Detection of Duplicate records by using Progressive Windowing Technique

Abstract:

Duplicate detection is process of finding multiple representation of same real word entity. In very large data sets, like malls and website data, the data is difficult to manage and if there are duplicate entries it is all the more time consuming. Process is required to find duplicate entities in very short time by maintaining the quality of dataset. In this proposed approach a novel method is used namely progressive windowing that considerably increases the efficiency of finding the duplicates and maintain quality of dataset.

Keywords: *Duplicate detection, cleaning of data, windowing.*

I. Introduction:

Identifying multiple representations of same real word entities is duplication detection. The large databases usually face the challenges of duplicate data due to erroneous data entry. For example the data in super markets, retailing industry face these problems. The problem of detecting duplicate entities is an important data cleansing task, necessary to improve data quality. When merging data from different sources, the result is unique i.e. ensure that an entity has only one representation in result.

In progressive windowing technique sorting method is applied on input dataset by using a predefined sorting key and then compares records which are present within a window in the sorted order. The instinct is that records that are near in the sorted list are have more possibility to be duplicates than records that are far apart, because they are already similar according to their sorting key.

This static approach has already been defined as the sorted list of record pairs (SLRPs) hint [1]. The Progressive windowing algorithm differs by dynamically varying the execution order of the comparisons based on window size and intermediate results. Furthermore, progressive windowing method integrates a progressive sorting phase and can progressively process significantly larger datasets.

II. Literature review:

This system Near-uniform Range Partition Approach for Increased Partitioning in Large Database by Jie Song et al. allows Database partitioning technique which adopts divide and conquer method can efficiently simplify the complexity of managing massive data and improve the performance of the system. According to the “divide and conquer” method, the table is partitioned into several parts.

The system Top-k Set Similarity JoinsbyChuan Xiao et al. proposed the problem of answering similarity join queries to retrieve top-k pairs of records ranked by their similarities. Traditional approaches for the similarity joins with a given threshold will have to make guesses on the similarity threshold and incur much redundant calculation. In this an efficient algorithm that computes the answers in a progressive manner, upper bound score and the Kth temporary result score are exploited to develop several optimizations and to improve the space and time efficiencies of the algorithm. This algorithm provides the top-K pair records ranked by their similarities by eliminating guess work of users.

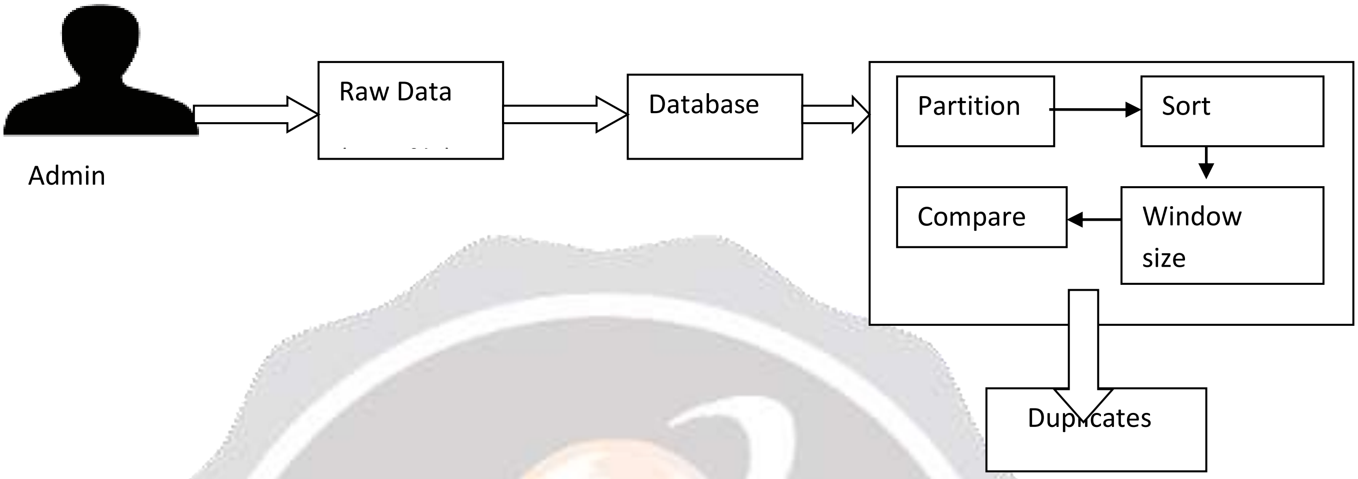
Framework for Evaluating Clustering Algorithms in Duplicate Detection by OktieHassanzaeh et al. proposed approach for duplicate detection using clustering framework. In this approach entity resolutions used as a part of the data cleaning process to identify records that possibly refer to the same real-world entity. It provides an evaluation framework for understanding what hurdles remain towards the goal of truly scalable and general purpose duplication detection algorithms. It generate results using partitioning of the similarity graph which is the common approach in many early duplicate detection techniques, confirms the common wisdom that this scalable approach results in poor quality of duplicate groups and it also shows that this quality is poor even when compared to other clustering algorithms that are as efficient. However this approach will work only in sequential manner, due to this time required for duplicate detection is large as compared to other state-of-art approaches and results are also not satisfactory.

III. Progressive Windowing Technique:

The algorithm takes total five input parameters as follows
D is a reference to the data, which has not been loaded from disk yet.
K sorting key defines the attribute that should be used in the sorting step.
W specifies the window size which is used to decide window size for record comparision,
Parameter I defines the enlargement interval for the progressive iterations.
For now, assume it has the default value 1.
The last parameter N defines the number of records in the dataset.

```
Step 1: procedure PSNM(D, K, W, I, N)
Step 2: pSize->calcPartitionSize (D)
Step 3: pNum->[N/pSize-W + 1)]
        Where,  array order size N as Integer
Step 5: array recs size pSize as Record
Step 6: order ->sort Progressive (D, K, I, pSize, pNum)
Step 7: for currentI-> 2 todW=Iedo
Step 8: for current->1 to pNum do
Step 9: recs□loadPartition (D, currentP)
Step 10: for dist belongs to range(currentI, I, W) do
Step 11: for i ->0 to |recs|_ dist do
Step 12: pair-> <recs[i], recs[i + dist]>
Step 13: if compare (pair) then
Step 14: emit (pair)
```

System Architecture:



IV. Conclusion:

.This algorithm increase the efficiency of duplicate detection for situations with limited execution time and high accuracy. In future work, want to combine these progressive approaches with scalable approaches for the duplicate detection in order to deliver the result even faster. The parallel sorted neighbourhood can be executed to find in parallel.

V. References:

[1] O. Hassanzadeh, F. Chiang, H. C. Lee, and R. J. Miller, “Framework for evaluating clustering algorithms in duplicate detection,” in Proceedings of the International Conference on Very Large Databases (VLDB), 2009.

[2] S. Yan, D. Lee, M. yen Kan, and C. L. Giles, “Adaptive sorted neighborhood methods for efficient record linkage,” in International Conference on Digital Libraries (ICDL), 2007.

[3] Jie Song, Yu-bin Bao “Near-uniform Range Partition Approach for Increased Partitioning in Large Database”,2010.

[4] U. Draisbach and F. Naumann, “A generalization of blocking and windowing algorithms for duplicate detection.” in International Conference on Data and Knowledge Engineering (ICDKE), 2011.

[5] H. B. Newcombe and J. M. Kennedy, “Record linkage: making maximum use of the discriminating power of identifying information,” Communications of the ACM, vol. 5, no. 11, 1962

[7] J. Madhavan, S. R. Jeffery, S. Cohen, X. Dong, D. Ko, C. Yu, and A. Halevy, "Web-scale data integration: You can only afford to pay as you go," in Proceedings of the Conference on Innovative Data Systems Research (CIDR), 2007.

[8] S. R. Jeffery, M. J. Franklin, and A. Y. Halevy, "Pay-as-you-go user feedback for dataspace systems," in Proceedings of the International Conference on Management of Data (SIGMOD), 2008.

[9]O. Hassanzadeh and R. J. Miller,"Creating probabilistic databases from duplicated data," VLDB Journal, vol. 18, no.5, 2009.

[10] F. Naumann and M. Herschel, An Introduction to Duplicate Detection. Morgan & Claypool, 2010.

[11] U. Draisbach, F. Naumann, S. Szott, and O. Wonneberg, "Adaptive windows for duplicate detection," in Proceedings of the International Conference on Data Engineering (ICDE), 2012.

[12] P. Christen, "A survey of indexing techniques for scalable record linkage and deduplication," IEEE Transactions on Knowledge and Data Engineering (TKDE), vol. 24, no. 9, 2012.