Journal of Computer Science 9 (11): 1514-1518, 2013
ISSN: 1549-3636
© 2013 Science Publications
doi:10.3844/jcssp.2013.1514.1518 Published Online 9 (11) 2013 (http://www.thescipub.com/jcs.toc)

CLUSTER BASED DUPLICATE DETECTION

A. Venkatesh Kumar and S. Vengataasalam

Department of Mathematics, Kongu Engineering College, Erode, Tamilnadu, India

Received 2013-09-09, Revised 2013-09-16; Accepted 2013-09-28

ABSTRACT

We propose a clustering technique for entropy based text dis-similarity calculation of de-duplication system. Improve the quality of grouping; in this study we propose a Multi-Level Group Detection (MLGD) algorithm which produces a most accurate group with most closely related object using Alternative Decision Tree (ADT) technique. Our propose a two new algorithm; first one is Multi-Level Group Detection (MLGD) formation using Alternative Decision Tree (AD Tree), which will split the bunch of record into self-sized cluster to reduce the volume of data for text comparisons. Second one is calculating the dis-similarity percentage using entropy and Information Gain (IG). We show experimentally our proposed technique achieves higher average accuracy than existing traditional de-duplication system. Further, our technique not required any manual tuning for clustering formations as well as dis-similarity calculation for any kind of business data. In this study, we have presented a new efficient method is introduced for clustering formation using ADTree algorithm for duplicate deduction. The new method offers more accuracy dis-similarity measure for each cluster data without manual intervention at the time of duplicate deduction.

Keywords: Clustering Algorithm, Alternative Decision Tree Algorithm, Duplicate Detection, Efficient Method, Manual Intervention, Cluster Data, Similarity Measure, Clustering Formation

1. INTRODUCTION

Improve a quality of data in data warehousing data cleansing is a very important task. Data cleansing deals with detecting and removing errors and inconsistent data. In data warehousing data from multiple source, which mean more than one data store location like different types of data base, flat file. Here we need transformation logic for converting source data into target database for standardize the record format as well as record value for detect a duplicate record. Therefore, data should be transformed and cleansed before loading into a target database. This process is usually called as Extract Transformation Loading (ETL) process.

Furthermore, data warehousing are used for decision making for real world business problem. So that correctness of data should be more important. Suppose, if anything wrong in the decision making or correctness of data then particular algorithm will produce a wrong result. However most of the existing cluster technique is designed for particular business problem.

In this study we propose a two new algorithm, First one is a clustering algorithm, which will overcome the existing clustering dis-advantage partition and hierarchical that may be either partition or hierarchical (Marrakchi *et al*., 2005). Second one is de-duplication algorithm, which will produce the dis-similarity, percentage of the pair of string in each cluster.

Here we introduced an efficient clustering mechanism as Multi-Level Group Detection using AD Tree for splitting a data into cluster, with most closely related object. Then we are applying the de-duplication mechanism in each clustered data, though this proposal method we can reduce the total time consumption for clustering formation and data comparison for de-duplication than existing traditional clustering mechanism and de-duplication mechanism.

Corresponding Author: A. Venkatesh Kumar, Department of Mathematics, Kongu Engineering College, Erode, Tamilnadu, India

2. MATERIALS AND METHODS

This problem taken up is to improve the performance of detecting duplicate record. The performance of entropy based duplicate detection could be enhanced through various means. It could be in the form of predictive accuracy, comprehensibility, speed and scalability. This research concentrates on the performance enhancement of duplicate detection through Multi-Level clustering technique and implemented it by using decision tree framework. The goal of this work is to identify groups of similar entities in the presents of linked environment and searching methods should reduce the number unwanted comparison during de-duplication. In order to achieve this goal, in this study we propose a new technique, First one is a clustering algorithm, which will overcome the existing clustering disadvantage, that may be either partition or hierarchical. Second one is de-duplication algorithm, which will produce the dis-similarity percentage of the pair of string in each clustered group.

2.1. Importance of the Work

Duplicate detection, which is an important subtask of data cleaning, Data integration and data quality are the two key components of a successful data warehouse as both completeness and accuracy of information are of paramount importance. Once this data is collected it can be made available both for direct analysis and for distribution to other, smaller data warehouses.

From a conceptual perspective, data warehouses store snapshots and aggregations of data collected from a variety of source systems. Data warehouses encompass a variety of subject areas. Each of these source systems could store the same data in different formats, with different editing rules and different value lists. For example, gender code could be represented in three separate systems as male/female, 0/1 and M/F respectively; dates might be stored in a year/month/day, month/day/year, or day/month/year format. In the United States “03062010” could represent March 6,2010 while in the United Kingdom it might represent June 3, 2010.

Data warehouses involve a long-term effort and they are usually built in an incremental fashion. In addition to adding new subject areas, at each iteration, the breadth of data content of existing subject areas is usually increased as users expand their analysis and their underlying data requirements.

2.2. Duplicate Detection

2.2.1. MLGD Formation Using ADTree

MLGD forms a tree for the clustering process (Perla and Belliveau, 2005). In the tree structure, the height of each level of nodes represents the dis-similar degree between each cluster. MLGD incorporate the futures of ADTree features and overcome the existing hierarchical clustering problem and reduce the time consumption for duplicate detection (Mirzaei and Rahmati, 2008) and number of record comparisons. Here we did not use any split algorithm for splitting data into a cluster; instead we are using ADTree technique for splitting a whole data into cluster. A condition predicates the attribute comparison value, here we are checking clustering index value contains the short_name value or not. ADTree divide the data based on short name; if cluster is already available with the short name then insert a record into a same cluster else create a new cluster with the new name of short name then insert into a new cluster. In each cluster sub-set short name pointing to the whole record. If cluster is already available then starts the de-duplication process else create a new cluster and then exit from the process.

2.3. AD Tree Implementation Algorithm

```
Initialize:  Parent_List[n] ←0,
             Child_List[n] ← 0,
             Grant_child_list[n] ←0;
Loop L1: while !endOfRecord[Record]
      C1 ← Level_1_cluster_attr_value;
     Position ← Size[Parent_List]+1;
If(is_valid[ C1])  then
C2 ← Level_2_cluster_attr_value;
C3 ← Level_3_cluster_attr_value;
Child_Position ← Size[Child_List]+1;
Loop L2:while !endOfRecord[Record]
If(!contains[parent_list, C2]) then
Parent_List[Position]← Create new Cluster
      C1, Parent_Position;
     Child_List[n] ← Create new Cluster C2,
      Child_Position;
Grant_child_list[n]←Create new Cluster C3,
Child_Position;
Parent_List[Position]← Insert into new Cluster
{Parent_Position, C1,vector[Record_Inform ation])};
Child_List[n] ← Insert into new Cluster
{Child_Position, (C2,vector[Record_Information])};
Grant_child_list[n]← Insert into new Cluster
{Child_Position, (C2, vector[Record_Information])};
       Return new_cluster;
```

```
else
Parent_List[Position] ← Insert into existing Cluster
{Parent_Position, (C1,vector[Record_Information])};
Child_List[n] ← Insertinto existing Cluster
{Child_Position, (C2,vector[Record_Information])};
Grant_child_list[n] ←Insert into existing Cluster
        Child_Position,(C2, ector[Record_Information])};
 existing_Cluster ← call Dis-Similarity Calculation
 Algorithm {C1, vector [Record_Information]}
         Return existing_Cluster;
       endif
Goto L2
else
      Return 0;
      endif
Goto L1
```

2.4. Dis Similarity Calculation

The cluster formation method (Srinivas and Mohan, 2010) mainly focus on form a similarity value in single group, for this purpose we are using different method and result of each method is different cluster based on data and spread condition. Here, we outline our main algorithm and give optimizations that we use in the experiments. For ease of presentation, we shall first explain a MLGD using ADTree clustering algorithm (Zeng *et al*., 2009), called MLGD, that forms the technical core of our approach.

And shown how to use the clustering algorithm to calculate dis-similarity value for de-duplication. First we are constructing a truthtable for each pair of string in the each cluster. Every Boolean function can be specified as a truthtable with the value of 0,1 and function has a "n" argument, then the total possible argument combinations are 2^n. To construct the logical representation of two different string tokens truthtable. In pair of strings which one have more length Where Ci, i = 1…..n represents the i'th character of column. Rj, j = 1…..m represents the j'th character of row.

Then apply the truthtable value into entropy and gain formula. The output of gain will be a dis-similarity percentage. The gain values are compared with existing cluster gain value if it is greater than existing and length of current string is greater than existing cluster string then set current as "PRIMARY" else set the current as "SECONDARY" and its dis-similarity score.

2.5. Truthtable Construction Algorithm

```
Initialize: Row ← 0; Column ← 0
Loop L1: While !empty(String_1)
C1 ← String_1 [Row]; Row = Row + 1;
Loop L2: While !empty(String_2)
C2 ← String_2 [Column]; Column = Column + 1
If C1 = C2 Then
Truth [Row, Column] = 1
Else Truth [Row, Column]=0
Endif Goto L2
Goto L1
```

2.6. Dis-Similarity Calculation Algorithm

```
Input:
C1 ← {Short_Name}; C2 ← {Actual_name}
VEC ← {Vector [Subject_Information]}
Process: Loop L1: While !endOfVector[Record]
Begin
        C3 ← VEC [C1].getActualName ();
         p,n ← Call TruthTable Construction Algorithm
        (C1,C2);
        Entropy ($p_i$,$n_i$) ← -p $\log_2$(p)-n $\log_2$(n)
        Gain$_i$ ← ∑ (Entropy value of child dataset)-∑
        (Entropy value of total dataset) * 100
        Loop L2: While !endOfVector[C1]
        Begin
        C4 ← VEC [C1].getDis_Sim_Score();
        If(Gain$_i$>C4 and Length(C3)> Length(C4))
        Then
               VEC ← Insert into Existing Cluster C2
                        And set it as "Primary"
                 VEC ← Update Existing Cluster C3
                          And set it as "Secondary"
        Else
                 VEC ← Insert into Existing Cluster C2
                 And set it as "Secondary"
        Endif
    Goto L2
Goto L1
Return VEC
```

2.7. Result Comparison

Table 1 and Fig. 1 show duplicate detection without grouping exponentially increase the number of iteration. Due to this large size of data, volume approach is not fit and also there is a chance to face an out of memory issue and its control is out of our hand. But duplicate detection with grouping gradually increases the number of iteration even, if we increase the data size. Throught grouping method may be able to control the out of memory issue. At the same time we can handle only one group for duplicate detection, as this control is in our hand.

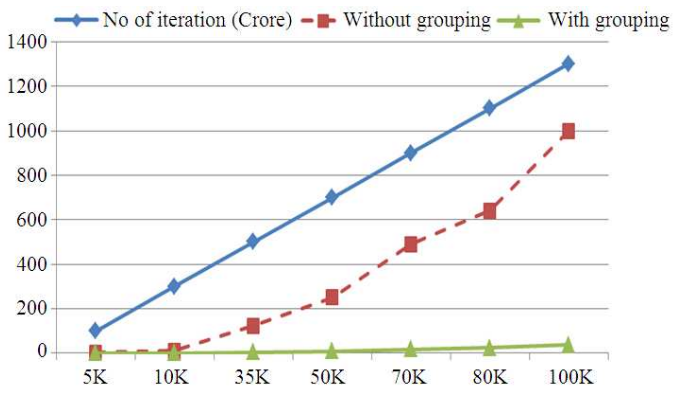

Fig. 1. Number of Iteration between with grouping and without grouping

Table 1. Total number of Iteration between with grouping and without grouping

Record volume	No of iteration (Croce)	Without grouping	With grouping
5 K	100	2.5	0.16
10 K	300	10.0	0.37
35 K	500	122.5	4.56
50 K	700	250.0	9.30
70 K	900	490.0	18.23
80 K	1100	640.0	23.80
100 K	1300	1000.0	37.19

None of the existing algorithm has this kind of fuctionality to find a duplicate value wihin that specified time line.

3. RESULTS AND DISCUSSION

We show experimentally our proposed technique achieves higher average accuracy than existing traditional de-duplication system. Further, our technique not required any manual tuning for clustering formations as well as dis-similarity calculation for any kind of business data.

ADTree_groupInductionAlgorithm () forms a tree for the clustering process. In the tree structure, the height of each level of nodes represents the dis-similar degree between each cluster. MLC incorporates the futures of ADTree features and it overcomes the existing hierarchical clustering problem and it reduces the time consumption for duplicate detection and number of record comparisons. Here we do not use any split algorithm for splitting data into a cluster; instead we are using ADTree technique for splitting a whole data into cluster. ADTree divides the data based on short name; if cluster is already available with the short name then insert a record into a same cluster else create a new cluster with the new name of short name and then insert into a new cluster. In each clusters sub-set short name pointing to the whole records If cluster is already available then starts the de-duplication process else create a new cluster and then exit from the process.

3.1. Limitation

At present algorithm will not support for detect the duplicate of two diffent image and two different video file. Even though normal word is saved as a image file or video file, current algorithm wont support.

3.2. Future Work

The present study can be extended in the following direction: The record hiding concept may be adopted to hide sensitive data to maintain the privacy of data. Future work will involve looking into ways to improve the scalability and to combine different de-duplication approaches into a cloud computing system. The algorithm may be extended to handle missing values, image value, video value in a natural way during grouping. To solve uncertainty grouping problem, the association rule mining may be integrated into grouping algorithm. This algorithm may be extended to grouping and to identify duplicate in web document, video text and image text.

4. CONCLUSION

In this study, we have presented a new efficient method is introduced for clustering formation using ADTree algorithm for duplicate deduction. The new method offers more accuracy dis-similarity measure for each cluster data without manual intervention at the time of duplicate deduction. Compare to existing clustering algorithm either partition or hierarchical, our new method is more robust and easy to reach the solution of real world complex business problem. If we apply the propose de-duplication algorithm with this new method, surely it will reduce the total time consumption as well as avoid the unwanted record comparison.

5. REFERENCES

Marrakchi, Z., H. Mrabet and H. Mehrez, 2005. Hierarchical FPGA clustering based on multilevel partitioning approach to improve routability and reduce power dissipation. Proceedings of the International Conference on Reconfigurable Computing and FPGAs, Sept. 28-30, IEEE Xplore Press, Puebla City, pp: 25-28. DOI: 10.1109/RECONFIG.2005.23

Mirzaei, A. and M. Rahmati, 2008. Combining Hierarchical clusterings using Min-transitive closure. Proceedings of the 19th International Conference on Pattern Recognition, Dec. 8-11, EEE Xplore Press, Tampa, FL., pp: 1-4. DOI: 10.1109/ICPR.2008.4761275

Perla, R.J. and P.P. Belliveau, 2005. Antibiogram-derived radial decision trees: An innovative approach to susceptibility data display. Am. J. Infect. Dis., 1: 124-127. DOI: 10.3844/ajidsp.2005.124.127

Srinivas, M. and C.K. Mohan, 2010. Efficient clustering approach using incremental and hierarchical clustering methods. Proceedings of the International Joint Conference on Neural Networks (IJCNN), Jul. 18-23, EEE Xplore Press, Barcelona, pp: 1-7. DOI: 10.1109/IJCNN.2010.5596666

Zeng, J., L. Gong, Q. Wang and C. Wu, 2009. Hierarchical clustering for topic analysis based on variable feature selection. Proceedings of the 6th International Conference on Fuzzy Systems and Knowledge Discovery, Aug. 14-16, EEE Xplore Press, Tianjin, pp: 477-481. DOI: 10.1109/FSKD.2009.205