Volume 6, Issue 8, August 2016 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Different Duplicate Detection Methods

Priyanka D N*, Sophiya Mathews
Computer Science and Engineering, MG University,
Kerala, India

Abstract— In the real world the data set may have one or more representation of the same real world entities. Duplicate may arise due to transaction errors and due to incomplete data. Removing such duplicate in real life is a complex task. It is not straightforward to efficiently find and remove the duplicates from a large data set. This paper focus on comparison with traditional duplicate detection methods Incremental Sorted Neighborhood Method (ISNM) and the Duplicate Count Strategy (DCS++) method with Progressive Sorted Neighborhood Method(PSNM) method.

Keywords— Duplicate Detection, Dataset

I. INTRODUCTION

Databases are important part of a company such as most of its data in kept in it, making the data set up to date without duplicates and to keep its data cleansing the duplicate detection plays an important role. For keeping the quality of data within the company its time consuming and costly. Most of the existing system faces the problem of finding duplicates earlier in the detection process.

Entity resolution method [1] identifies the multiple representation of same identity. The progressive Sorted Neighborhood method [2] reduces the average time for which duplicate is found. But the existing methods Incremental Sorted Neighborhood Method[5] finds the duplicates by incremental comparison between the data in a given window and duplicate count strategy[3] , finds the duplicate by increasing the window size by the number of duplicates detected .

This report is organized as follows: Sect ion II presents brief description of Incremental Sorted Neighborhood for duplicate detection, Section III presents DSC ++ method to find the duplicate data , Section IV Section III presents Progressive Sorted Neighborhood Method to find the duplicate data, compare these method and finally conclusion is presented in Section V.

II. INCREMENTAL DUPLICATE DETECTION METHOD

This Method is the extension of the basic Sorted Neighborhood Method. In this method initially it sorts the given data set using selection sort based on a sorting key. Sorting is performed so that the similar tuples are close to each other .The sorting key is unique and is not virtual then defines the window size then compares the records within that window specified. Incremental SNM simply increments the size of window when duplicates are found. There is no windowing concept. It is much more efficient than the Sorted Neighborhood Method for ISNM technique:[5]

- Sort the data set
- Specify the initial window size
- Compare all records within the window
- If duplicate found is d then increment the window
- If no such duplicate find just slide the window size
- Duplicate Detected

Fig.1 shows how the specified window size slide across the given data set. There are lots of comparisons in this method in order to reduce the comparison and to find more duplicates within the specified window Duplicate Count Strategy ++ is used which increases the window size based on the number of duplicates detected.

III. DUPLICATE COUNT STRATEGY ++

This method is the extension of DCS Duplicate Count Strategy. It is a strategy which dynamically adapts the window size. That is it varies the window size based on the number of duplicates detected. Adaptation will sometimes increase or decrease the number of comparisons If more duplicates of a record are found within a window, the larger the window should be If no such duplicate of a record within its neighborhood is found, assume that there are no duplicates or the duplicates are very far away in the sorting order. Each tuple t_i is once at the beginning of a window w it is then Compare with w 1 successors If no duplicate for t_i is found, continue as normal else increase window.

DCS+ for finding original source is describes as follows: i

- Sorts the given data set
- specify the window w

- Compare the first record in the window with that of all of the records in the window which is shown in Fig.2
- Increase window size while duplicate detected / comparison $\geq \varphi$ where φ is a threshold
- Slide the window if no duplicates found within the window
- If duplicates found , for each detected duplicate the next w-1 records of that duplicate are added to the window.
- Duplicates are detected.

The Fig.3 above shows how to perform these duplicate detection methods

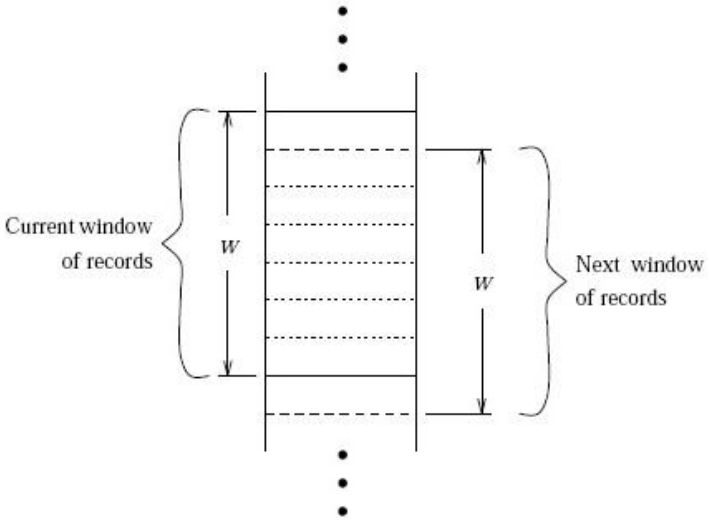

Fig. 1. Window Sliding Example

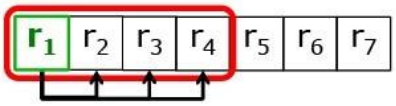

Fig. 2. Comparing records in DCS++

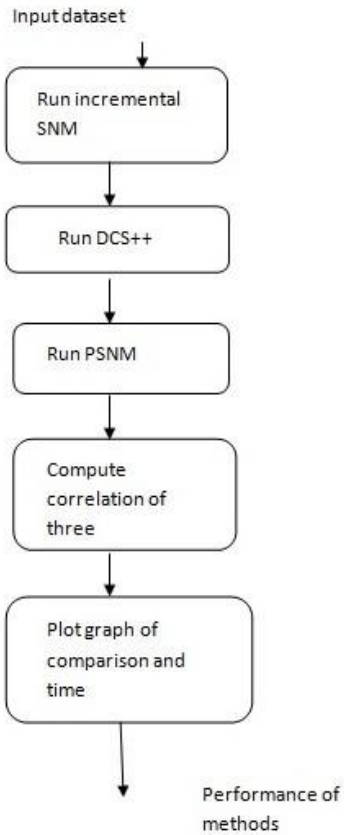

Fig. 3. Flow Chart for performing the detection

IV. PROGRESSIVE SORTED NEIGHBORHOOD METHOD

This method progressively finds the duplicate. Initially it sorts the input data and define a window size it partition the entire data set based on the partition size and compares the records within the window specified in each partition. In order to progressively find the duplicates the PSNM algorithm defines an enlargement interval . The enlargement interval varies from the smallest window size to the maximum that is w-1. Thus ensuring that the promising close neighbors are selected first and less promising records later. The PSNM algorithm increases the efficiency of finding duplicates by dynamically changing the window size. In the existing method there is a problem load the data set each time to compare but in this method it load the partition once and by changing the enlargement interval it progressively detect the duplicates.

V. CALCULATING SIMILARITY MEASURES

The analysis of these methods can be performed using the Cosine Similarity and TF-IDF Cosine similarity is a measure of similarity between two vectors here two tuples of an inner product space that measures the cosine of the angle between them. Term Frequency also known as TF measures the number of times a term (word) occurs in a document. TF(t) = (Number of times term t appears in a document) / (Total number of terms in the document).IDF Inverse Document Frequency IDF(t) = Mathematical Log(Total number of documents / Number of documents with term t in it).

VI. COMPARISON

Compared to the two techniques used in duplicate detection , ISNM and DCS++ method only find duplicate. it is not much efficient. Initially the duplicates are not found, only fewer comparisons and fewer number of the duplicate detected.

1) Duplicate Detection Method which is efficient and cost effective
2) It reports the duplicates earlier in the detection process
3) Windowing concept is used for finding the number of duplicates. So the PSNM is the relevant method to solve current detection problems.

VII. RESULT

a) : From Fig.4 it is clear that with limited number of comparison PSNM finds more duplicates than the ISNM and DCS++ DCS++ is more efficient than the ISNM bt less efficient than the PSNM. That is with limited number of comparison PSNM finds more duplicate thus it reduces the amount of time need to find the duplicates. Thus it reduces the average time after which a duplicate is found.

b) : Figure5 shows how the methods finds the duplicates based on the execution time. PSNM finds the duplicates initially in the detection phase .ISNM and DCS++ finds the duplicate only after a certain time the PSNM finds the duplicate. Thus it indicates that PSNM algorithm runs within a limited time. PSNM finds maximum number of duplicates than the rest of methods.

c) : Figure 4 shows the comparison between three duplicate detection Methods and table 1 shows the comparison table.

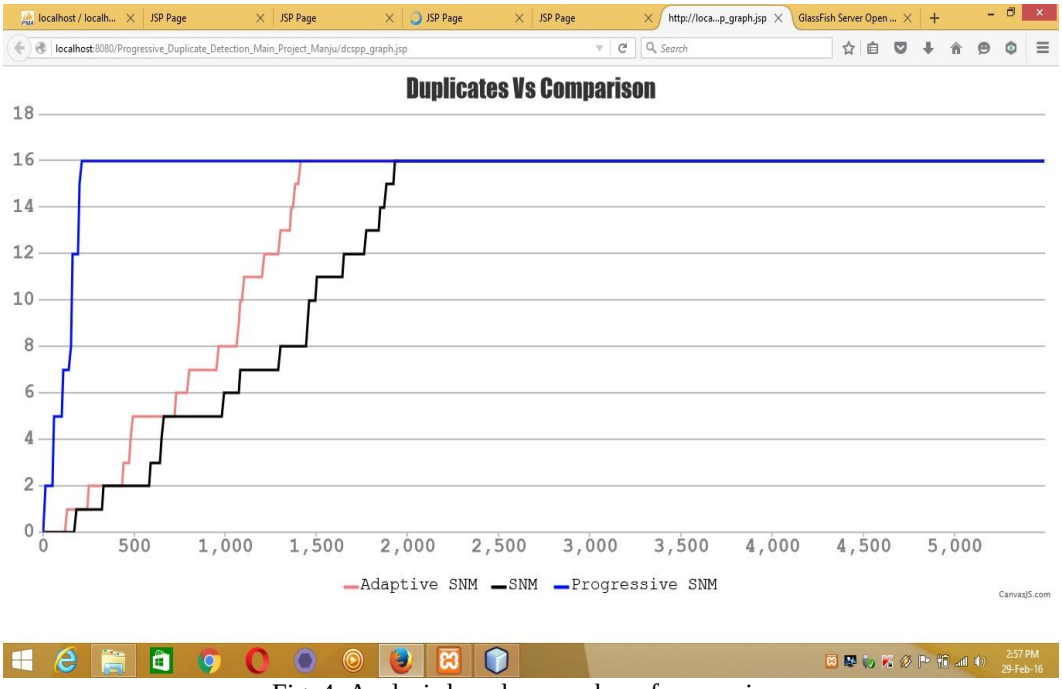

Fig. 4. Analysis based on number of comparison

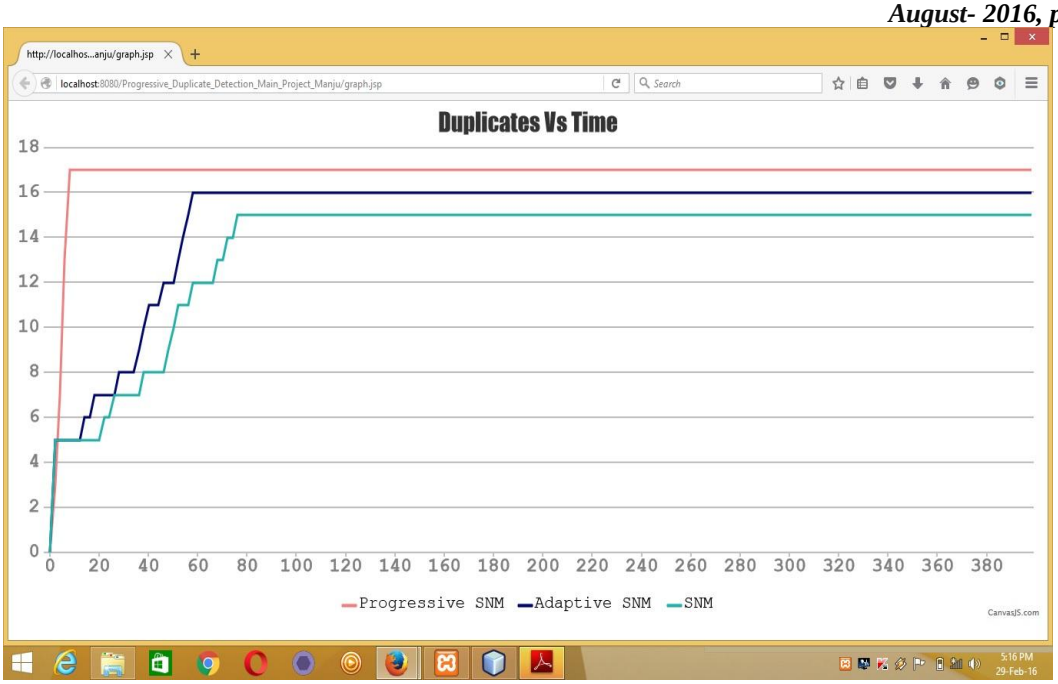

Fig. 5. Analysis based on time taken

Table 1: Comparison Table

S.NO:	METHOD	FUNCTION
1	ISNM	• No widowing Concept • Less efficient
2	DCS++	• Window size increases based on the number of the duplicates detected
3	PSNM	• Progressive Technique • More efficient

VIII. CONCLUSION AND FUTURE WORK

Progressive Duplicate Detection Method try to report the number of duplicate found in the data set. Thus this method tries t_i improve the average time between the duplicate detection. PSNM algorithm progressively run the algorithm by the simply selecting the user specified parameters such as window size, block size etc. when we try to implement this efficient algorithm as a web application it will take more time to find duplicates. Hadoop Map reduce can be used to improve the efficiency by giving the key and the count of duplicate detected.

REFERENCES

[1] Progressive Duplicate Detection Thorsten Papenbrock, Arvid Heise, and Felix Naumann in 2015

[2] S. E. Whang, D. Marmaros, and H. Garcia-Molina, Pay-as-you-go entity resolution, IEEE Trans. Knowl. Data Eng., vol. 25, no. 5, May 2012

[3] M. A. Hernandez and S. J. Stolfo, Real-world data is dirty: Data cleansing and the merge/purge problem, Data Mining Knowl. Discovery, vol. 2, no. 1, pp. 937, 1998

[4] U. Draisbach and F. Naumann, A generalization of blocking and windowing algorithms for duplicate detection, in Proc. Int. Conf. Data Knowl. Eng., 2011, pp. 1824.

[5] S. Yan, D. Lee, M.-Y. Kan, and L. C. Giles, Adaptive sorted neighborhood methods for efficient record linkage, in Proc. 7th ACM/ IEEE Joint Int. Conf. Digit. Libraries, 2007, pp. 185194.

[6] U. Draisbach, F. Naumann, S. Szott, and O. Wonneberg, Adaptive windows for duplicate detection, in Proc. IEEE 28th Int. Conf. Data Eng., 2012, pp. 10731083.

