Novel and Efficient Clustering Algorithm Using Structured Query Language

Suresh L
Professor,Cambridge Institute of Technology,
Visveswaraiah Technological University
Bangalore, India
suri akls@vahoo.com

Dr. Jay.B.Simha
Chief Technological Officer,
Abiba Systems,
Dr.MGR University.
Chennai, India

Dr.V.Rajappa
Professor & Dean,
Cambridge Institute of Technology,
Bangalore, India

Summary
Clustering becomes an indispensable requirement while dealing with immense volume of data. Since Database Management tools does not provide an inbuilt mechanism to cluster datasets of higher magnitude it inevitably requires an external module to perform the same. This external module should be devised specifically to deal with the data extracted from the data source. There exists myriad of techniques for clustering and the conventional approaches deficits like excessive time and computational complexity. Computational complexity becomes a factor of consideration when data is extracted voluminously and the process of clustering and filtration is performed as a subsequent separate operation. The concept proposed is crafted with an object of eliminating the inherent redundancies in the accustomed practice. The proposed solution relies on exploiting the processing power of the Database Management tool by streamlining the SQL used for data extraction. And hence a procedure is formulated that can combine data retrieval and clustering to one single operation and leave it to DBMS without letting it to dissipate to the adjoining tiers.

Key words:
Data mining, Clustering Algorithms, Relational DBMS, Structured Query Language (SQL).

1. Introduction

In regard to the advent of information technology and its varied applications, and the cost of storage dropping down from expensive to cheap, gathering massive data has never been as easy as in the past few decades. The patterns that were formerly unknown are now being discovered massively within fractions of seconds, eventually leading to better decision making, thereby phenomenal competitive gains. It has been proved in the recent years that Online Analytical Data processing (OLAP) alone does not suffice for the analysis of data, thus leading to the fact that an automated process is essential to discover interesting and hidden patterns in data [1],[4]. Data mining techniques that are applied in real world databases are thus been studied extensively.

The process of characterization through large quantity of data and choosing the appropriate information is termed as Data mining. In technical terms, locating correlations or patterns from huge number of fields in a relational database loaded with immense data is called data mining.

Apart from the traditional users such as the business intelligence organizations and financial analysts who still gain tremendously from data mining; recently, the utilization of Data mining techniques by the scientists for information abstraction from the huge number of data sets resulting from contemporary experimental methods has increased. Typically the process of obtaining the knowledge concealed in large data sets is entitled as Data Mining [5] [11].

In an attempt to permit an adequate exploration of data and to deal with the enormous amounts of data that an organization has collected over the years [7], high performance is a key factor for data mining systems. There are two keys to success in data mining. Coming up with an accurate formulation of the problem to be solved is the first key. A concentrated statement generally results in the finest payoff. The second key is using the right data. We may require changing and merging it in considerable ways [6], subsequent to preferring from the data obtainable to us or possibly buying external data. In order to devour an entity at an instant, data mining algorithms are designed. With relative simplicity, traditional data mining algorithms can be imposed initially. Since it eradicates the requirement for data mining algorithms to do considerably [8], it enhances scalability subsequently.

In the presence of large amounts of data, the data mining algorithms confront the very important issue of scalability. The existing various types of data base technologies contain relational database technologies, hierarchical database technologies, and other types of database technologies as well. A set of inter-related tables, containing rows and columns, have been incorporated by a relational database. It will be a complicated mission to administer large data sets without DBMS support. The DBMS is mechanically taking care of space management, fault tolerance, secure access, concurrency control, etc., for the majority part. The time to export it to a workstation can be considerably important [18], even though it is probable to competently cluster an extremely large data set outside a relational database.

In data mining, Clustering is one of the most significant tasks [3]. Clustering divides a database into different groups. A number of algorithms appropriate to extremely large databases are available, and a few of those focus on high-dimensional data. Clustering is a division of data into groups of comparable objects. Fewer clusters symbolizing data essentially misplaces certain fine details, however attains simplification. In data mining applications for instance scientific data exploration, information retrieval and text mining, spatial database applications, Web analysis, medical diagnostics, computational biology, and many others, clustering plays a crucial role. Data mining enhances clustering of the complications of extremely large datasets with many attributes of dissimilar types. This imposes distinctive computational necessities on pertinent clustering algorithms. Various clustering algorithms, meeting these necessities, have been newly emerged and were effectively applied to real-life data mining problems [9].

In the data mining literature, a lot of competent clustering algorithms exist. A few of them are Hierarchical Methods, Partitioning Methods, Probabilistic Clustering, K-means Methods, Density-Based Algorithms, Density-Based Connectivity Clustering, Density Functions Clustering, Grid-Based Methods, Scalable Clustering Algorithms, Algorithms for High Dimensional Data [14] [13] etc. The disk access has been lessened by the majority of the approaches and doing most of the work in main memory. Regrettably, executing a real DBMS is extremely tough in many of those algorithms where the programmer requires to be concerned about storage management, concurrent access, memory leaks, fault tolerance, security and so on [16].

In this paper, a novel clustering algorithm with relational databases using SQL, the standard interactive and programming language for querying and modifying data and managing data in relational databases, has been principally focused. This command language permits the recovery, inclusion, updating, and removal of data, and too executing management and administrative functions.

The rest of the paper is organized as follows; Section 2 presents a brief review of proposed works. Section 2.1 presents the proposed algorithm framework. The experimental results are given in Section 3 and conclusions are summed up in Section 4.

1.1 Structured Query Language (SQL)

For accessing and manipulating relational database content, Structured Query Language (SQL) is an ANSI standard. It is extensively used in industry and is supported by major database management systems (DBMS). For textual and numeric data, DBMS are first and foremost used.

These textual and numeric data have been divided into records and fields [15]. SQL is available in every relational DBMS. To turn into fairly comprehensive and complex query languagewhere the features are automatically managed or it can be adjusted by the database application programmer, SQL has been rising over the years. Furthermore, to interact with a relational DBMS, it is the distinctive approach. Because, this approach recuperates the speed, storage management, concurrent accesses, etc. [16]. With the facility to recover records independently from the database, carry out all computation in main memory, and write any required changes back to the databases [17], the basic idea employs the DBMS as a simple storage.

The application programmer has been secluded by SQL from internal mechanisms of the DBMS. In a relational database and dissimilar subsets of data points, numerous data sets are stored. To do inside a DBMS with SQL query, dimensions are more flexible, faster and easier than outside with alternative tools [16]. Application downtime, increased scalability and performance, and flexible security controls have been condensed by SQL. To deploy, handle, and optimize enterprise data and analytical applications, SQL creates it simpler and easier. Many benefits in terms of speed, scheduling capabilities, data analysis, security, scalability, and reliability have been presented by the implementation of SQL in data mining. To execute complex mathematical operations, SQL has a few restrictions. To manipulate matrices, SQL never provide arrays and functions. As a result, if feasible to express in SQL, many computations can turn into unwieldy. SQL queries incur higher overheads than directly accessing the file system with an elevated programming language like Java [16].

2. Proposed Algorithm

The efficient database management systems have been very important assets for management of a large amount of data and especially for effective and efficient retrieval of particular information from a large collection whenever needed.

There are many existing clustering algorithms which work more efficiently to cluster the large datasets. One of the important among them is k-means clustering algorithm. If there are huge amount of records in the dataset, the records are to be clustered very efficiently according to their related fields. Cluster a large datasets using k-means algorithm with relational database seems to be more iterative and requires more time. But using SQL the massive datasets can be clustered very effectively.

So we have proposed an algorithm namely SQL based clustering algorithm which clearly explains about clustering the large data set using SQL with minimum time duration. The goal of clustering is to find groups that are very different from each other, and whose members are very similar to each other. SQL queries have been used to retrieve data in the form of contingency table, which is further used to extract patterns in the data [9]. The number of fields present in the large datasets can be clustered by using SQL queries. The initial set of the related groups are efficiently clustered by using the SQL with minimum amount of time. Then, with the help of our proposed framework, the datasets can be clustered very effectively.

2.1 SQL Based Clustering Algorithm

The enormous amount of datasets has been clustered very efficiently by using this SQL based clustering algorithm. The efficiency of this algorithm is reaching peak improvement by means of reducing the iteration time and it is the basic limitation in other clustering techniques. Also, it is more efficient only if the key fields chosen for clustering have obtained significant redundant data, the SQL algorithm will perform better than other conventional algorithms.

The pseudo code and its explanation is given below,

Assumptions

T_d → table contains input data sets.
T_g → table contains grouped data sets.
T_{gd} → table contains index and Euclidean distance of grouped data sets.
T_c → table contains clustered data sets.
NF → Number of fields

The required number of fields (NF) mainly depends on the amount of data which is given by the user. As the amount of data increases the number of usage of fields will also increases.

In the first stage of the clustering the redundant data in the different fields have been merged by using a query q which is given by the equation as

$$q = \sum_{i=0}^{NF-1} f_1 * pow(10, i)$$

Where f_1 is the data field.

```
INSERT INTO Tg
SELECT + q +
from Td
GROUP BY + g;
```

The merged fields and their id's are grouped and finally it will be inserted into the table T_g. Thus only the id values and their corresponding field values will reside in the table T_g.

Ed →Euclidean Distance

```
INSERT INTO Tgd(Index1, Index2, value1, value2,
Euclidean Distance)
SELECT g1.id, g2.id, g1.f1, g2.f1, Ed (g2.f1-g1.f1)
FROM Tg
WHERE (([g1].[id]+1) = [g2].[id])
ORDER BY Ed (g2.f1-g1.f1)
```

The Euclidean distance is calculated by the difference between two consecutive values of grouped fields. The values will be sorted in ascending order and it will be inserted into the table T_{gd} which contains indices, values and corresponding Euclidean distances.

Then the SQL script to get the values from the table T_{gd} is given by

```
SELECT Index1, value1, value2, Euclidean Distance
from Tgd
```

This SQL based algorithm is used to store the clustered data into the concluding table T_c.

Assumptions

I_{DS} →Vector having the index of the grouped data set.
S_D →Vector having grouped data set. I_C →Index of cluster
K → number of clusters.
De → Euclidean distance of grouped data set.

```
Set rs as RecordSet;
```

The values of the table T_{gd} is taken as a record set for further programming.

```
d ← 0;
g ← 0;
a ← 0;

while (rs != empty) do
        IDS[0] ← rs.getInt(1);
        IDS[1] ← IDS[0] + 1;


        SD[0] ← rs.getInt(2);
        SD[1] ← rs.getInt(3);
        De    ← rs.getInt(4);
        if g < K then
                if De == d then
                        if (a !=  IDS[0] ) then
                                g ← g + 1;
                        end if
                else
                        g ← g + 1;
                end if
                for each element i in SD
```

```
                    insert into T_C values (g, S_D[i]);
                    I_C[g] ← S_D[1];
            end for
    else
            s ← Φ; where Φ is a greater value
            v ← 0;
            ind ← 0;
            for each element i in S_D
                    for each element j in I_C
                            v ← abs(I_C[i] – S_D[j]);
                             if s > v then
                                    s ← v;
                                    ind ← j;
                             end if
                    end for
                    insert into T_C (g, S_D[i] )
                    I_C[ind] ← S_D[i];
            end for
    end if
    d ←De;
    a ← I_DS[1];
end while
```

The indexes of the grouped dataset are taken from the record set. Then the values of the different Euclidean distance are assigned in different clusters. This operation will be performed up to K number of clusters. The remaining values are assigned to the cluster which is having minimum difference between the Euclidean distances of the two values. This will be continued up to all the values in the record set. Eventually, the clustered data are inserted into the concluding table T_c.

3. Experimental Results

This section contains an extensive experimental evaluation. We have implemented the SQL based clustering algorithm of the proposed framework in Java. Using this algorithm, the redundant data in the selected fields have been grouped initially with minimum amount of time. The results have been analyzed by comparing the K-means algorithm with our proposed algorithm.

The main limitation of k-means implementations is that the datasets needs to be scanned several times. So this algorithm requires more time for clustering than our proposed solution. In this, we compare the time to export data sets by the K-means iteration. We emphasize that the iteration operation is performed few times for one data set, whereas K-means iterations are repeated many times. But our algorithm performs better than k-means and results compared with the k-means algorithm by means of reducing the computational complexity and time consumption. We have compared both the SQL Query clustering and K-means clustering results with large throughputs and those experimental results are listed in charts as follows. In the chart, the x-axis is represented by the quantity of datasets and y-axis is represented by the time needed for clustering those datasets.

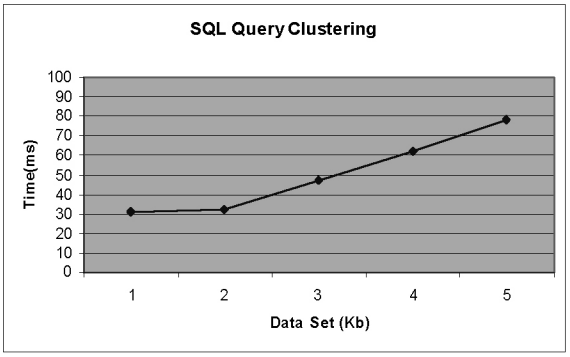

Fig.1 SQL based clustering algorithm

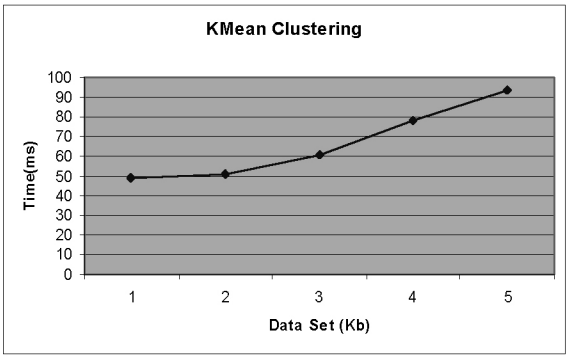

Fig. 2 K-Means clustering algorithm

4. Conclusion

Clustering large datasets has some important issues like excessive time, computational complexity and so on. The existing works have not specified the effectiveness of database management systems using SQL in clustering algorithms. So we have proposed an efficient clustering algorithm in this framework by using SQL. The SQL algorithm is used to cluster a large datasets very efficiently than the other existing clustering techniques, when the key fields chosen for clustering have obtained significant redundant data. This algorithm gives the better results when compared with the k-means algorithm by the way of reducing the computational complexity and time consumption.

References

[1] J. P. Bigus., "Data Mining with Neural Networks", McGraw-Hill, 1996.

[2] T. Zhang, R. Ramakrishnan, and M. Livny., "BIRCH: An efficient data clustering method for very large databases", In ACM SIGMOD Conference, pp. 103-114, 1996.
[3] Fayyad U., Piatetsky G., Smyth P., Uthurusay R., "Advances in knowledge discovery", AAAI press, Cambridge,1996.
[4] T. M. Mitchell., "Machine Learning", McGraw-Hill, 1997.
[5] Ralf Rantzau and Holger Schwarz, "A Multi-Tier Architecture for High-Performance Data Mining", University of Stuttgart, Institute of Parallel and Distributed High-Performance Systems (IPVR), Breitwiesenstr, pp. 20-22, 1999.
[6] Herbert A. Edelstein, "Introduction to data mining and knowledge discovery", 3rd edition, published in two crows corporation, 1999.
[7] J. Zytkow and S. Gupta , "Mining Medical Data Using SQL Queries and Contingency Tables", Proceedings of the Fifth European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD '01), 2001.
[8] Netz, A. et al, "Integrating Data Mining with SQL Databases: OLE DB for Data Mining", Precedence of the 17th International Conference on Data Engineering, pp. 379-387, 2001.
[9] Pavel Berkhin, "Survey of Clustering Data Mining Techniques- technical report", Accrue software, 2002.
[10] Lorinda Visnik, "Clustering Techniques", A technical White paper, 2002.
[11] Ramesh Natarajan, Radu Sion, Chid Apte, and Inderpal S. Narang, "A Grid-based Approach for Enterprise-Scale Data Mining", 2004.
[12] Carlos Ordonez, "programming the k-means clustering in SQL", conference on knowledge discovery and data mining, 2004.
[13] Yihong Dong, Yueting Zhuang, "Fuzzy hierarchical clustering algorithm facing large databases", Fifth World Conference on Intelligent Control and Automation, WCICA 2004, Issue- 15-19, Vol. 5, pp. 4282 - 4286, 2004.
[14] Zhijie Xu, Laisheng Wang, Jiancheng Luo, Jianqin Zhang, "A modified clustering algorithm for data mining", Proceedings of the IEEE International on Geoscience and Remote Sensing Symposium, ,Vol. 2 , pp. 25-29 , 2005.
[15] Wolfram, Dietmar, "Applications of SQL for Informetric Data Processing", Proceedings of the 33rd conference of the Canadian Association for Information Science, 2005.
[16] Carlos Ordonez, "Integrating K-Means Clustering with a Relational DBMS Using SQL", IEEE transactions on knowledge and data engineering, Vol. 18, No.2, pp. 188-201, 2006.
[17] Zou, B., Ma, X., Kemme, B., Newton, G. and Precup, "Data mining using Relational Database Management Systems", Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), 2006.
[18] Carlos Ordonez, "k-means clustering using structured query language statements and sufficient statistics", April 2008.

About the Authors:

Suresh L is working as a Professor & Head, Department of Information Science and Engineering at Cambridge Institute of Technology, Bangalore, INDIA. He has received his B.E., M.S. in Computer Science & Engineering and pursuing his Ph.D at Dr. M.G.R.University, Chennai, Tamilnadu, INDIA. His Research area is Data Mining.

Dr. Jay B.Simha is working as a Chief Technology Officer at Abiba Systems, Bangalore, INDIA. He has received Ph. D. degree in Computer Science from Bangalore University in 2003. His research areas are Data Mining, Fuzzy Logic and Artificial Intelligence.

Dr. Veluru Rajappa is working as a Professor and Dean Academics at Cambridge Institute of Technology, Bangalore, INDIA. He has received Bachelor, Master Degree in Elecronics Discipline from Gulbarga University, Gulbarga and Ph. D. Degree in Graph Theory and its Computer Applications. He has been involved in the organization of a number of conferences and other courses at CITech., Bangalore. His main research interests are in Graph Theory, Data Base Management System, Ad-Hoc Networks, and Signals and Systems