998.
15. P. Liu, D. Zhou, and N. Wu, "VDBSCAN: varied density based spatial clustering of applications with noise," Serv. Syst. Serv. , pp. 1–4, Jun. 2007.
16. J. Owens, M. Houston, D. Luebke, S. Green, J. Stone, and J. Phillips, "Gpu computing," Proceedings of the IEEE, vol. 96, no. 5, pp. 879–899, May 2008.
17. R. Cook, E. Dube, I. Lee, L. Nau, C. Shereda, and F. Wang, "Survey of novel programming models for parallelizing applications at exascale," Technical report, Lawrence Livermore National Laboratory, 2011. 24, Tech. Rep., 2011.
18. Y. Zhao, Z. Huang, B. Chen, Y. Fang, M. Yan, and Z. Yang, "Local acceleration in Distributed Geographic Information Processing with CUDA," 2010 18th Int. Conf. Geoinformatics, pp. 1–6, Jun. 2010.
19. M. Ankerst, M. Breunig, H. -P. Kriegel, and J. Sander. OPTICS: ordering points to identify the clustering structure. In Proc. 1999 ACM-SIGMOG Int.Conf. anagement of Data (SIGMOD'99), pages 49-60, Philadelphia, PA, June 1999.
20. R. Agrawal, J. Gehrke, D.Gunopulos, and P. Raghavan Automatic subspace clustering of high dimensional data for data mining applications. In Proc. 1998 ACM-SIGMOG Int. Conf. Management of Data (SIGMOD'98), pages 94-105, Seattle, WA, June 1998.
21. P. Arabie and L. J. Hubert. An overview of combinatorial data analyis. In P. Arabie, L. Hubert, and G. D. Soete, editors, Clustering and Classification, pages 5- 63. World Scientific Pub., New Jersey, 1996.
22. Char C. Aggarwal, Philip S. Yu. Finding Generalized projected in High Dimensional Spaces. In Proc. 2000 ACM-SIGMOG Int. Conf. Management of Data (IGMOD'00).
23. P. Bradley, U. Fayyad, and C. Reina. Scaling clustering algorithms to large databases. In Proc. 1998 Int. Conf. Knowledge Discovery and Data Mining (KKD'98), pages 9-15, New York, NY, Aug. 1998.

24. Markus M. Breunig, Hans-peter Kriegel, Peer Kroger, Jorg Sander. Data Bubbles: Quality Preserving Performance Boosting for Hierarchical Clustering. In Proc. 2001 ACM-SIGMOG Int. Conf. Management of Data (SIGMOD'01).
25. Markus M. Breunig, Hans-peter Kriegel, Raymond T. Ng, Jorg Sander. LOF: Identifying density-Based Local Outliers. In Proc. 2000 ACM-SIGMOG Int. Conf. Management of Data (SIGMOD'00).
26. T. Cormen, C. Leiserson, and R. Rivest. Introduction to Algorithms. The MIT Press, Cambridge, MA, 1990.
27. J. Han and M. Kamber. Data Mining: Concepts and Techniques. Morgan Kaufmann, 2000.

AUTHOR(S) PROFILE

Bharath Andugula, received the M.Tech in Computer Science and Engineering. Presently he is working as an Assistant Professor in Gurunanak Institute of Technology, Hyderabad. His area of interest is in Data Mining and Network Security.

Sindhu Boinapalli, received the M.Tech in Computer Science and Engineering. Presently she is working as an Assistant Professor in Keshav memorial Institute of Technology, Hyderabad. Her area of interest is in Data Mining and Image Processing.

Kathi Venkataramana, received the M.Tech in Computer Science and Engineering. His area of interest is in Operating Systems, Data Mining and Cloud Computing.

