2009 International Conference on Computer Engineering and Applications
IPCSIT vol.2 (2011) © (2011) IACSIT Press, Singapore

An Efficient Density Based Incremental Clustering Algorithm in Data Warehousing Environment

Navneet Goyal, Poonam Goyal, K Venkatramaiah, Deepak P C, and Sanoop P S

Department of Computer Science & Information Systems,BITS, Pilani, INDIA - 333031

Abstract. Data Warehouses are a good source of data for downstream data mining applications. New data arrives in data warehouses during the periodic refresh cycles. Appending of data on existing data requires that all patterns discovered earlier using various data mining algorithms are updated with each refresh. In this paper, we present an incremental density based clustering algorithm. Incremental DBSCAN is an existing incremental algorithm in which data can be added/deleted to/from existing clusters, one point at a time. Our algorithm is capable of adding points in bulk to existing set of clusters. In this new algorithm, the data points to be added are first clustered using the DBSCAN algorithm and then these new clusters are merged with existing clusters, to come up with the modified set of clusters. That is, we add the clusters incrementally rather than adding points incrementally. It is found that the proposed incremental clustering algorithm produces the same clusters as obtained by Incremental DBSCAN. We have used R*-trees as the data structure to hold the multidimensional data that we need to cluster. One of the major advantages of the proposed approach is that it allows us to see the clustering patterns of the new data along with the existing clustering patterns. Moreover, we can see the merged clusters as well. The proposed algorithm is capable of considerable savings, in terms of region queries performed, as compared to incremental DBSCAN. Results are presented to support the claim.

Keywords: Incremental clustering, DBSCAN, Incremental DBSCAN

1. Introduction

Data Warehouses are an ideal source of data for data mining tasks like clustering, classification, and association rule mining. As data in a data warehouse is refreshed periodically in a batch mode, it becomes necessary to refresh the patterns discovered earlier using different data mining algorithms. Refreshing the patterns must be completed before the data warehouse is made available to users. As the data in data warehouse is increasing coupled with shrinking windows of acceptable downtime, it becomes critical that we refresh the patterns incrementally. Incremental data mining algorithms have been the focus of many researchers in the recent past.

Clustering is an important data mining technique used to find data segmentation and pattern information [1]. It is known as unsupervised learning process, as there is no a-priori knowledge about the data. The problem of clustering is to partition a given set of data objects into clusters such that intra-cluster similarity is maximized and inter-cluster similarity is minimized.

In this paper, we have proposed an incremental version of the popular density based clustering algorithm, DBSCAN (Density-Based Spatial Clustering of Applications with Noise) [2]. DBSCAN is capable of discovering clusters of arbitrary shape. We have used R*-trees [3] as the data structure to hold the multidimensional data that we need to cluster. Incremental DBSCAN [4] is an existing incremental algorithm in which data can be added/deleted to/from existing clusters, one data point at a time. When data is added one point at a time, and when the existing data set is huge, we need to perform lot of region queries on R*-trees. And this problem compounds as the existing data set swells with time.

Our algorithm is capable of adding points in bulk to an existing set of clusters. An interesting application of this kind of scenario is the clustering of earthquake epicenters stored in an earthquake catalogue, where we just keep on adding data. In many data warehouse applications also, we always add data and there is never a need to purge data from the data warehouse. Also our approach can be used to merge clusters of customers of companies in case of mergers and acquisitions

In this new algorithm, the data points to be added are first clustered using the DBSCAN algorithm and then these new clusters are merged with existing clusters, to come up with the modified set of clusters. That is, we add the clusters incrementally rather than adding points incrementally. In this case, while merging the

+ Corresponding author. Tel.: +91-1596-515409; Fax: +91-1596-244183
E-mail address: nt_goyal@yahoo.com

clusters we look at only those new points that are present in the intersection of the old and new data sets and use Incremental DBSCAN clustering algorithm to add these points. So our method gains significantly when the number of intersection points is less.

It is found that the proposed incremental clustering algorithm produces the same clusters as obtained by Incremental DBSCAN. One of the major advantages of the proposed approach is that it allows us to see the clustering patterns of the new data along with the existing clustering patterns. Moreover, we can see the merged clusters as well. Experimental results are presented to establish the proposed algorithm and comparisons are done with incremental DBSCAN.

The rest of the paper is organized as follows. We discuss related work on incremental clustering algorithms in section 2. In section 3, we present the proposed algorithm. Experimental set up and performance evaluation is given in section 4. Some directions for future research are given in section 5.

2. Related Work

With the increase in size of databases and the storage capacity, the volume of data used in data mining is constantly increasing there by necessitating the need for efficient algorithms. Applying these algorithms, with all their efficiency, on the entire database after each update is prohibitively expensive. This gave birth to idea of incremental clustering algorithms. In incremental algorithms, the cost of update is proportional to the size of the update. Many researchers, in the recent past, have developed incremental algorithms for various data mining tasks. Algorithms for finding association rules incrementally are listed [5-7]. Incremental algorithms for classification allow us to incorporate the feedback into our model efficiently. Incremental algorithms for classification are reported in [8-10].

Incremental clustering is the process of updating an existing set of clusters incrementally rather than mining them from the scratch on each database update. A brief overview of work done on incremental clustering algorithms is given next. COBWEB was proposed by Fisher [11]. It is an incremental clustering algorithm that builds taxonomy of clusters without having a pre-defined number of clusters. Gennary et al. [12] proposed CLASSIT which associates normal distributions with cluster nodes. The main drawback of both COBWEB and CLASSIT is that they results in highly unbalanced trees.

Charikar et al., [13] introduced new deterministic and randomized incremental clustering algorithms while trying to minimize the maximum diameters of the clusters. The diameter of a cluster is its maximum distance among its points and is used in the restructuring process of the clusters. When a new point arrives, it is either assigned to one of the current clusters or it initializes its own cluster while two existing clusters are combined into one. Ester et al. [14] presented Incremental DBSCAN suitable for mining in a data warehousing environment in 1998. Incremental DBSCAN is based on the DBSCAN algorithm which is a density based clustering algorithm. It uses R* Tree as an index structure to perform region queries. Due to its density-based qualities, in Incremental DBSCAN the effects of inserting and deleting objects are limited only to the neighborhood of these objects. Incremental DBSCAN requires only a distance function and is applicable to any data set from a metric space. However, the proposed method does not address the problem of changing point densities over time, which would require adapting the input parameters for Incremental DBSCAN over time. Another limitation of the algorithm is that it adds or deletes one data point at a time.

Fractal clustering (FC) was proposed Barbara and Chen[15]. FC belongs to the class of Grid based techniques for clustering. The concept of HFD (Hausdorff Fractal Dimension) is fundamental to FC algorithm for numeric attributes, which works with several layers of grid. FC starts with initializing of k clusters, where k is determined using an initialization threshold and data sample. Then FC tries to add all the data incrementally.

Barbara [16] has given the requirements for clustering streaming data. Aggarwal et al. [17] uses the data summarization method BIRCH in the context of mining data streams. It combines online micro clustering with offline macro clustering to cluster data streams.

An agglomerative hierarchical clustering using partial maximum array and incremental similarity computation method is presented in [18]. Guillermo et al. [19] proposed the GLC+ algorithm for very large mixed incomplete data sets in order to obtain partitions in connected sets. The main advantage is that it handles data in any space. The GLC+ initially clusters objects in sub-clusters, which will be connected sets and then gives them a final clustering in connected sets, both the phases mentioned are incremental. Incremental hierarchical clustering algorithm GRIN was proposed by Chen et al. [20] for numerical datasets, which is based on gravity theory. Agglomerative incremental hierarchical clustering (IHC) algorithm [21] utilizes a restructuring process, working in a bottom up fashion, while preserving homogeneity of the clusters and monotonicity of the cluster hierarchy.

Nassar et al. [22] proposed a new scheme to maintain data bubbles incrementally. By using incremental data bubbles, a high-quality hierarchical clustering is quickly available at any point in time. GenIc, a single pass generalized incremental algorithm for clustering, was given by Gupta et al. [23]. An incremental clustering algorithm based on SWARM intelligence is given in [24].

3. The Proposed Algorithm

We propose an incremental density based clustering algorithm which is capable of adding points in bulk to an existing set of clusters. The main concept of this algorithm is that we first cluster the points to be added and these new clusters are then merged with existing clusters. The steps involved in the algorithm are as follows: 1) the new points to be added are clustered using the DBSCAN algorithm[2], 2) the new data points which intersect with old data points (see Figure 1) are determined, 3) for each intersection point from the new data set, we use incremental DBSCAN algorithm [4] to determine its new cluster membership, 4) The cluster membership of the remaining new points is then updated. Before we proceed on to give more details of the algorithm, we need to give a few definitions:

Definition 1: (intersection data points) Let D & D' be databases of points and p be some point such that $p \in D'$ and $p \notin D$. We define p as an intersection point if $\exists$ at least one object $\in D$ in $N_\varepsilon(p)$.

Definition 2: (affected data points [4]) Let D be a database of points and p be some point (either in D on not in D). We define a set of points in D affected by the insertion of p as

$$Affected_D(p) = N_\varepsilon(p) \cup \{q \mid \exists\ o \in N_\varepsilon(p) \wedge q_{D \cup \{p\}}\ o\}$$

The pseudo code for the proposed algorithm is given below:

```
NewIncrementalDBscan ( Old RTree, New RTree, Point)
   Get the Neighborhood Points of the Point in Both the RTree
   For Every Neighborhood Point of the Given Point
       If the Neighborhood Point is a core Point
          If the Neighborhood Point was not a Core Point earlier
              If the Neighborhood Point belongs to a Cluster
                 Add the Neighborhood point to a List 'Change' to process Later
              Else
                 Mark to Change the Cluster Later
          Get the Neighborhood Points of the Neighborhood Point and add them into a list
                Np
          For every Point in the List Np
              If the Point is Noise
                 Mark to Change Later
              If the Point Belongs to a Cluster
                 Add the point to the List 'Change' to process Later
       Else
          Add the Neighborhood Point to the List 'Change' to process Later
   IF No New Core Points
       Assign the Cluster ID as Noise
   Else
       IF 'Change' List has No elements Then
          Assign all Marked Points to a New Cluster
       Else If Change List has Only one Element
          Add the Point to the 'Change' Cluster
       Else
          Update All the CusterID of the Points in the 'Change' List
End NewIncrementalDBSCAN
```

In case of addition of new data points, only the following transitions are possible as we are using the same ε and minpoints: Noise ➔ Border/core, Border ➔ Core, & Core ➔ Core.

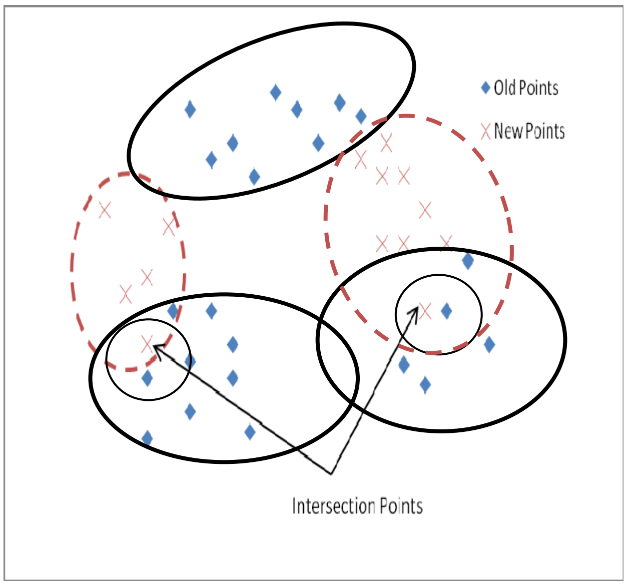
The set of existing clusters are referred to as *old clusters* and the clusters of points to be added are referred to as *new clusters*. We have considered the following cases for an intersection point:

Case 1: Noise point in new data set. We examine the affected points to see if it is density reachable from a core point of any old cluster. If so, its cluster membership is changed. If it becomes a core point, then formation of a new cluster may happen.

Case 2: Core point in new data set. We examine the affected points to see if it is density reachable from a core point of any old cluster. If so, the two clusters are merged.

Case 3: Border point in new data set. If it is becoming a core point, the two clusters in question are merged. If the border point is not becoming a core point, then it retains its cluster membership. In all the above cases, if the new point is not an intersection point, then its cluster membership will not change.

Case 4: Noise point in old data set. If it becomes a border point of a new cluster, then it will be absorbed in the cluster. If it becomes a core point, then formation of a new cluster or merging of clusters may happen.



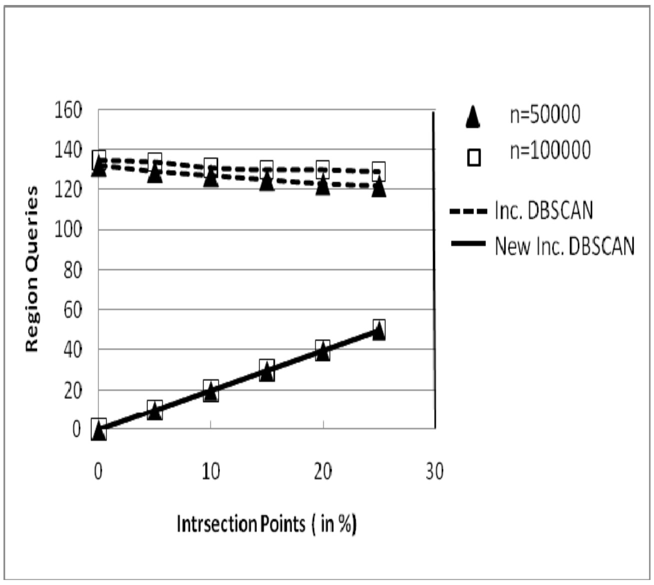


Fig. 1. Intersection points

Fig. 2. Region Queries (New points = 100)

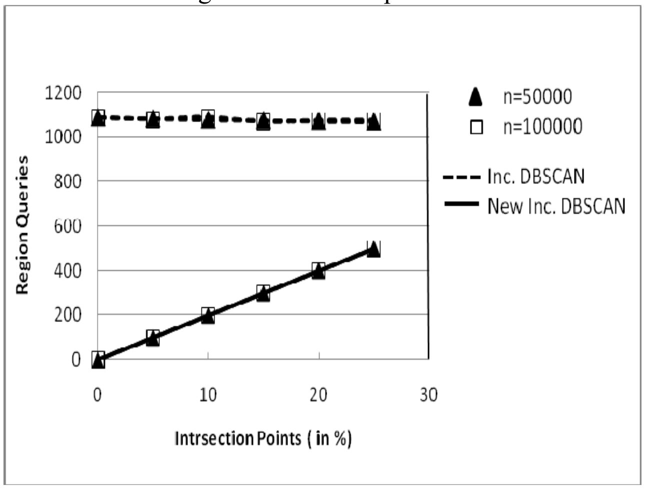


Fig. 3. Region Queries (New points = 1000)

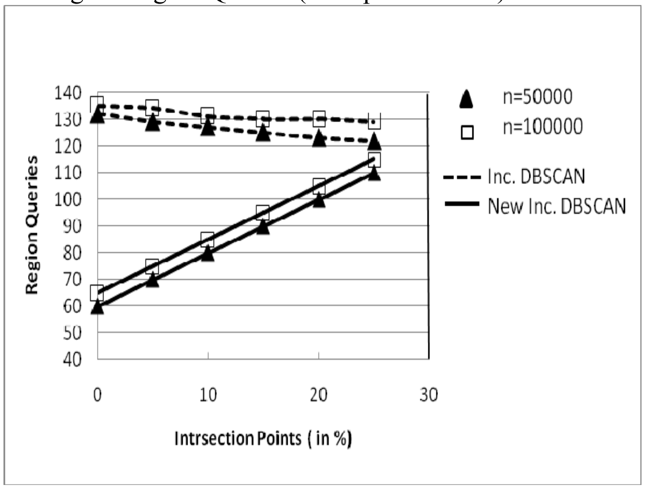


Fig. 4. Region Queries with 2% Noise (New points=100)

4. Results and Discussions

We have given an algorithm for adding points in bulk to an existing set of clusters. This kind of situation arises in case of data warehouses where data is appended periodically during refresh cycles. The proposed algorithm has many applications. It can be applied to Census data, where new data points are added from time to time. Also our approach can be used to merge clusters of customers of companies in case of mergers and acquisitions. The main advantage of the approach is that we are able to see the old clusters, the new clusters and the merged clusters simultaneously.

We have compared the performance of our algorithm with the Incremental DBSCAN algorithm [4] in terms of regions queries that need to be performed on the R*-tree. Results are presented for adding different number of points to a set of n=50000 and n=100000 points with different levels of noise. For all our results, we have take ε=5 and minpoints=3.

From figures 2-4 it can be very clearly seen that the proposed algorithm is scalable with increasing old data points. This is especially true when there is no noise as can be seen from figures 2 & 3. In case of noise (2%), it is found that with increasing number of intersection points, the number of region queries tend to approach the region queries required by Incremental DBSCAN. We can conclude from the results presented that our approach is considerably efficient as compared to Incremental DBSCAN when the number of intersection points are less and when there is little or no noise.

5. Future Research Directions

It would be challenging to delete clusters incrementally from an existing set of clusters. In data warehouses, many times when new data is added during the refresh cycle, old data is purged. It would be good if we can delete clusters belonging to a particular time period and see its effect on the existing clusters.

6. References

[1] L. Kaufman, and P. J. Rousseeuw, "Finding groups in data: An introduction to cluster analysis, " John Wiley & Sons, 1990.

[2] M. Ester, H. –P. Kriegel, J. Sander, 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, Portland, OR, 1996, pp. 226-231.

[3] N. Beckmann, H. –P. Kriegel, R. Schneider, and B. Seeger, "The R*-tree: An efficient and robust access method for points and Rectangles," Proc. ACM SIGMOD Int. Conf. on Management of Data, Atlantic City, NJ, 1990, pp. 322-331.

[4] M. Ester, H. –P. Kriegel, J. Sander, M. Wimmer, and X. Xu, "Incremental clustering for mining in a Data Warehousing environment," Proc. 24th Int. Conf. on Very Large Databases (VLDB'98), New York, USA, 1998, pp. 323-333.

[5] D. W. Cheung, J. Han, V. T. Ng, and C. Y. Wong, "Maintenance of discovered association rules in large Databases: An incremental updating technique," 12th Int. Conf. on Data Engineering (ICDE'96), 1996.

[6] C. –H. Lee, C. –R. Lin, and M. –S. Chen, "Sliding Window filtering: An efficient algorithm for incremental mining," **Proc. of the 10th Int. Conf. on Information and Knowledge Management.** CIKM01, Atlanta, Georgia, USA, November 5-10, 2001.

[7] Z. Geng, and Q. Zhu, "Incremental rules mining for information compression matrix algorithm," Proc. of the 5th World Congress on Intelligent Control and Automation. June 15-19, 2004.

[8] R. Xiao, J. Wang, and F. Zhang, "An approach to incremental SVM learning algorithm, " 12th IEEE Int. Conf. on Tools with Artificial Intelligence, 2000.

[9] Z. Wenhua, and M. Jain, "A novel incremental SVM learning algorithm," Proc. of the 8th Int. Conf. on Computer Supported Cooperative work in Design, IEEE, 2003.

[10] Y. Liu, Q. He, and Q. Chen, "Incremental batch learning with Support Vector Machines," In Proceedings of the 5th World Congress on Intelligent Control and Automation, June 15-19, IEEE, 2004.

[11] D. Fisher, "Knowledge acquisition via incremental conceptual clustering," Machine Learning, vol. 2, 1987, pp. 139-172.

[12] J. Gennary, P. Langley, and D. Fisher, "Model of Incremental Concept Formation," Artificial Intelligence Journal, vol. 40, 1989, pp. 11-61.

[13] M. Charikar, C. Chekuri, T. Feder, and R. Motwani, "Incremental clustering and dynamic information retrieval," 29th Symposium on Theory of Computing, 1997, pp. 626—635.

[14] M. Ester, H. –P. Kriegel, J. Sander, M. Wimmer, and X. Xu, X, "Incremental clustering for mining in a Data Warehousing environment," Proc. of the 24th Int. Conf. on Very Large Databases (VLDB'98), New York, USA, 1998, pp. 323-333.

[15] D. Barbara and P. Chen P, "Using the fractal dimension to cluster datasets," Proc. of the 6th ACM SIGKDD, Boston, MA, 2000, pp. 260—264.

[16] D. Barbara, "Requirements for clustering data streams," SIGKDD Explorations, vol. 3, 2002, pp. 23-27.

[17] C. C. Aggarwal, J. Han, J. Wang, and P. S. A. Yu, "A framework for clustering evolving data streams," VLDB'03, 2003, pp. 81-92.

[18] J. Sung, and T. S. Kim, "An agglomerative hierarchical clustering using partial maximum array and incremental similarity computation method," IEEE Int. Conf. on Data Mining (ICDM '01), San Jose, California, 2001, pp. 1129-1202.

[19] S. –D. Guillermo, and J. R. Shulcloper, "A clustering method for very large mixed data sets," IEEE Int. Conf. on Data Mining (ICDM '01), San Jose, California, 2001.

[20] C. Chen, S. Hwang, and Y. Oyang, "An incremental hierarchical data clustering algorithm based on gravity theory," 6th Pacific Asia Conference on Knowledge Discovery and Data Mining, 2002.

[21] D. H. Widyantoro, R. I. Thomas, and J. Yen, "An incremental approach to building a cluster hierarchy," IEEE Int. Conf. on Data Mining (ICDM '02), 2002, pp. 705-708.

[22] S. Nassar, J. Sander, and C. Cheng, "Incremental and effective data summarization for dynamic hierarchical clustering," Proc. of the 2004 ACM SIGMOD Int. Conf. on Management of data, 2004, pp. 467-478.

[23] C. Gupta, and R. Grossman, "GenIc: A single pass generalized incremental algorithm for clustering," SIAM International Conference on Data Mining (SDM 04), 2004.

[24] Z. Chen, Z and Q. –C, Meng, "An incremental clustering algorithm based on SWARM intelligence theory," Proc. of the 3rd Int. Conf. on Machine Learning and Cybernetics, Shanghai, 26-29 August, 2004.