ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 4, April 2015

EDDDS: An Efficient Duplicate Data Detection System

Bhavana Dhake[1], Dr.S.S.Lomte[2], Prof.Y.R.Nagargoje[3], Prof.R.A.Auti[4], Prof.B.K.Patil[5]

ME Student, CSE Department, Dr.SQIET College, Aurangabad, India[1]

CSE Department, Dr.SQIET College, Aurangabad, India[2,3,4]

Abstract: Duplicate Detection is critical task of any database of any organization. Duplicates are nothing but the same real time entities or objects are presented in the form of different structure and in the different formats. We can find out the duplicates in relational data, in complex data and hierarchical data like XML. There are lots of works already presented in the past for finding the duplicates in the relational data. But nowadays there is more focus on finding duplicates in the XML data. Because of XML is very popular for data storing and extensively used for data exchange between the organizations. Here we have done an extensive literature survey on this topic and proposed a duplicate detection method that incorporates some of the existing paper's ideas and some of our original ideas. In addition to improving the efficiency and effectiveness, we also checks for its typographical errors when comparing the two XML elements. To test the correctness of our method, we are comparing it with existing duplicate detection system, and giving more focus on how we get higher precision and recall values in the various datasets we have used.

Keywords: Duplicate detection, record linkage, entity resolution, XML, Bayesian networks, data cleaning, optimization

I. INTRODUCTION

Today all the applications, business process & decisions are based on electronic data. In this data quality plays an central role. There are many sources of errors which affects the quality of data. We are here focusing on one type of error that is fuzzy duplication or duplicates. Duplicates means repenting the same real world entities in more than one forms. What makes duplicate detection a nontrivial task is the fact that duplicates are not exactly equal, often due to errors in the data. Consequently, we cannot use common comparison algorithms that detect exact duplicates. Instead, we have to compare all object representations, using a possibly complex matching strategy, to decide if they refer to the same real-world object or not.

Due to its highly practical relevance in data cleaning and threshold.

In this paper, we first present a probabilistic duplicate detection algorithm for hierarchical data called XMLDup. This algorithm considers both the similarity of attribute contents and the relative importance of descendant elements, with respect to the overall similarity score. The algorithm presented here extends our previous work by two things 1)significantly improving efficiency and 2) showing a more extensive set of experiments

II. EXISTING SYSTEM

A. Efficient and Effective Duplicate Detection in Hierarchical Data :

Luis L. et. al. [1] proposed his own unique method for XML data duplication called as XMLDup. The basis of XMLDup is Bayesian network. Bayesian network used to determine the probability of two XML data elements which is used to be compared for duplicate detection. Here authors not only consider the information within the elements but also the way that information is structured. XMLDup mainly uses 2 types of probabilities Prior probability and Conditional probability. It requires little user intervention for the fixing of threshold values. The model is very flexible. XMLDup gives good results in the form of precision and recall. To check the run time efficiency of XMLDup, a network pruning strategy is given. This works on either of the 2 ways that is lossless approach and lossy approach. In lossless approach there is no impact on the final result but in the lossy approach there is slightly impact on the recall parameter.

B.Eliminating fuzzy duplicates in data warehouse :

S. Chaudhuri et. al. [8] develops an algorithm to remove duplicates in dimensional tables in data warehouse called as DELPHI (**D**uplicate **EL**imination in the **P**resence of **HI**erarchies) which reduces the number of false positives without missing out on detecting duplicates. Authors use dimensional hierarchy which consists of a chain of relations linked by key – foreign key dependency to develop high quality duplicate elimination algorithm and then it evaluates on real datasets from an operational data warehouses. Here final duplicate detection function is a weighted voting of the predictions from using co-occurrence similarity function and textual similarity function.

C.DogmatiX Tracks down Duplicates in XML :

F. Naumann et. al. [2] proposed a framework for both efficiency & effectiveness in duplicate detection called as DogmatiX. DogmatiX stands for **D**uplicate **O**bjects **G**et **MAT**ched **I**n **X**ml. The framework mainly contains three

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 4, April 2015

steps: 1) Candidate Definition, 2) Duplicate Definition, and 3) Duplicate Detection. Here first two steps are carried out in offline mode when system setup & last step is carried out in online mode where actual algorithm exists. Candidate definition gives which objects we want choose for duplicate detection. Duplicate definition characterizes which portion of actual data is used for selection to find out duplicates. Duplicate detection performs six various sub-steps on actual data to detect duplicates in duplicate candidates. Here first three prepare the data for comparisons, whereas the remaining three perform the actual duplicate detection. While comparing XML elements, DogmatiX not only consider their data values but also the similarity of the children, parents, structure etc.

D.Structure-Based Inference of XML Similarity for Fuzzy Duplicate Detection :

M. Weis et. al. [3] proposed a unique method for Fuzzy duplicate detection in semi structured or hierarchical XML data. It not only focuses on the duplicate status of the children nodes but also gives more importance to the probability of descendants being duplicates. Probabilities of two XML elements are efficiently calculated with the help of Bayesian network model. This model derives from the structure of the XML objects that being compared. This algorithm provides great flexibility in its configuration, by allowing the use of different similarity measures for the data values and different conditional probabilities to join the similarity probabilities of the XML elements to be compared. This algorithm gives high precision and recall values while data sets contains higher amount of errors and missing information.

E. Matching XML Documents in Highly Dynamic Applications :

A Kade et. al. [4] proposed scheme where matching of XML documents are carried out in the Highly dynamic applications like web and peer-to-peer system. For highly dynamic application systems requires great effort for document management system. The author takes full advantage of the flexibility of XML documents for matching the similar XML documents. This unique method solves the matching problem of XML documents i. e. the problem of defining which parts of two XML document contains the same information. Matching is the first step of integration process. This approach is unique in the sense; it joins similarity information from the content of the elements with information from the structure of the documents. The total work is divided into three parts to calculate similarity between two XML documents: 1) the node's name, 2) the element's content's, and 3) the node's path.

F. Finding Similar Identities among Objects from Multiple Web Sources:

J. Carvalho et. al. [18] proposed approach to find out similar identities among objects from multiple web sources. In this approach, identification of objects is works like a relational join operation where similarity function takes as place of the equality condition. This approach is used to identify objects more complexly structured like an XML documents. Here authors proposed four different strategies to define the similarity function using vector space model. When comparing objects that represent same real world entities like as a case of movies and restaurant datasets, any one of the four strategies might be applied. It totally depends on which strategy is semantically related to given four strategies.

G.Duplicate Detection through Structure Optimization

Pavel C. et. al. [10] proposed unique method which automatically restructures database objects in order to take full advantage of the relations between its attributes. This new structure of objects reflects the relative importance of the attributes in the database and avoids doing the manual selection. This approach gives the hierarchal structure of objects attributes. It does not required user An easy way to comply with the conference paper formatting requirements is to use this document as a template and simply type your text into it. intervention. This approach depends on the basis that data attributes should be placed in the structure according to their overall importance in distinguishing between two objects. This approach maps attributes from original object structure into a new structure level with their relevance to the duplicate detection process.

HXML Path Matching for Different Hierarchy Order of Elements in XML Documents :

S. Intakosum et. al. [12] presents a method called as PathMatch which helps to find out semantic similarity rate by cost matrix model between the two XML paths with the help of edit distance algorithm. PathMatch similarity function is an idea of path matching that improves the PathSim algorithm. The steps for the PathMatch algorithm is as follows

1) Create a matrix M with m rows and n columns where m is equal to or less than n.
2) Fill dissimilarity rate of every row until finish by compare with each column.(1 – Sim(A[i], B[j])).
3) Sum the minimum number of dissimilarity rate of each column.
4) Calculate PathMatch similarity rate

III. PROPOSED SYSTEM

The goal of the system is to design & implement the method which find out duplicates in the XML data elements. And also check for its typographical errors when comparing two XML elements. The concern with accuracy was later approached by Carvalho and da Silva, in [12]. Although not specifically focused on XML, their work proposes a solution to the problem of integrating tree-structured data extracted from the web. Two object representations, e.g., two hierarchical representations of person elements, are compared by transforming each into a vector of terms and using a variation of the cosine measure to evaluate their similarity [13]. The hierarchical structure of object representations is mostly ignored, and a linear combination of weighted similarities is used to account for the relative importance of the different fields within the

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 4, April 2015

vectors. The authors show that this simple strategy manages to achieve high precision values in a collection of scientific publications. Nevertheless, and because of its more general nature, their approach does not take advantage of the useful features existing in XML databases, such as the element structure or tag semantics.
Only more recently has research been performed with the specific goal of discovering duplicate object representations in XML databases [5], [6], [8], [10]. These works differ from previous approaches since they were specifically designed to exploit the distinctive characteristics of XML object representations: their structure, textual content, and the semantics implicit in the XML labels. We briefly describe the main features of these methods here, and refer readers to [9] for a detailed theoretical and experimental comparison of these approaches.

The DogmatiX framework aims at both efficiency and effectiveness in duplicate detection [5] . The framework consists of three main steps: candidate definition, duplicate definition, and duplicate detection. Whereas the first two provide the definitions necessary for duplicate detection (i.e., the set of object representations to compare and the duplicate classifier to use), the third component includes the actual algorithm, an extension to XML data of the work of Ananthakrishna et al. [3].

The XMLDup system first proposed in [6] uses a Bayesian Network model (BN) for XML duplicate detection. Milano et al. propose a distance measure between two XML object representations that is defined based on the concept of overlays [8]. An overlay between two XML tree U and V is a mapping between their nodes, such that a node u 2 U, is mapped to a single node v 2 V if, and only if, they have the same path from the root. This measure is then used to perform a pairwise comparison between all candidates. If the distance measure determines that two XML candidates are closer than a given threshold, the pair is classified as a duplicate.

Finally, SXNM (Sorted XML Neighborhood Method) [10] is a duplicate detection method that adapts the relational sorted neighborhood approach (SNM) [14] to XML data.Like the original SNM, the idea is to avoid performing useless comparisons between objects by grouping together those that are more likely to be similar.The Methods are as follows :

A. bayesian Network For Duplicate Detection :
We now present the XMLDup approach to XML duplicate detection. We first present how to construct a Bayesian Network model for duplicate detection, and then show how this model is used to compute the similarity between XML object representations. Given this similarity, we classify two XML objects as duplicates if it is above a given threshold.

1) Let us consider the two XML elements that represent the same persons which are shown as trees in figure1. Here both represent object person named as ‘prs’. These elements have two attributes that are name & dob (date of birth) & also two child node as pob (place of birth) & cnt (contact).

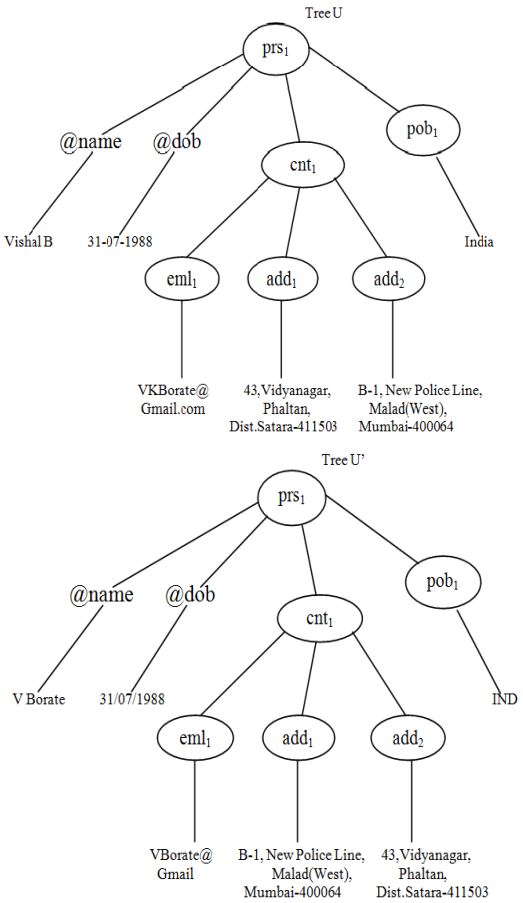

Figure1.Two XML elements that represents the same person.

Again at next level contact contains add1 (correspondence address), add2 (permanent address) & eml1 (email). Here leaf elements have a text node which contains the data. For example name has text node consists of text as a “Vishal B” as its value. In this example the ultimate goal of our method is to detect the both the XML elements as duplicate with respects to their different data values.

To carry out this we compare first value of attributes of both tree with each other i.e. name & dob and then check for its children node i. e. pob & cnt. Furthermore, pob node are duplicate depending on whether or not their values are duplicates, and the cnt node are duplicate depending on whether or not their children nodes i. e. eml and add are duplicates. This process carried out until the leaf nodes are reached.

Figure2 represents the Bayesian Network to compute the similarity between 2 XML objects that represent same elements which is shown in figure 1. In this type of network the node prs11 represent the possibility of node prs1 in the XML TreeU with respect to the duplicate of node prs1 in the XML TreeU’.

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 4, April 2015

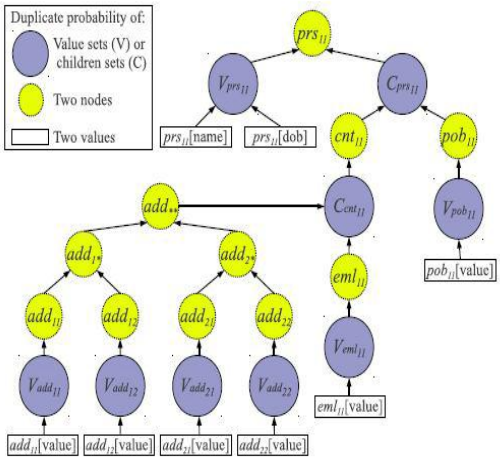

Figure2 BN to compute the similarity of the trees in Figure 1.

This root node prs11 has two child node Vprs11 and Cprs11 where node Vprs11 which represent the possibility of attribute values in the prs nodes being duplicates and node Cprs11 which represent the possibility of children nodes of the prs nodes being duplicates. In detailed, node Vprs11 contains value of attributes as prs11[name] and prs11[dob] as shown in rectangle which checks for its duplication. Node Cprs11 contains node pob11 and cnt11 where node pob11 which represent the possibility of node pob1 in the XML TreeU with respect to the duplicate of node pob1 in the XML TreeU'; node cnt11 represent the possibility of node cnt 11 in the XML TreeU with respect to the duplicate of node cnt1 in the XML TreeU'.

B.Accelerating The Bn Evaluation :

In order to improve the BN evaluation time, we propose a lossless pruning strategy. This strategy is lossless in the sense that no duplicate objects are lost. Only object pairs incapable of reaching a given duplicate probability threshold are discarded.

As stated before, network evaluation is performed by doing a propagation of the prior probabilities, in a bottom up fashion, until reaching the topmost node. The prior probabilities are obtained by applying a similarity measure to the pair of values represented by the content of the leaf nodes. Computing such similarities is the most expensive operation in the network evaluation, and in the duplicate detection process in general. Therefore, the idea behind our pruning proposal lies in avoiding the calculation of prior probabilities, unless they are strictly necessary.

The strategy follows the premise that, before comparing two objects, all the similarities are assumed to be 1 (i.e., the maximum possible score). The idea is to, at every step of the process, maintain an upper bound on the final probability value. At each step, whenever a new similarity is computed, the final probability is estimated taking into consideration the already known similarities and the unknown similarities that we assume to be 1. When we verify that the network root node probability can no longer achieve a score higher than the defined duplicate threshold, the object pair is discarded and, thus, the remaining calculations are avoided.

IV. CONCLUSION

We proposed the novel method which can find out duplicates in the XML data elements with the help of Bayesian network . We will evaluate the results of recall values for those datasets which gives false positive results.

V. ACKNOWLEDGMENT

We would like to thank our guide Dr.S.S.Lomte, their guidance and feedback during the course of the project. We would also like to thank our department for giving us the resources and the freedom to pursue this project.

REFERENCES

[1]. E. Rahm and H.H. Do, "Data Cleaning: Problems and Current Approaches," IEEE Data Eng. Bull., vol. 23, no. 4, pp. 3-13, Dec. 2000.
[2]. F. Naumann and M. Herschel, An Introduction to Duplicate Detection. Morgan and Claypool, 2010.
[3]. R. Ananthakrishna, S. Chaudhuri, and V. Ganti, "Eliminating Fuzzy Duplicates in Data Warehouses," Proc. Conf. Very Large Databases (VLDB), pp. 586-597, 2002.
[4]. D.V.Kalashnikov & S.Mehrotra, "Domain-Independent Data Cleaning via Analysis of Entity-Relationship Graph." ACM Trans. Database Systems, vol. 31, no. 2, pp. 716-767, 2006.
[5]. M. Weis and F. Naumann, "Dogmatix Tracks Down Duplicates in XML," Proc. ACM SIGMOD Conf. Management of Data, pp. 431-442, 2005.
[6]. L. Leita˜o, P. Calado, and M. Weis, "Structure-Based Inference of XML Similarity for Fuzzy Duplicate Detection," Proc. 16th ACM Int'l Conf. Information and Knowledge Management, pp. 293-302, 2007.
[7]. A.M. Kade and C.A. Heuser, "Matching XML Documents in Highly Dynamic Applications," Proc. ACM Symp. Document Eng. (DocEng), pp. 191-198, 2008.
[8]. D. Milano, M. Scannapieco, and T. Catarci, "Structure Aware XML Object Identification," Proc. VLDB Workshop Clean Databases (CleanDB), 2006.
[9]. P. Calado, M. Herschel, and L. Leita˜o, "An Overview of XML Duplicate Detection Algorithms," Soft Computing in XML Data Management, Studies in Fuzziness and Soft Computing, vol. 255, pp. 193-224, 2010.
[10]. S. Puhlmann, M. Weis, and F. Naumann, "XML Duplicate Detection Using Sorted Neighborhoods," Proc. Conf. Extending Database Technology (EDBT), pp. 773-791, 2006.
[11]. S. Guha, H.V. Jagadish, N. Koudas, D. Srivastava, and T. Yu, "Approximate XML Joins," Proc. ACM SIGMOD Conf. Management of Data, 2002.
[12]. J.C.P. Carvalho and A.S. da Silva, "Finding Similar Identities among Objects from Multiple Web Sources," Proc. CIKM Workshop Web Information and Data Management (WIDM), pp. 90-93, 2003.
[13]. R.A. Baeza-Yates and B. Ribeiro-Neto, Modern Information Retrieval. Addison-Wesley Longman Publishing Co., Inc., 1999.
[14]. M.A. Herna´ndez and S.J. Stolfo, "The Merge/Purge Problem for Large Databases," Proc. ACM SIGMOD Conf. Management of Data, pp. 127-138, 1995.
[15]. J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, second ed. Morgan Kaufmann Publishers, 1988.
[16]. L. Leita˜o and P. Calado, "Duplicate Detection through Structure Optimization," Proc. 20th ACM Int'l Conf. Information and Knowledge Management, pp. 443-452, 2011.