Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 5 Issue 03, March-2016

Elimination of Data Redundancy before Persisting into DBMS using SVM Classification

M. Nalini[1]
Research Scholar,
Department of Computer Science and Engineering
St.Peter's University,
Avadi, Chennai, India

S. Anbu[2]
Professor,
Department of Computer Science and Engineering
St.Peter's College of Engineering and Technology
Avadi, Chennai, India

***Abstract*—Data Base Management System is one of the growing fields in computing world. Grid computing, internet sharing, distributed computing, parallel processing and cloud are the areas store their huge amount of data in a DBMS to maintain the structure of the data. Memory management is one of the major portions in DBMS due to edit, delete, recover and commit operations used on the records. To improve the memory utilization efficiently, the redundant data should be eliminated accurately. In this paper, the redundant data is fetched by the Quick Search Bad Character (QSBC) function and intimate to the DB admin to remove the redundancy. QSBC function compares the entire data with patterns taken from index table created for all the data persisted in the DBMS to easy comparison of redundant (duplicate) data in the database. This experiment in examined in SQL server software on a university student database and performance is evaluated in terms of time and accuracy. The database is having 15000 students data involved in various activities.**

Keywords*—*Data redundancy, Data Base Management System, Support Vector Machine, Data Duplicate.

I. INTRODUCTION

The growing (prenominal) mass of information present in digital media has become a resistive problem for data administrators. Usually, shaped on data congregate from distinct origin, data repositories such as those used by digital libraries and e-commerce agent based records with disparate schemata and structures. Also problems regarding to low response time, availability, security and quality assurance become more troublesome to manage as the amount of data grow larger. It is practicable to specimen that the peculiarity of the data that an association uses in its systems is relative to its efficiency for offering beneficial services to their users. In this environment, the determination of maintenance repositories with "dirty" data (i.e., with replicas, identification errors, equal patterns, etc.) goes greatly beyond technical discussion such as the everywhere quickness or accomplishment of data administration systems. The solutions available for addressing this situation need more than technical efforts; they need administration and cultural changes as well.

To distinguishing and manipulation replicas is essential to assure the peculiarity of the information made present by emphasizing system such as digital libraries and e-commerce agent. These systems may rely on compatible data to propose exalted profession benefit, and may be inclined by the existence of replica in their repositories. A Hereditary Scheme (HS) approach was used to register deduplication [1]. The problem of find out and destroy replica entries in a repository is known as record deduplication [2]. This approach bind several dissimilar portion of attestation extracted from the data content to exhibit a deduplication function that is capable to recognize whether two or more entries in a database are replicas or not. Since record deduplication is a time consuming work even for small databases, our scope is to encourage a process that predicting a peculiar combination of the best pieces of evidence, thus yielding a deduplication function that improves the performance using a method to compare the corresponding data for training process. Finally, this function can be applied on the entire data or even applied to other databases with similar characteristics. Moreover, modern supplemental data can be entreated similarly by the suggested function, as long as there is no unexpected deviate in the data patterns, something that is very important in huge databases. It is valuable consideration that these (arithmetical) services, which can be considered as a combination of several powerful deduplication regulations, is easy, fast and strong to calculate, permit its effectual technique to the deduplication of huge databases. By record deduplication using HS approach that generates gene excellence for each record using genetic operation. If that gene value equal with any other record that record was considered as a duplicate record. These trading operations are to increase the characteristic of given record. Genetic Operations are Reproduction, Mutation and Crossover [1]. From this, it can be understand that genetic operations can impact the performance of the record deduplication task. From the experimental results it can be concluded that the significant difference among the various efforts required obtaining suitable solution. The main contribution of the existing approach is to eliminate the record duplication. The experimental results obtained from the existing approaches PSO and GA are compared to evaluate the performance, where PSO is better than GA is proved [3]. Some of the research methods such as anomaly detection methods are used in various applications like online banking, credit card fraud, insurance and health care areas. The quality of the software product is depends on the data models used for managing dynamic changes on the data [4]. Some of the research works do testing process also integrating with the anomaly detection [5]. Few research works are removing the duplicate record removal in file system either at sub-directory or in the whole directory [6-

9]. Some of the existing approaches mainly divided into two types of categories like de-duplication, one is file-level [10-12] and the other one is Block-level de-duplication [13-15]. Means the duplication records are analyzed in terms of internal information and external information of the file system.

From the above background study, it is essential to develop a method for deleting data duplication to increase the quality of the data.

II. PROBLEM STATEMENT

Data and DBMS based application are growing speedily in most of the computing fields today. All the earlier approaches discussed in the earlier research works are concentrating on data cleaning, data arranging, error correction and duplication removal through anomaly detection. But it is essential to understand and apply data redundancy removal in advanced data mining and big-data analysis applications to provide customer satisfaction and quality service provision. Because, nowadays data centers and DBMA are interconnected into cloud environment which are integrated with parallel as well as distributed computing. Hence, it is necessary to provide speed access, memory management and accurate peculiar data retrieval. This problem is taken into account and in this paper by using a pattern recognition approaches data redundancy will be eliminated and improve the quality of service.

The contribution of the proposed approach in this paper is:

1. Read the data and create custom-index for each data, where the custom-index is considered as the pattern for comparison.
2. Each interval of time, during insertion of a new record, deletion of a record and altering the data the custom-index will be examined and eliminated.

III. PROPOSED SYSTEM MODEL

The entire functionality of this paper is illustrated in the following Figure-1. It is assumed that the data is maintained in the form of records, stored in a DBMS (as a Table). Each data has more number of fields $\boldsymbol{F = \{F_1, F_2, \ldots, F_n\}}$, all the fields are unique in terms of attributes. All the attributes are configured while structuring the table. A new table is created for each table within single column as the custom-index.

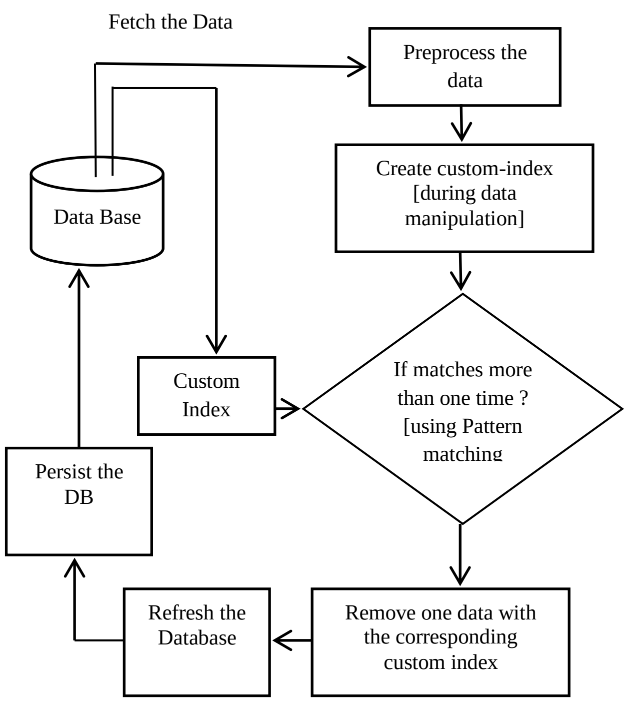

Fetch the Data
Preprocess the data
Create custom-index [during data manipulation]
Data Base
Custom Index
If matches more than one time ? [using Pattern matching
Persist the DB
Refresh the Database
Remove one data with the corresponding custom index

Figure-1. Proposed Model for Data

The custom index is created as a pattern. An exact pattern matching is to find all the occurrences of a particular pattern ($P = P_1 P_2 P_3 \ldots P_m$) where the pattern $\boldsymbol{P}$ contains $\boldsymbol{m}$ number of characters. In the earlier research works considered that the pattern $\boldsymbol{P}$ occur in the data $Y=\{Y_1, Y_2, \ldots, Y_n\}$.

IV. PREPROCESSING

In this paper Quick Search Bad Character (QSBC) function is used to verify all the characters in the data set. It read the data row by row and column by column and examine from left to right. If QSBC finds a same custom-index value more than one time, then it will intimate to the DB administrator to eliminate the duplicate record. At the same time it creates a log file to verify the performance of the QSBC function. QSBC is selected as a perfect solution for comparing the patterns due to the following reasons are:

(i). QSBC accurately matches the custom index speedily, and it is independent to other methods related to the DBMS.

(ii). QBSC examines and compare the patterns character by character using more shift operations.

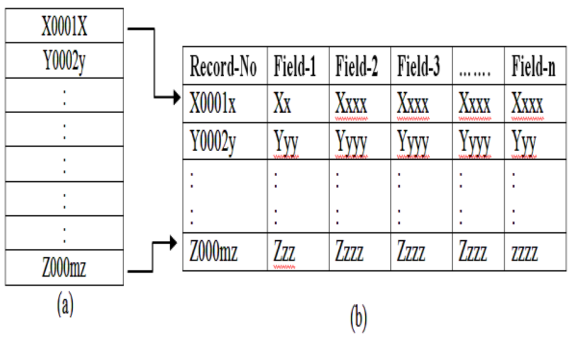
(iii). QBSC compares the pattern from left to right by creating sub patters in the entire data string.

This preprocessing phase helps to obtain the number of duplicate patterns available in the database. The above three stages of the pattern searching algorithm is given below:

```
1.  Void QSBC(Char *X, int m, int QSBC[])
2.  {
        a.  int i;
        b.  for(i=0; i<Size(DATA); i++)
        c.  {
        d.  if(lastCha == y[j + m-1] && first Cha
            == y[j])
                i.   {
                ii.  if(i <= 0)
                iii. {
                         1.  output(j);
                iv.  }
        e.  }
        f.  }
3.  }
```

Three steps are applied in the preprocessing stage. In step-1, character-character comparison is applied within the entire data and pattern. To find out the resemblance among the pattern and the data, the last character of the patters is compared with the entire data. If it matches with any patter, the first character of the pattern and the corresponding character in the entire data are compared. If these characters match, the algorithm enters into the next stage; otherwise, it goes to the last step. Whenever the algorithm finds a matching, it maintains a count and log information for further verification and intimation to the admin. The value of Size(DATA) is dependent on the alphabet size. For efficiency considerations, we have chosen a value accordingly to cover all the ASCII values of the characters defined in the present alphabet set. Void OUTPUT (int) is a function used to print the position (j) of the current window on the text. The following Figure-2 shows the custom-index and the data persisted in the DBMS.

The custom index is created by combining the first and the second field from the main table. The first and the last character are taken from the first and the last character of the Field-2. The middle characters in the custom index are a k-digit number generated automatically in increasing order [like auto number]. This number indicates the record number to search speedily from the data base separately as well as integrated with the custom index.



X0001X
Y0002y
:
:
:
:
:
Z000mz

(a)

Record-No	Field-1	Field-2	Field-3	…….	Field-n
X0001x	Xx	Xxxx	Xxxx	Xxxx	Xxxx
Y0002y	Yyy	Yyyy	Yyyy	Yyyy	Yyy
:	:	:	:	:	:
:	:	:	:	:	:
Z000mz	Zzz	Zzzz	Zzzz	Zzzz	zzzz

(b)

Figure- 2 . (a). Custom Index Table, (b). Main table

The entire proposed approach is implemented in SQL-SERVER 2008 software for persisting the data. The front end is designed in DOTNET software to provide user interaction. The proposed QSBC and pattern matching operations are coded in C# language and the results are obtained in terms of data size, number of duplicate records created and the time taken to detect and delete duplication.

V. RESULTS AND DISCUSSION

To evaluate the performance the experiment is examined more number of times with various numbers of records in each round. The number of records in each round is 1000, 2000, 3000, 4000 and 5000. The efficiency is verified by calculating the number of data redundancy eliminated in the database. Also the time taken to eliminate the redundant data is calculated to find out the efficiency of the proposed approach.

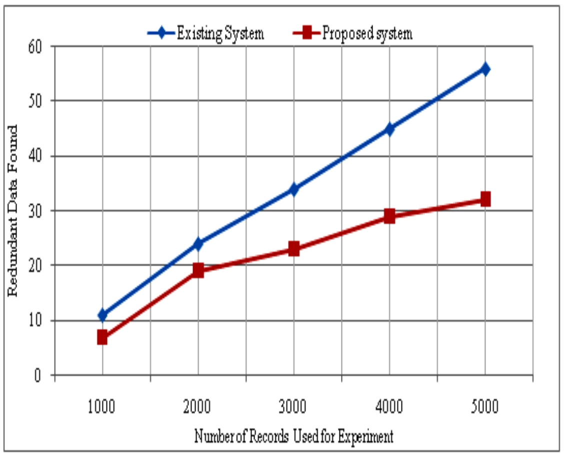


Figure-3 . Performance Evaluation of Finding Redundant Data

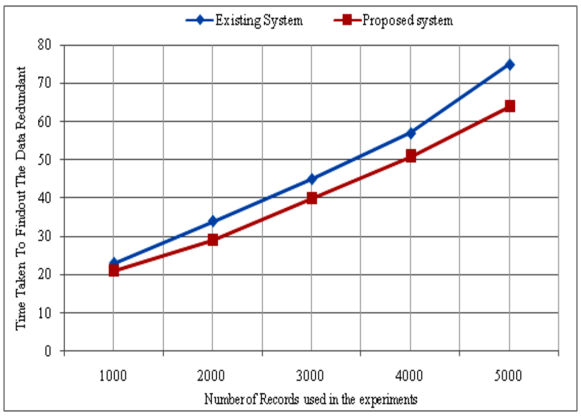


Figure-4 . Performance Evaluation By calculating the Time taken to eliminate the Redundant Data

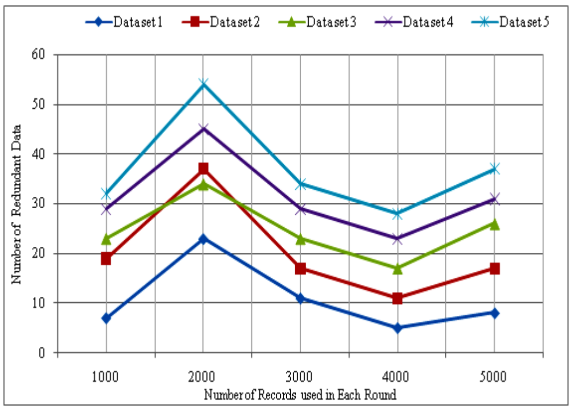


Figure-5 . Performance Evaluation in terms of Various Data Sets

Comparing with the existing approach, the performance of the proposed approach is better in term of time and elimination of redundant data in the DBMS. This results comparison is shown in the following Figure-3 and in Figure-4. Figure-5 shows the performance of the proposed approach in terms of redundant data detection in different dataset. The number of data redundancy detection is depend on the dataset. Also according to the time taken and elimination ratio, our proposed approach is better than the existing approach and it is shown in the following Figures-3 to Figure-5.

VI. CONCLUSION

To fulfill the objective of this paper, a custom index is maintained for all the data going to be persisted in the DBMS. The proposed QSBC approach can do detection and elimination of redundant data in any kind of DBMS speedily and accurately. Some of the anomaly functions are also carried out by the proposed approach is data cleaning and alignment, by comparing the custom-index. It is verified before and after data manipulation processes. From the obtained results and the graphs shown in Figure-3 to Figure-5, it is concluded that QSBC approach is a suitable, better approach for DBMS.

VII. REFERENCES

[1] Moises G. de Carvalho, Alberto H. F.Laender, Marcos Andre Goncalves, Altigran S. da Silva, “A Genetic Programming Approach to Record Deduplication”, IEEE Transaction on Knowledge and Data Engineering,pp 399-412, 2011.

[2] N. Koudas, S. Sarawagi, and D. Srivastava, “Record linkage: similarity measures and algorithms,” in Proceedings of the2006 ACM SIGMOD International Conference on Management of Data, pp. 802–803, 2006.

[3] Ye Qingwei, WuDongxing, Zhou Yu, Wang Xiaodong, " The duplicated of partial content detection based on PSO ", IEEE FifthInternational Conference on Bio-Inspired Computing: Theories and Applications, pp: 350 - 353, 2010.

[4] E.J. Weyuker and F.I. Vokolos, “Experience with Performance Testing of Software Systems: Issues,” IEEE Trans. Software Eng., vol. 26, no. 12, pp. 1147-1156, Dec. 2000, doi: 10.1109/32.888628.

[5] G. Denaro, A. Polini, and W. Emmerich, “Early Performance Testing of Distributed Software Applications,” Proc. Fourth Int‟l Workshop Software and Performance (WOSP ‟04), pp. 94-103, 2004, doi: 10.1145/974044.974059.

[6] Dutch, T.M. and W.J. Bolosky, 2011, “A study of practical de-duplication”, ACM Trans. Storage- DOI: 10.1145/2078861.2078864.

[7] Dubnicki, C., L. Gryz, L. Heldt, M. Kaczmarczyk and W. Kilianet al., 2009. Hydrastor, ”A scalable secondary storage”, Proccedings of the 7th Conference on File and Storage Technologies, (FST „09), pp: 197-210.

[8] Ungureanu, C., B. Atkin, A. Aranya, S. Gokhale and S.Ragoet al., 2010. Hydra FS, “A high-throughput file system for the HYDRAstor content-addressable storage system”, Proceedings of the 8th USENIX Conference on File and Storage Technologies, (FST‟ 10), USENIX Association Berkeley, CA, USA., pp: 17-17.

[9] Bolosky, W.J., S. Corbin, D. Goebel and J.R. Douceur, 2000, “Single instance storage in Windows® 2000”, Proceedings of the 4th Conference on USENIX Windows Systems Symposium, (WSS ‟00), USENIX Association Berkeley, CA, USA, pp: 2-2.

[10] Harnik, D., B. Pinkas and A. Shulman-Peleg, 2010, “Side channels in cloud services: De-duplication in cloud storage”, IEEE Security Privacy, 8: 40-47. DOI: 10.1109/MSP.2010.187.

[11] Gunawi, H.S., N. Agrawal, A.C. Arpaci-Dusseau, R.H. Arpaci-Dusseau and J. Schindler, 2005, “Deconstructing commodity storage clusters”, Proceedings of the 32nd Annual International Symposium on Computer Architecture, Jun. 4-8, IEEE Xplore Press, pp: 60-71. DOI:10.1109/ISCA.2005.20.

[12] Douceur, J.R., A. Adya, W.J. Bolosky, D. Simon and M. Theimer, 2002,”Reclaiming space from duplicate files in a server less distributed file system,” Proceedings of the 22nd International Conference on Distributed Computing Systems, (ICDCS‟ 02), ACM, USA., pp: 617-617.

[13] Quinlan, S. and S. Dorward, 2002, “Venti: A new approach to archival storage”, Bell Labs, Lucent Technologies.

[14] Muthitacharoen, A., B. Chen and D. Mazieres, 2001,”A low-bandwidth network file system”, Proceedings of the 18th ACM Symposium on Operating Systems Principles, Oct. 21-24, ACM Press, Banff, Canada, pp: 174-187. DOI: 10.1145/502034.502052.

[15] Vrable, M., S. Savage and G.M. Voelker, 2009, “Cumulus: File system backup to the cloud”, ACM Trans. Storage. DOI: 10.1145/1629080.1629084.