IJRSET NOVEMBER 2016 Volume 3, Issue 11 Pages: 11-17

ISSN 2394-739X

International Journal for Research in Science Engineering and Technology

PROGRESSIVE DUPLICATION DETECTION USING RABIN-KARP ALGORITHM

[1] P. Sundari, [2] S. Deepasamili

[1] Assistant Professor, [2] M.Phil. Research Scholar

[1] PG & Research Department of Information Technology, [2] PG & Research Department of Computer Science

[1, 2] Government Arts College (Autonomous), Coimbatore-18.

ABSTRECT: It is an important task of detecting exact duplicate records from the data source. Duplicate record detection is the problem of identifying records in the database that represent the same real-world entity. Duplicate records do not share a common key and that makes detecting the duplicates a difficult problem. An efficient and accurate content-based online duplicate detection method is a fundamental research goal to identify duplicate content on the large storage datasets. Despite the recent progress made duplicate detection, it remains very challenging to develop accurate duplicate detection mechanism for large-scale databases. This paper presents progressive duplicate detection algorithm that will increase the efficiency of finding duplicates from the web online dataset. This approach specifies a progressive duplication detection method using Rabin-Karp algorithm. The hash value is generated for each data in the tuples. This hash value is used for matching the data. Through experiments conducted, the algorithm achieves high precision and better accuracy in duplicate detection with many datasets. Rabin-Karp duplicate detection algorithm outperforms other duplicate detection algorithm in terms of throughput and efficiency.

Keywords: [Duplicate Detection, Hash value, Rabin-karp.]

1. INTRODUCTION

Today most of the databases are generated to manage web page contents and create an interactive environment to answer the user queries through the web. These dynamic web databases are supposed to have a much larger amount of structured information and also have a rapid development rate when compared to the static web. Most of the web databases are accessed through query interface where the user can submit their queries. The submitted queries will be processed at the web server. Based on the query, it retrieves the result from the database.

The problem of duplicate detection is long as established and certain communities have worked on it using different terminology. The statistics community calls the duplicate problem as record correlation. Other communities like machine learning or natural language processing investigate similar problems such as identity uncertainty, object identification, object combination, co-reference resolution or entity resolution.

Duplicate detection aims to find similar objects from various data sources. That indicates the same real world entity, where these particular objects might be inaccurate and insufficient. In addition, there exists a unique hash value for the objects that would allow directly to find the duplicate record. When handling with an enormous amount of data, it is important to have framework and tested methodology to filter duplicate records from the database. This considers the end results relevant to the queries. In duplicate detection process, fields are matched with web dataset to detect the duplicate records. Using definite matching technique as part of preprocessing, records that are absolutely the same in all relevant matching fields can be absorbed.

Duplicate Record Detection: A survey [5] predicts the duplicate record detection. Similarity metrics help to detect the common fields within the record, and present mechanism of duplicate detection to detect duplicate records in a database. There are numerous techniques for improving the efficiency and scalability of exact duplicate detection algorithms.

A study and survey on various progressive duplicate detection mechanisms [14][15]shows various technique like Progressive Blocking and Progressive Neighborhood to detect the duplicate with a minimum execution time and without interrupting any quality of dataset.

Progressive Sorted Neighborhood Method (PSNM) uses parallel approach for finding exact duplicate records in datasets. Progressive Blocking algorithm is applied to large datasets to discover duplicates in minimum time duration.

In this paper, the new approach is targeted mainly on adequate duplicate detection. The main goal of this research work is to detect definite and indefinite duplicate by using duplicate detection framework. This paper presents progressive duplicate method using Rabin-Karp algorithm based on unique hash value generation. The hash value is generated for each data in the tuples and these values are unique. It is used for matching the data.

2. LITERATURE REVIEW

This section analyzes the previous approaches for duplicate detection.

Pay-As-You-Go Entity Resolution: [16]

Pay-As-You-Go entity resolution is proposed to calculate the hints using all partitions. The algorithm uses a global ranking for the comparisons and considers the limited amount of available main memory. The third issue of the algorithms introduced by Whang et al. relates to the proposed pre-partitioning strategy. It also progressively solves the multi-pass method and transitive closure calculation, which are essential for a completely progressive workflow. Finally, we provide a more extensive evaluation on considerably larger datasets and employ a novel quality measure to quantify the performance of our progressive algorithms.

Progressive Sorted Neighborhood Method (PSNM):

The Progressive Sorted Neighborhood Method (PSNM) is based on the traditional Sorted Neighborhood Method. PSNM sorts the input data using a predefined sorting key and compares records that are within a window of records in the sorted order. The intuition is that records that are close in the sorted order are more likely to be duplicates than records that are far apart, because they are already similar with respect to their sorting key. More specifically, the distance of two records in their sort ranks (rank-distance) gives PSNM an estimate of their matching likelihood. The PSNM algorithm uses this intuition to iteratively vary the window size, starting with a small window of size two that quickly finds the most promising records. This static approach has already been proposed as the sorted list of record pairs. The PSNM algorithm differs by dynamically changing the execution order of the comparisons based on intermediate results (Look-Ahead).

Furthermore, PSNM integrates a progressive sorting phase (Magpie Sort) and can progressively process significantly larger datasets.[15]

Progressive Blocking:

In contrast to windowing algorithms, blocking algorithms assign each record to a fixed group of similar records (the blocks) and then compare all pairs of records within these groups. Progressive Blocking (PB) is a novel approach that builds upon an equidistant blocking technique and the successive enlargement of blocks. Like PSNM, it also pre-sorts the records to use their rank-distance in this sorting for similarity estimation. Based on the sorting, PB first creates and then progressively extends a fine-grained blocking. These block extensions are specifically executed on neighborhoods around already identified duplicates, which enables PB to expose clusters earlier than PSNM.[15]

Parallel Progressive Duplicate Detection Method (PPSNM):

PPSNM is used for duplicate record detection and duplicate record deletion. On one hand, the extraction of PPSNM is faster than PSNM due to the Map Reduce concept. On the other hand, the improvement in detection effectiveness is consistently observed in two applications. This is achieved by indexing the PPSNM with Map Reduce [14].

Probabilistic Databases from Imprecise Time-Series Data:

It is a novel approach to create table-level variation databases from (imprecise) time-series data. To the best of our knowledge, this is the first work that brings a generic solution for creating potential databases from random time series, which can work online as well as an offline sphere.[9]

A New Multiple-Pattern Matching Algorithm:

Different from those existing efforts as it is interested in building a graph transition relation and characterized linked list. Search based on matching technique for multi-pattern matching that will match a large amount of dataset and can easily be compared with any existing multi-pattern matching application.[3]

Clone Detection Using Rabin-Karp Parallel Algorithm:

Code duplication is a usual problem found in software development. It could produce various clones. Clone is a block of code. It reproduces many times on the source code. The existence of clone is highly probable to intensify the risk on software progress. Technique for detecting clone includes textual, lexical, syntactic, and semantic approach. In this algorithm, estimate by utilize various aspects based on the condition of every pair. It progress a novel method to detect clone by using Rabin-Karp parallel algorithm. The algorithm is more efficient than the Rabin-Karp algorithm. In cases of estimation, it constructs a detecting tool competent of processing source code in both lexical and syntactic manner. This algorithm estimate the performance of the proposed method. To do so, contrast parallel Rabin-Karp to Traditional Rabin-Karp. The result express parallel Rabin-Karp could gain best performance.[12]

Plagiarism Detection by using Karp-Rabin and String Matching Algorithm Together:

This suggests new plagiarism detection techniques by using Rabin-Karp algorithm and string matching algorithm. Here data reliance expression file, take out keyword and utilize twin algorithm technique which conquer all problems of matrix, parallel hash value as well as string matching, which detects plagiarized programs or documents by using hash function. Experiments have well verified its effectiveness over existing tools and it is appropriate in practice.[13]

3. PROPOSED WORK

This section explains how the proposed Rabin-Karp algorithm are important

for selecting and extracting pairs of record from the dataset to classify the duplicate fields. Progressive Duplication detection method using Rabin-Karp algorithm is proposed based on unique hash value. The hash value is generated for each group of data in the table. This hash values are used for matching with previous values among the table. This matching mechanism would help to identify the duplicate record within the tables and match query result with different datasets. This research mainly focused on recognizing the duplicate records that include some predefine comparison rules. Matching rules that have been implemented, especially that is learned offline by some learning technique with a set of training examples. This technique performs well in a large set of databases, where all kind of expected database can be frequently accessed.

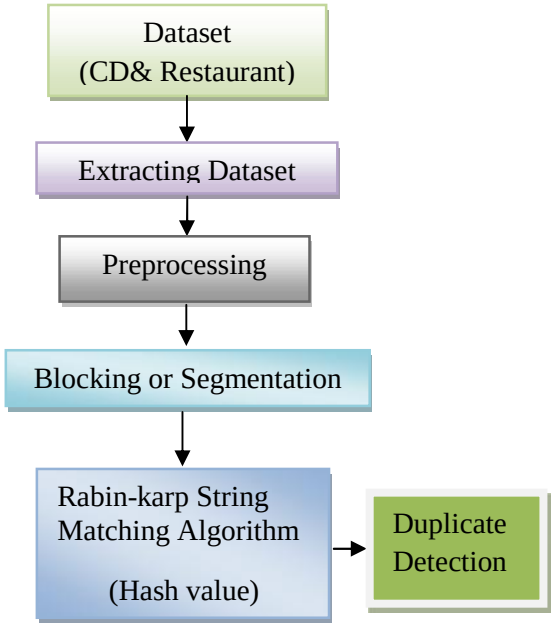

Figure 1-Architecture Diagram

3.1 Dataset Extraction

A dataset is an ontology that expresses a dataset vocabulary in a modular way. Its components are selected based on measures that indicate their importance in capturing the core knowledge in a dataset. In this duplicate detection process CD and Restaurant dataset are selected for further processing. In this records are extracted from the data source and each extracted data have a unique identity.

3.2 Data Preprocessing

In pre-processing step, data is sorted and exact matching records are deleted. This is done by comparing all the fields and collecting the data. This ensures that the same data doesn't exist and it is a basic check that can be done. Extracted data are processed and the total number of records are counted in the dataset. Filter out the training dataset using a specified algorithm and take the data based on condition. This process achieves a better solution and compare with many datasets and retrieve the require data.

3.3 Blocking Mechanism

Blocking is another important step in duplicate detection process, where the records are divided into segments or blocks. This will improve the processing speed and easily match with various segments of data. This term simply describes the grouping of data within a record. Blocking algorithms use some blocking key to partition a set of records into disjoint partitions. The comparison of record pairs is then limited to records within the same partition. Thus, the overall number of comparisons is greatly reduced.

3.4 Rabin-Karp Algorithm

Record matching is the process of identifying the duplicate records. The Rabin-Karp algorithm is used for identifying the duplicate contents in the dataset. The main aim of the record matching is to establish records in the similar or various databases that direct to the same real-world object. In the considerable ironic sphere, the same problem has occurred numerous times within a database.

Hash Value Generation process

The key to the Rabin–Karp algorithm's performance is the efficient computation of hash value of the successive substrings of the text. The Rabin finger print is a familiar and effective rolling hash function. The Rabin

fingerprint treats each substring as a number in base, the base being generally a large prime. For example, if the substring is "hi" and the base is 101, and then the hash value would be $104 \times 101^1 + 105 \times 101^0 = 10609$ (ASCII of 'h' is 104 and of 'i' is 105). Systematically, this algorithm is only same to the true number in a non-decimal system description, because for the example have the "base" less than one of the "digits". Let hash function for a much more detailed discussion. The important benefit achieved by using a rolling hash such as the Rabin fingerprint is that it is possible to calculate the hash value of the next substring from the earlier one by doing only a constant number of operations, independent of the substring's lengths. [17] For example, if we have text "abracadabra" and we are searching for a pattern of length 3, the hash of the first substring, "abr", utilizing 101 as base is: ASCII a = 97, b = 98, r = 114. Hash ("abr") = $(97 \times 101^2) + (98 \times 101^1) + (114 \times 101^0) = 999{,}509$. After then calculate the hash of the next substring, "bra", from the hash of "abr" by subtracting the number further added for the first 'a' of "abr", example 97×101^2, multiplying by the base and adding for the last a of "bra", i.e. 97×101^0. Like so: base old hash old 'a' new 'a' Hash ("bra") = $[101 \times (999{,}509 - (97 \times 101^2))] + (97 \times 101^0) = 1{,}011{,}309$. If the substrings in question are lengthy, this algorithm accomplishes great savings compared with many other hashing schemes.

Pseudo code (Hash Generation)

```
def hash(astring, tablesize)
//compute hash function using strings and
tables//
sum = 0        //initializing//
for pos in range(len(astring))
sum = sum + ord(astring[p])
 //processing string and pattern//
return sum%tablesize
```

After completing matching process based on the hash value it detects the duplicate. If hash values don't have the same identity, it does not indicate a duplicate warning. Otherwise the hash values are same it will denote the duplicate record. The proposed system is Progressive Duplication detection method using Rabin-Karp Algorithm. The hash value is generated for each data in the tables and this value is used for matching the data.

- Compute the "signature" of the pattern.
- Compute the "signature" of each substring of the text.
- Scan text until signature matches for possible match, so perform string comparison.

Pseudocode:

```
1 function RabinKarpSet(string s[1..n], set of
string subs, m):
//computing string matching using Rabin-karp
hash function//
2. Set hsubs := emptySet //initialization//
3. for each sub in subs
4.insert hash(sub[1..m]) into hsubs
//insert string//
5. hs := hash(s[1..m]) //assign number of
string from the data set//
6. for i from 1 to n-m+1
7. if hs ∈ hsubs and s[i..i+m-1] ∈ subs
//comparision of signature with every string//
8. return i
9. hs := hash(s[i+1..i+m])
10. return not found
```

4. EXPERIMENTAL RESULT AND DISCUSSION

The prior sections changed several major aspects of Rabin-Karp's original algorithm into a new application using dataset. The primary goal is to determine how newly created application can outperform. Compared to existing algorithms this Rabin-Karp algorithm's performance is increased. The following tables represent the accurate values of current process and existing values.

Algorithm/No.of Records	200	400	600	800
PSNM	82%	78%	77.5%	76%
PB	84%	80%	81%	78%
PPSNM	88%	82%	80%	82%
RABIN	94%	96%	92%	90%

Table 1- Accuracy of algorithm

Algorithm/No. of Records	200	400	600	800
PSNM	3.5ms	5ms	5.5ms	5.8ms
PB	3ms	4.2ms	4.8ms	5ms
PPSNM	2.4ms	3.8ms	4.2ms	4.6ms
RABIN	2ms	3ms	3.9ms	4.2ms

Table 2- Processing Time

Algorithm/No.of Records	200	400	600	800
PSNM	89 %	86 %	85.5 %	84%
PB	90 %	88 %	87%	85.8 %
PPSNM	94 %	90 %	89%	87%
RABIN	96 %	94 %	93.5 %	91%

Table 3- Throughput values

Table 1 presents the average accuracy on each dataset which is above 96% for proposed system and suddenly drops for existing system as it operates on multiple datasets. It is represented in figure2 (accuracy chart). Table 2 shows the time performance values with respect to a pruning factor of the existing system and proposed system. It shows that if pruning factor has increased the runtime also increases but in the proposed system there is no user intrusion to provide pruning factor. It is indicated in figure3 (processing time chart). Table 3 shows the runtime of CD dataset which is artificial dataset polluted by some dirty data. It shows the result on unsorted CD dataset and also compares with the result if test performed on a dataset with respect to depth and average string size. It is mentioned by figure4 (throughput chart).

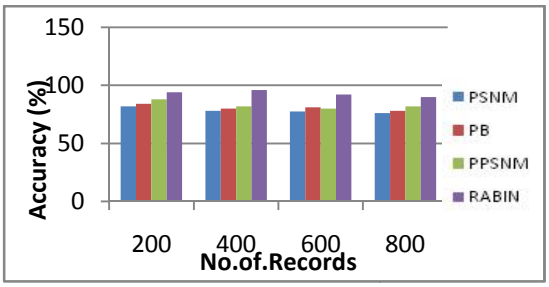

Figure 2- Accuracy Chart

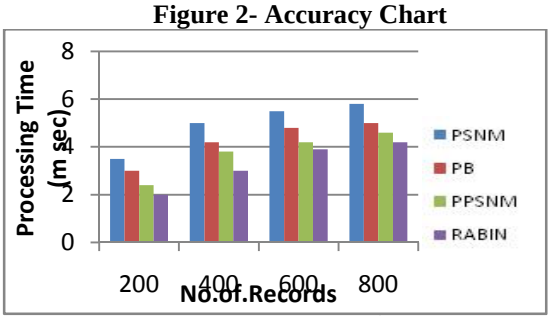

Figure 3-Processing Time Chart

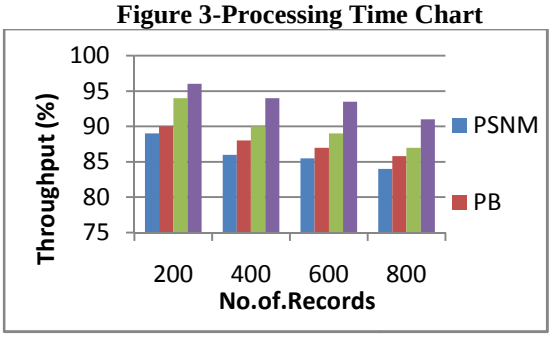

Figure 4- Throughput chart

This algorithm is designed based on varying one parameter at a time and holding all other parameters constant. CDs dataset are used for detecting the duplicate records. This section provides a comparative analysis of the proposed algorithm with the previous method. The proposed Rabin-Karp process is to improve the efficiency and accuracy. This study compares existing duplicate detection method such as PSNM, PB, and PPSNM with proposed Rabin-Karp progressive duplication detection method using CD dataset.

CONCLUSION

The duplicate detection has been one of the most important techniques for data redundancy and duplication. The methodology

proposed to avoid the duplication is the unique hash value based Rabin-Karp algorithm, which provides better performance and accuracy than the existing algorithm. This approach gives a better performance with various datasets using hash value based record matching. The experimentation of the proposed algorithms showed significant results. This approach is implemented with web datasets as cd and restaurant. The various duplicate detection algorithms are evaluated on different factors such as accuracy, processing time and throughput using cd and restaurant dataset. The results showed that the proposed Rabin-Karp algorithm is better than the previous algorithm.

REFERENCE

[1]. F.J.Damerau, "A technique for computer detection and correction of spelling errors," Commun. ACM, vol. 7, no. 3, pp. 171–176, 1964.

[2]. X. Dong, A. Halevy, and J. Madhavan, "Reference reconciliation in complex information spaces," in Proc. Int. Conf. Manage. Data, 2005, pp. 85–96.

[3]. 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.

[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. 18–24.

[5]. 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.

[6]. 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.

[7]. M.A.Hern_andez and S. J. Stolfo, "Real-world data are dirty: Data cleansing and the merge/purge problem," Data Mining Knowl. Discovery, vol. 2, no. 1, pp. 9–37, 1998.

[8]. 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.

[9]. 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]. F. Naumann and M. Herschel, An Introduction to Duplicate Detection. San Rafael, CA, USA: Morgan & Claypool, 2010.

[11]. 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.

[12]. Rahadian Dustrial Dewandono, Fahmi Akbar Saputra, Siti Rochimah"Clone Detection Using 8. 8. Rabin-Karp Parallel Algorithm" The Proceedings of The 7th ICTS, Bali, May 15th-16th, 2013 (ISSN: 9772338185001)

[13]. Sonawane Kiran Shivaji, Prabhudeva S "Plagiarism Detection by using Karp-Rabin and String Matching Algorithm Together" International Journal of Computer Applications (0975 – 8887) Volume 116 – No. 23, April 2015.

[14]. Shubhangi Anandrao Dhane1, Prof. Amrit Priyadarshi2 "Parallel PSNM Duplicate Record Detection with Map Reduce" International Journal of Advanced Research in Computer and Communication Engineering Vol. 5, Issue 6, June 2016

[15]. Thorsten Papenbrock, Arvid Heise, and Felix Naumann" Progressive Duplicate Detection"IEEE trans May 2015.

[16]. 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.

[17]. Wikipedia The free Encyclopedia en.wikipedia.org/wiki/hash value, string matching algorithms.