iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 4 **Issue:** VIII **Month of publication:** August 2016

DOI:

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

Volume 4 Issue VIII, August 2016
ISSN: 2321-9653
International Journal for Research in Applied Science & Engineering Technology (IJRASET)

A Novel Approach for Eliminating Duplicates in Large Dataset

N. Chaithanya[1], Appini Narayanarao[2], M. Srinivasulu[3]
[1]M. Tech, [2]Associate Professor, [3]Assistant Professor
Shree Institute of Technical Education

Abstract ***One of the serious problems faced in several applications with personal details management, customer affiliation management, data mining, etc is duplicate detection. This survey deals with the various duplicate record detection techniques in both small and large datasets. To detect the duplicity with less time of execution and also without disturbing the dataset quality, methods like Progressive Blocking and Progressive Neighborhood are used. Progressive sorted neighborhood method also called as PSNM is used in this model for finding or detecting the duplicate in a parallel approach. Progressive Blocking algorithm works on large datasets where finding duplication requires immense time. These algorithms are used to enhance duplicate detection system. The efficiency can be doubled over the conventional duplicate detection method using this algorithm. Several different methods of data analysis are studied here with various approaches for duplicate detection.***
Keywords: Data Duplicity Detection, Progressive deduplication, PSNM, Data Mining

I. INTRODUCTION

Data are among the most important assets of a company. But due to data changes and sloppy data entry, errors such as duplicate entries might occur, making data cleansing and in particular duplicate detection indispensable. However, the pure size of today's datasets render duplicate detection processes expensive. Online retailers, for example, offer huge catalogs comprising a constantly growing set of items from many different suppliers. As independent persons change the product portfolio, duplicates arise. Although there is an obvious need for de duplication, online shops without downtime cannot afford traditional de duplication. Progressive duplicate detection identifies most duplicate pairs early in the detection process. Instead of reducing the overall time needed to finish the entire process, progressive approaches try to reduce the average time after which a duplicate is found. Early termination, in particular, then yields more complete results on a progressive algorithm than on any traditional approach.

II. RELATED WORKS

P. G. Ipeirotis et al. proposed the following concepts in [9] which states that the ER algorithm is used in this paper for focusing on determine the expected records that are alike first. This technique gives various hints like the other general techniques. But still many problems are yet to be solved. There are three different types of hints which match the several ER algorithms called sorted list of record pairs, hierarchy of record partitions and order list of records. The hints are used to maximize the count of similar records recognized with less work and to increase ER quality. S. E. Whang et al. [10] stated a survey on the active methods and non identical duplicate entries present in the records of the database records are all investigated in this paper.

It works for both the duplicate record detection approaches. 1) Distance Based technique that measures the distance among the individual fields, by using distance metrics of all the fields and later computing the distance among the records. 2) Rule based technique that uses rules for defining that if two records are same or different. Rule based technique is measured using distance based methods in which the distances are 0 or 1. The techniques for duplicate record detection are very essential to improve the extracted data quality.

U. Draisbach et al. in [11] denoted a Duplicate Count Strategy is used which become accustomed to the window size depending on the count of duplicates detected. There are three strategies:

A. Key similarity strategy: The associations of the sorting keys influence the window size which is improved when the sorting keys are alike. Then we can expect several related records in this model.

B. Record similarity strategy: The associations of the records influence the window size. The replacement of the real resemblance of the records is present inside the window.

C. Duplicate count strategy: The count of the known duplicates influence the window size. DCS++ algorithm proves to be

trustworthy than the SNM algorithm without losing the effectiveness. The algorithm of DCS++ is used to calculate the transitive closure and then save comparisons.

U.Draisbach and F.Naumann in [12] proposed two major methods called blocking and windowing used to reduce the comparisons are studied in this paper. Sorted Blocks that denotes a generalization of these two methods are also analyzed here. Blocking divides the records to disjoint subsets and windowing slides a window on the sorted records and then comparison is made between records within the window. The sorted Blocks have advantages like the variable size of partition size instead of the size of the window. A.Thor et al. [13] proposed a theory of deduplication which is also known as Entity Resolution which is used for determining entities associated to similar object of the real world. It is very important for data integration and data quality. Map Reduce is used for SN blocking execution. Both blocking methods and methods of parallel processing are used in the implementation of entity resolution of huge datasets. Map Reduce steps:-

1) Demonstrating how to apply map reduce for a common entity having blocking and matching policies.
2) Identifying the main challenges and proposing two JobSN and RepSN approaches for Sorted Neighborhood Blocking.
3) Evaluating the two approaches and displaying its efficiencies. The size of the window and data skew both influences the evaluation.

III. PROPOSED SYSTEM

The proposed solution uses two types of novel algorithms for progressive duplicate detection, which are as follows: PSNM – It is known as Progressive sorted neighborhood method and it is performed over clean and small datasets. PB – It is known as Progressive blocking and it is performed over dirty and large datasets. Both these algorithms improve the efficiencies over huge datasets. Progressive duplicate detection algorithm when compared with the conventional duplicate persuades two conditions which are as follows [1]:

A. Improved early quality: The target time when the results are necessary is denoted as t. Then the duplicate pairs are detected at t when compared to the associated conventional algorithm. The value of t is less when compared to the conventional algorithm's runtime.
B. Same eventual quality: When both the progressive detection algorithm and conventional algorithm finishes its execution on the same time, without terminating t earlier. Then the produced results are the same.

As demonstrated in Fig. 1 i.e. System Architecture, initially a database is picked for deduplication and for practical processing of data, the data is split into numerous partitions and blocks. Clustering and classification is used after sorting the data to make it more ordered for efficiency. Next step the pair wise matching is done to find duplicates in blocks and through new transformed dataset is generated. Finally the transformed data is updated in database after all filtrations. When the time slot of fixed is given then the progressive detection algorithms works on maximizing the efficiencies. Thus PSNM and PB algorithms are dynamically adjusted using their optimal parameters like window sizes, sorting keys, block sizes, etc. The following contributions are made which are as follows:

1) PSNM and PB are two algorithms that are proposed for progressive duplicate detection. It exposes several strengths.
2) This approach is suitable for a multiple pass method and an algorithm for incremental transitive closure is adapted.
3) To rank the performance, the progressive duplicate detection is measured using a quality measures.
4) Many real world databases are evaluated by testing the algorithms previously known.

There are three stages in this workflow which are as follows:

a) Pair selection
b) Pair wise comparison
c) *Clustering*

Only the pair selection and clustering stages should be modified for a good workflow.

IV. ALGORITHMS

A. Progressive SNM

The algorithm takes five input parameters: D is a reference to the data, which has not been loaded from disk yet. The sorting key K defines the attribute or attributes combination that should be used in the sorting step. W specifies the maximum window size, which corresponds to the window size of the traditional sorted neighborhood method. When using early termination, this parameter can be

set to an optimistically high default value. Parameter I defines the enlargement interval for the progressive iterations. For now, assume it has the default value 1. The last parameter N specifies the number of records in the dataset. This number can be gleaned in the sorting step, but we list it as a parameter for presentation purposes. Progressive Sorted Neighborhood Require: dataset reference D, sorting key K, window size W, enlargement interval size I, number of records N

Step 1: procedure PSNM(D, K, W, I, N)
Step 2: pSize□□calcPartitionSize(D)
Step 3: pNum □[N/pSize-W + 1)]
Step 4: array order size N as Integer
Step 5: array recs size pSize as Record
Step 6: order □sortProgressive(D, K, I, pSize, pNum)
Step 7: for currentI□□2 todW=Iedo
Step 8: for currentP □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)
Step 15: lookAhead(pair)

B. Progressive Blocking

The algorithm accepts five input parameters: The dataset reference D specifies the dataset to be cleaned and the key attribute or key attribute combination K defines the sorting. The parameter R limits the maximum block range, which is the maximum rank-distance of two blocks in a block pair, and S specifies the size of the blocks. Finally, N is the size of the input dataset. Progressive Blocking Require: dataset reference D, key attribute K, maximum block range R, block size S and record number N

Step 1: procedure PB(D, K, R, S, N)
Step 2: pSize □□calcPartitionSize(D)
Step 3: bPerP □□[pSize/S]
Step 4: bNum □□[N/S]
Step 5: pNum □□[bNum/bPerP]
Step 6: array order size N as Integer
Step 7: array blocks size bPerP as <Integer; Record[]>
Step 8: priority queue bPairs as <Integer; Integer; Integer>
Step 9: bPairs □{<1,1,->, . . . ,<bNum, bNum,->}
Step 10: order □sortProgressive(D, K, S, bPerP, bPairs)
Step 11: for i □0 to pNum - 1 do
Step 12: pBPs □□get(bPairs, i . bPerP, (i+1) . bPerP)
Step 13: blocks □□loadBlocks(pBPs, S, order)
Step 14: compare(blocks, pBPs, order)
Step 15: while bPairs is not empty do
Step 16: pBPs□□{}
Step 17: bestBPs□□takeBest([bPerP/4], bPairs, R)
Step 18: for bestBP belongs to bestBPs do
Step 19: if bestBP[1] _ bestBP[0] < R then
Step 20: pBPs□□pBPs U extend(bestBP)
Step 21: blocks □loadBlocks(pBPs, S, order)
Step 22: compare(blocks, pBPs, order)
Step 23: bPairs □bPairs U pBPs
Step 24: procedure compare(blocks, pBPs, order)

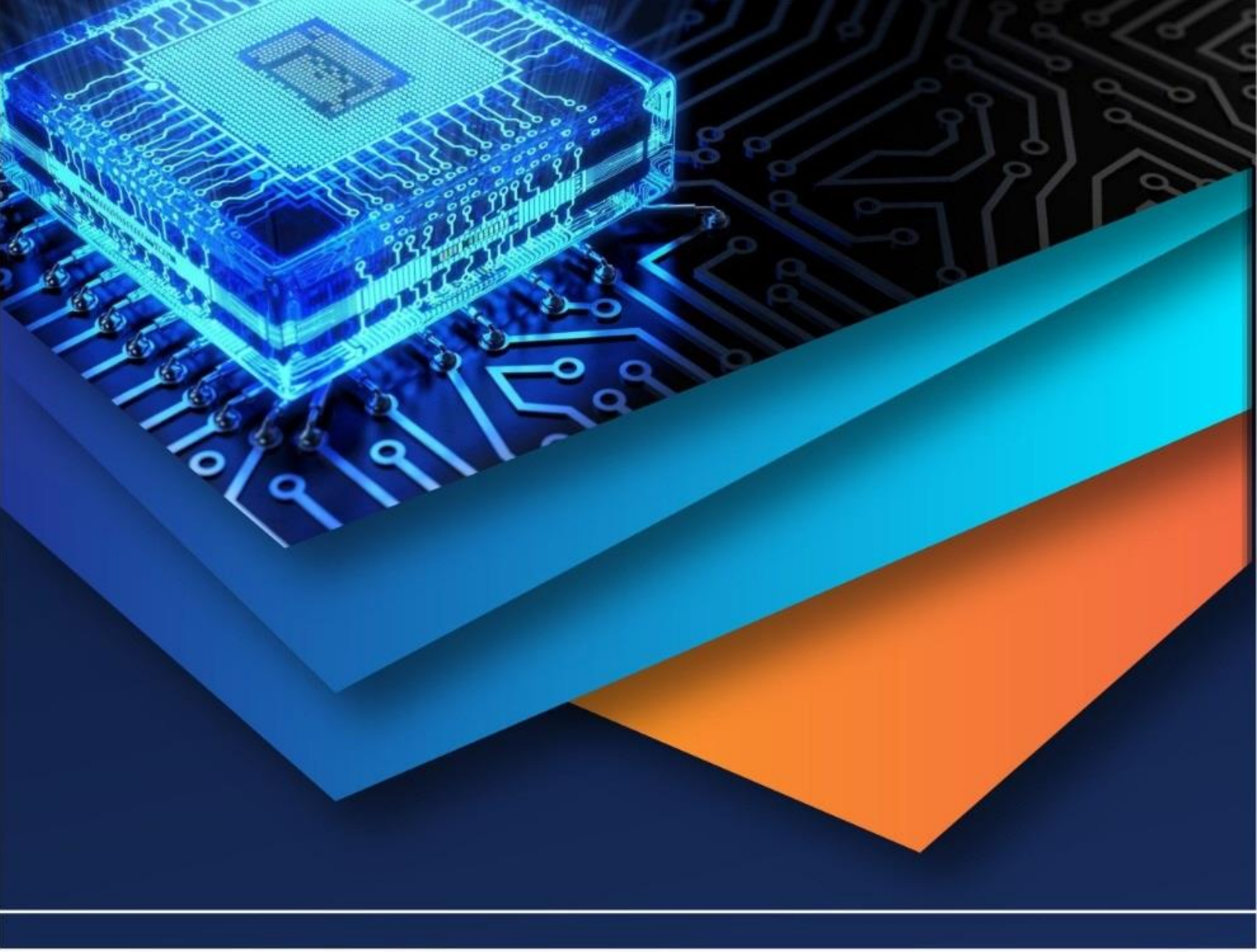
Step 25: for pBP belongs to pBPs do
Step 26: <dPairs,cNum> comp(pBP, blocks, order)
Step 27: emit(dPairs)
Step 28: pBP[2] □|dPairs|/ cNum

V. CONCLUSION

This paper introduced the progressive sorted neighborhood method and progressive blocking. Both algorithms increase the efficiency of duplicate detection for situations with limited execution time; they dynamically change the ranking ofcomparison candidates based on intermediate results to execute promising comparisons first and less promising comparisons later. To determine the performance gain of our algorithms, we proposed a novel quality measure for progressiveness that integrates seamlessly with existing measures. Using this measure, experiments showed that our approaches outperform the traditional SNM by up to 100 percent and related work by up to 30 percent For the construction of a fully progressive duplicate detection workflow, I proposed a progressive sorting method, Magpie, a progressive multi-pass execution model, Attribute Concurrency, and an incremental transitive closure algorithm. The adaptations AC-PSNM and AC-PB use multiple sort keys concurrently to interleave their progressive iterations. By analyzing intermediate results, both approaches dynamically rank the different sort keys at runtime, drastically easing the key selection problem. In future work, I want to combine our progressive approaches with scalable approaches for duplicate detection to deliver results even faster. In particular, Kolb et al. introduced a two phase parallel SNM [21], which executesa traditional SNM on balanced, overlapping partitions. Here, we can instead use our PSNM to progressively find duplicates in parallel.

REFERENCES

[1] S. E. Whang, D. Marmaros, and H. Garcia-Molina, “Pay-as-you-go entity resolution,” IEEE Trans. Knowl. Data Eng., vol. 25, no. 5, pp. 1111–1124, May 2012.
[2] A. K. Elmagarmid, P. G. Ipeirotis, and V. S. Verykios, “Duplicate record detection: A survey,” IEEE Trans. Knowl. Data Eng., vol. 19, no. 1, pp. 1–16, Jan. 2007.
[3] F. Naumann and M. Herschel, An Introduction to Duplicate Detection. San Rafael, CA, USA: Morgan & Claypool, 2010.
[4] H. B. Newcombe and J. M. Kennedy, “Record linkage: Making maximum use of the discriminating power of identifying information,” Commun. ACM, vol. 5, no. 11, pp. 563–566, 1962.
[5] 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. 9–37, 1998.
[6] X. Dong, A. Halevy, and J. Madhavan, “Reference reconciliation in complex information spaces,” in Proc. Int. Conf. Manage. Data, 2005, pp. 85–96.
[7] O. Hassanzadeh, F. Chiang, H. C. Lee, and R. J. Miller, “Framework for evaluating clustering algorithms in duplicate detection,” Proc. Very Large Databases Endowment, vol. 2, pp. 1282– 1293, 2009.
[8] O. Hassanzadeh and R. J. Miller, “Creating probabilistic databases from duplicated data,” VLDB J., vol. 18, no. 5, pp. 1141–1166, 2009.
[9] U. Draisbach, F. Naumann, S. Szott, and O. Wonneberg, “Adaptive windows for duplicate detection,” in Proc. IEEE 28th Int. Conf. Data Eng., 2012, pp. 1073–1083.
[10] 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. 185–194.
[11] 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 Proc. Conf. Innovative Data Syst. Res., 2007.
[12] S. R. Jeffery, M. J. Franklin, and A. Y. Halevy, “Pay-as-you-go user feedback for dataspace systems,” in Proc. Int. Conf. Manage. Data, 2008, pp. 847–860.
[13] C. Xiao, W. Wang, X. Lin, and H. Shang, “Top-k set similarity joins,” in Proc. IEEE Int. Conf. Data Eng., 2009, pp. 916–927.
[14] P. Indyk, “A small approximately min-wise independent family of hash functions,” in Proc. 10th Annu. ACM-SIAM Symp. Discrete Algorithms, 1999, pp. 454–456. Fig. 10. Duplicates found in the plista-dataset.
[15] U. Draisbach and F. Naumann, “A generalization of blocking and windowing algorithms for duplicate detection,” in Proc. Int. Conf. Data Knowl. Eng., 2011, pp. 18–24.
[16] H. S. Warren, Jr., “A modification of Warshall’s algorithm for the transitive closure of binary relations,” Commun. ACM, vol. 18, no. 4, pp. 218–220, 1975.
[17] M. Wallace and S. Kollias, “Computationally efficient incremental transitive closure of sparse fuzzy binary relations,” in Proc. IEEE Int. Conf. Fuzzy Syst., 2004, pp. 1561–1565.
[18] F. J. Damerau, “A technique for computer detection and correction of spelling errors,” Commun. ACM, vol. 7, no. 3, pp. 171–176, 1964.
[19] P. Christen, “A survey of indexing techniques for scalable record linkage and deduplication,” IEEE Trans. Knowl. Data Eng., vol. 24, no. 9, pp. 1537–1555, Sep. 2012.
[20] B. Kille, F. Hopfgartner, T. Brodt, and T. Heintz, “The Plista dataset,” in Proc. Int. Workshop Challenge News Recommender Syst., 2013, pp. 16–23.
[21] L. Kolb, A. Thor, and E. Rahm, “Parallel sorted neighborhood blocking with MapReduce,” in Proc. Conf. Datenbanksysteme in B€uro, Technik und Wissenschaft, 2011.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY