ISSN 2395-1621

Data De-duplication using Large Scale Pattern Search

[#1]Murte Vijaya, [#2]Mayuri Devtarse, [#3]Kishor Yoge, [#4]Chaudhai Revti

[1]vijayamurte2014@gmail.com
[2]devtarsemayuri@gmail.com
[3]kishoryoge308@gmail.com
[4]devtarsemayuri@gmail.com

[#1234]Department of Computer Engineering
JSPM's
Imperial College of Engineering & Research
Wagholi, Pune--412207

ABSTRACT

With increase of de-duplication in data sets of voter card or pan card, finding and removing the de-duplication is the key challenge. Record linkage is the process of matching records from many databases that mention to the same entities. When apply on a one database, this process is called as de-duplication. In this paper the investigation is completed to how to remove the de-duplication with the help of suffix arrays. Suffix array is better organized data structure for search pattern. This paper covers similarity metrics that are commonly used to spot similar field entries, and present a common set of duplicate detection algorithms that can identify nearly duplicate record in a database. It too covers several techniques for refining the efficiency and scalability of projected duplicate detection algorithms. It base on the algorithms, the paper presents how to eliminate and show the de-duplication from dataset.

ARTICLE INFO

Article History

Received: 28th May 2016

Received in revised form :

29th May 2016

Accepted: 31th May 2016

Published online :

1sr June 2016

Keywords: **String search, pattern matching, suffix array, suffix tree.**

I. INTRODUCTION

Databases play anvital role in today's' world. Many different areas depend on the correctness of databases to bring out operations. Therefore, the quality of the data kept in databases can have major effects on the system. An essential step in adding data from different sources is to identify and remove duplicate records that refer to the same entity. It is knownas De-duplication. String search is a well known problem: given a textB[0 . . . m−1] over some alphabet Σof size st=|Σ| and a pattern P[0 . . .k− 1], locate the occurrences of P in B. Several different query modes are possible: whether or not P occurs (existence queries); how many times P occurs (count queries); how many byte locations in B at which P occurs (locate queries); and a set of extracted contexts of B that includes each occurrence of P(context queries) [1].When B and P are provided on a one-off basis, sequential pattern search methods take O(m+ k) time. When B is fixed, and many patterns are to be processed, it is likely to be more efficient to pre-process B and construct an index. The suffix array is one such index, allowing locate queries to be answered in O(k+ log m+ y) time when there are y occurrences of P in B, using O(mlogm) bits of space in addition to B. But suffix arrays only provide efficient querying if B plus the index require less main memory than is available on the host computer, because several access are required to both. For big text, two-tier structures are needed, with an in-memory component consulted first in order to identify the data that must be retrieved from anon-disk index. As many businesses, government agencies and research projects collect increasingly huge amount of data, techniques that allow efficient processing, analyze and mining of such enormous databases have in recently years attracted concern from both academic and industry. One task that has been predictable to be of increasing importance in many application domains is the matching of records that related to the same entities from some databases. Often, information from multiple sources needs to be combined and joined in order to increase data quality, or to improve data to facilitate more detailed data analysis. The records to be matched frequently relate to entities that refer to people, such as clients or customers, patients, employees, taxpayers, students, or travelers.

II. LITERATURE SURVEY

De-duplication is necessary is for the building of web portals which integrated data from different pages possibly created in a distributed method by millions of people. The crucial

challenge in this task is to find a function that can determine when two records refer to the identical entity in malice of errors and conflicts in the data. One responsibility that has been recognized to be of rising importance in many application domains is the matching of records which arecount to the same entities from several databases. Many businesses practice de-duplication and record linkage techniques with the objective to de-duplicate their databases to rise data quality or compile mailing lists, or to match their data through organizations, for example for cooperative marketing and e-Commerce projects. Many government organizations are now always more employing record linkage, for example within and amongst taxation offices and departments of social security to recognize people who register for help many times, or who work and gather unemployment benefits. This isat present not pure which indexing technique is suitable for what type of data and what kind of record link or de-duplication application. This practice has newly been planned as an efficient domain liberated approach to multi-source information combination. The newidea is to add the BKVs(Bounded Key Value) and their suffixes into a suffix array based inverted index. A suffix array holds sequences or strings and their suffixes in an alphabetically arranged order. Indexing based on suffix arrays has efficiently been used for both English and Japanese databases. One of the best studied particular cases is edit distance, which permits to delete, insert and replace characters(by a unlike one) in both strings. If the different operations have different cost or the cost depend on the characters involved, weare speaking of general edit distance.

III. PROPOSED SYSTEM

Pattern matching is an play the important role of checking a given sequence of tokens for the presence of constituents of some pattern. The pattern, usually have the form of either sequences or tree structures. Pattern matching uses different fields likes of voter card to generate a Bounded Key Value(BKV) which is used for suffix array generation. A distributed database for the particular region is maintained. The data has p attributes a1; : : : ap, each of which could be textual or numeric. The objective of the system is to find the subset of pairs in the cross-product DxD that can be considered as duplicates. The data preparation step contains a parsing, data transformation and a standardization step. BKV is made by combining few characters of the selected fields, which can be changed dynamically according to the size of the fields. Once the size is determined it will be same for the rest of the records. This can be a grouping of characters and numbers. Size of the fields depends on the importance of several fields. Different fields will be given more weightage as compared to other fields. Similarity algorithm considers a group of function which calculates the similar match between two records based on the subset created from the suffix arrays. This algorithm executes operation based on the subset of the suffix array. Either it performs insertion, deletion or substitution. Suffix array is generated from BKV. A suffix array is a sorted array of all suffixes of a string. Dependent on the minimum size of suffix, numbers of suffixes are created. These suffixes are kept in suffix array. Suffix arrays are narrowlyinterrelated to suffix trees. Suffix array can be made by performing a depth first traversal on a suffix tree. A suffix tree for text T is a modified suffix tree in which the parent-child edges represent sequences of symbols from rather than single symbols; and in which internal nodes that only have a single child are removed.

IV. ARCHITECTURE

Here the architecture shows who the system is going to work: Database is a collection of data and in this project we are using the election database.
Cleaning means in front of name Miss or Mrs this comman part is not consider or clean. standardization means any data is convert in standard format. Like mobile number having +91 or 0 this part is not consider but only consider 10 digit mobile number this process called standardization . Indexing is main part in this system indexing require suffix of data and also Bounded key value. Bkv is provided certain limit after generation of bkv minimum suffix size is decided. Edit distance algorithm use and compare the pair and to apply limit and the similarity, non-similarity, possible similarity.

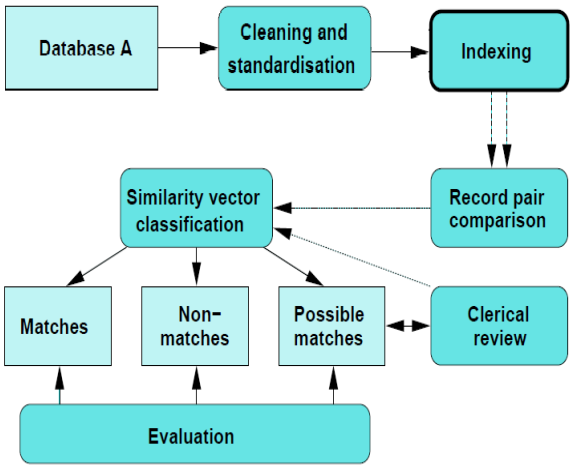

Figure 1: Architecture of the system

1) It is the distributed database which contains of records to be checked. These records must be store in the database.
2) Cleaning and standardization important task is to translate the raw data in well defined data.
3) In indexing step generates the record pairs of candidate records. Some candidaterecords thatare generated are compared in detail using comparison function.
4) In record pair comparison all the candidate records are compared that were generated by indexing and for comparison it uses some similarity algorithm.
5) Using some similarity algorithm like edit distance, jaro distance are used for strings are compare with each other.
6) If the duplicates are matched or not matched or possible matched evaluation process takes place.
7) Whatever similarity was found was revised.
8) Evaluation process is required to check what percentage of duplication is there.
9) After the evaluation review the duplicate result or report.

V. ALGORITHMS

1: Edit Distance: The edit distance between two strings S1 and S2 is the minimum number of edit operations of single

letter needed to transform the string S1 into S2 . There are three types of edit operations: like insert, delete and replace.

- insert a letter into the string,
- delete a letter from the string, and
- Replace one letter with a different letter in the string.

EDIT DISTANCE(s1,s2)

```
1   int m[I, j]=0
2   for i<-1 to |s1|
3   do m[i,0]=i
4   for j<-1 to |s2|
5   do m[I, j]=j
6   for i<-1 to |s1|
7   do for j<-1 to |s2|
8   do  m[I, j]= min{m[i-1,j-1]+ if (s1[i]= s2[j])then 0
    else 1fi,
9           m[i-1,j]+1,
10          m[I,j-1]+1}
11  Return m [|s1|,|s2|]
```

The edit distance are used to describes how similar or dissimilar two strings are by the number of steps it takes to turn from oneinto the other, where a step is defined as a single letter change. Example for edit distance algorithm is as following: What is edit distance between "pittin" and "sitteng" ? A minimal edit script that transforms the former into the latter is:

· pittin → kitten (replace of "k" for "p")

· sittin → sitten (replace of "e" for "i")

· sitten → sitteng(insertion of "g" at the end).

Therefore the edit distance between “kittin ” and “sitteng” is 3.

VI. RESULT

As in existing system Jaro similarity measure is used to compare suffixes.Praposed system uses edit distance which gives more similarity than existing Jaro. Table 3 shows result of Jaro and edit distance similarity for suffixes. From this it is clear that edit distance is more efficient and gives more similarity than Jaro for suffixes. From fig 3 shows time required for improved suffix array blocking and proposed suffix array blocking.

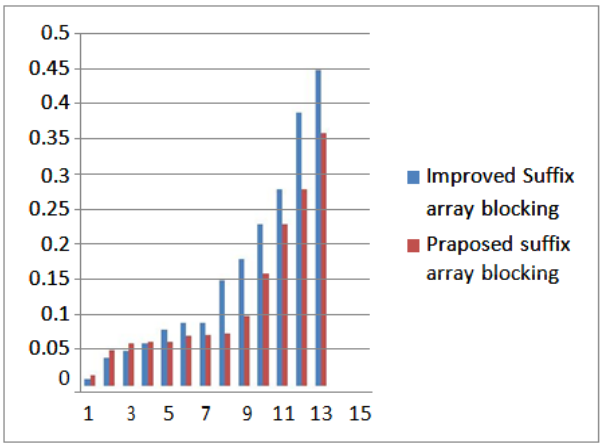

Fig 2. Time comparison graph.

Suffixes	Jaro Similarity	Edit Distance
2353-23534	0.84	0.87
2353-235345	0.79	0.83
2353-235345k	0.76	0.81
2353-235345ke	0.74	0.79
2353-235345kar	0.72	0.77
2353-235345kara	0.7	0.76
23534-235345	0.86	0.89
23534-235345k	0.82	0.86
23534-235345ka	0.8	0.84

VII. FUTURE DIRECTION AND CONCLUSION

One Huffman code can be used to classify the duplication in database more efficiently. This necessities extra efforts and more expertise. Also, in that we can use one artificial intelligence to find out the duplications and it becomes more powerful with time and experience. This will provide more accurateness as compared to any other alternative. But this growths the complexity of the system. It will display the possible duplicate records and give the appropriate result. In short, the future is bright indeed. Much will change in the years ahead, but one thing is certain: No more unethical means will be used to vote or to use privileged application. Proposed system will remove all the de-duplication from the data sets. So in sensitive things such as voting there duplication of voter cards can be avoided. We have carried out the detailed study on our proposed system and also the algorithm we are using that is edit distance algorithm.

VIII. ACKNOWLEDGMENT

We are working on this project under the guidance of Mrs. Pravin Nimbalker., Assistant Professor at JSPM imperial college of engineering and Research of Wagholi, Pune.

REFERENCES

[1] Simon Gog, Alistair Moffat, J. Shane Culpepper, Andrew Turpin, and Anthony Wirth, “Large-Scale Pattern Search Using Reduced-Space On-Disk Suffix Arrays”, IEEE transactions on knowledge and data engineering, vol. 26, no. 8, august 2014.

[2] Peter Christen,”A Survey of Indexing Techniques for Scalable Record Linkage and De-duplication”, IEEE transactions on knowledge and data engineering, vol. z, no. y, zzzz 2011.

[3] Ahmed K. Elmagarmid, Panagiotis G. Ipeirotis, Vassilios S. Verykios,Duplicate Record Detection: A Survey”, IEEE transactions on knowledge and data engineering, vol. 19, no. 1, january 2007.

[4]SunitaSarawagi,AnuradhaBhamidipaty,”InteractiveDedup lication using Active Learning”, IIT Bombay .

[5] Gonzalo Navarro." A Guided Tour To Approximate String Matching",Dept.of Computer Science,University of Chile,BlancoEncalada 2120 –Santigo-Chile.

[6] M.A. Jaro, "Advances in Record-Linkage Methodology as Appliedto Matching the 1985 Census of Tampa, Florida," J. Am. StatisticalAssoc., vol. 84, no. 406, pp. 414-420, June 1989.

[7] W. W. Cohen, P. Ravikumar, and S. Fienberg, "A comparison ofstring distance metrics for name-matching tasks," in *Workshop onInformation Integration on the Web, held at IJCAI'03*, Acapulco, 2003.

[8] A. Moffat, S. J. Puglisi, and R. Sinha, "Reducing space requirementsfor disk resident suffix arrays," in *Proc. 14th Int. Conf.DASFAA*, Brisbane, QLD, Australia, 2009, pp. 730–744.

[9] V. Mäkinen and G. Navarro, "Compressed compact suffix arrays,"in*Proc. Symp. CPM*, Istanbul, Turkey, 2004, pp. 420–433.

[10] A. Moffat, S. J. Puglisi, and R. Sinha, "Reducing space requirements for disk resident suffix arrays," in *Proc. 14th Int. Conf. DASFAA*, Brisbane, QLD, Australia, 2009, pp. 730–744.

