Detection of Similar Identities in XML Documents

Amita Fulsundar, Dr. K.V.Metre

Abstract— **Duplicate detection is an important part of data cleaning; it is the process of detecting multiple representations of a same real-world object in the data sources. Numbers of solutions are available for detecting duplicates in XML data. One of the novel methods for XML duplicate detection is XMLDup. XMLDup makes use of a Bayesian network to evaluate the probability of two XML elements are duplicates. In addition a network pruning strategy is also used for improving the evaluation of the Bayesian network. A DOM tree construction algorithm for constructing the tree of the input XML data is proposed. It is seen that by using DOM tree construction algorithm higher efficiency is achieved for detection of similar identities in XML Documents**.

Index Terms—**Duplicate Detection, XML, DOM, Bayesian network,
data cleaning.**

I. Introduction

XML is extensively used as a data exchange standard between different networks and it is also used for uploading data on internet. It has an ability to represent data from wide variety of sources. XML data have vital advantage over a relational database, as it acts as a communication medium between different applications. Most of the time XML documents from different sources may contain errors and inconsistencies. It is important to ensure quality of data which is uploaded on web [2]. But quality of data can be compromised due to introduction of different types of errors like fuzzy duplicates. Duplicates are multiple representations of same real world entities. Errors are introduced due to typos, misspelling and different representation format. Identification of duplicates has become important task as duplicates are not exactly equal.
Data may be represented in various formats. It is essential to use a proper matching strategy to determine if they refer to the same real world entity or not.
In relational structure the tuples comparison are done and their similarity values are evaluated based on their attribute values. Methods used for duplicate detection in a single relation are not applicable to XML data, due to differences between the two data models. The hierarchical structures in XML provide useful additional information that helps in improving quality of duplicate detection.
Consider the two XML elements shown as Tree U and Tree U' in Fig. 1. Both represents publication details and labeled as pub. These elements have two attributes, namely id and title. They have further XML elements representing venue (ven) and author (aut).The XML element venue have further children XML elements as venue name (vname) and volume (vol). The author has further children XML elements as author id (id) and author name (name). Leaf elements have a text node which stores the actual data.

Manuscript received May 25, 2015
Miss Amita Fulsundar, Computer Department, MET BKCIOE Savitribai Phule Pune University, Nasik, India,
Dr.K.V.Metre, Computer Department, MET BKCIOE Savitribai Phule Pune University, Nasik, India,

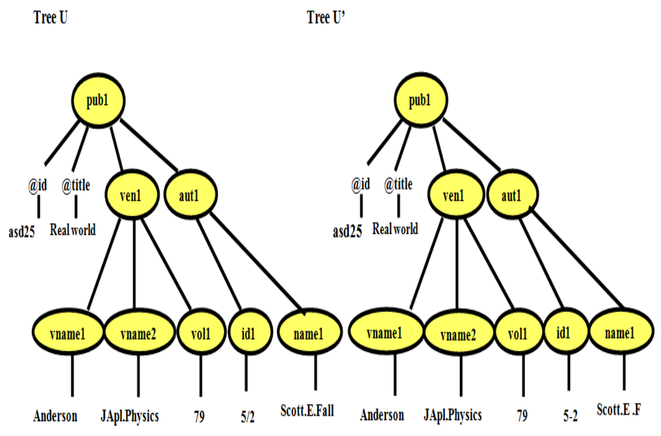


Fig 1: XML elements that represent the same publication information.

In the given example, the motivation of duplicate detection algorithm is to detect that both publication details are duplicates, despite the differences in the data. For this text values are compared. There is hierarchical organization of XML data which is used for duplicate detection. Since child elements can be detected to be similar, this increases the probability of parent elements being similar, and so on in a top-down manner.

II. Literature Survey

Several other methods exist for the problem of duplicate detection. A method for eliminating similarities in dimensional tables in a data warehouse was developed, which was usually associated with hierarchies [3]. A hierarchy to develop a good quality, scalable duplicate elimination method was exploited, and computation on datasets from an operational data warehouse.
An algorithm to perform approximate joins in XML data was proposed by Guha et al. [4]. Actual focus was on how to

join two sets of similar elements in best manner. Thus, they focused on a well organized implementation of a tree edit distance, which could later be applied in an XML join algorithm.

Dogmatix framework which focuses at both efficiency and effectiveness in duplicate detection was proposed by M.Weis et.al [6]. The framework consists of three main tasks. 1) candidate definition; 2) duplicate definition, which provides the definitions necessary for duplicate detection and 3) duplicate detection, which consists of the actual algorithm, an extension to XML data[3]. In this framework, XML elements based not only on their direct data values, but also on the similarity of their parents, children, structure, etc.

A probabilistic duplicate detection algorithm for hierarchical data known as XMLDup was presented [5]. This algorithm used Bayesian network for calculating probabilities and network pruning algorithm for improving the Bayesian network evaluation time[1][8].

A distance measure between two XML object representations which is based on the concept of overlays is proposed by Milano et al. [8]. An overlay between two XML trees U and V is a mapping between their nodes, such that a node u€U is mapped to a single node v € V if, and only if, they have the same path from the root. This measure is then used to make a pairwise comparison between all candidates. If the distance measure evaluates that two XML candidates are closer than a given threshold, the pair is detected as a duplicate.

SXNM (Sorted XML Neighborhood Method) is a duplicate detection approach that adapts the idea of relational sorted neighborhood method (SNM)[10] 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.

A new method for fuzzy duplicate detection in hierarchical and semi-structured XML data was proposed [6]. It not only considers the duplicate status of children, but rather the probability of children being duplicates. Probabilities are computed efficiently using a Bayesian network. Experiments show the algorithm is able to maintain high precision and recall values.

III. PROPOSED SYSTEM

For constructing tree of input XML data, a DOM Tree Construction algorithm is proposed. For hierarchical data, a probabilistic duplicate detection algorithm known as XMLDup is used. For duplicate detection, a Bayesian Network model is constructed, and this model is used to compute the similarity between XML object representations. Given this similarity, two XML objects are as duplicates if it is above a predefined [7].

A major result in that XMLDup exceed the performance of existing algorithm more efficiently for XML duplicate detection. A schema mapping step has preceded duplicate detection, so that all XML elements compared comply with the same schema. It provides a more extensive evaluation of algorithms than in previous work. A distance measure between two XML object representations that is defined based on the concept of overlays is more effective algorithm.

A. DOM Tree Construction

For XML documents, an application programming interface is Document Object Model (DOM). It gives the logical structure of documents. It defines the way a document is accessed. The term ”document” is used in the wide sense, XML is a way of representing various types of information that may be stored in different systems, and much of this would be viewed as data rather than as documents. XML represents this data as documents, and the DOM manage this data.

DOM Tree construction algorithm is used to construct DOM tree of the input XML data. Algorithm: DOMTreeConstruction(X, P)

```
Input: X is set of XML elements in XML document. P is tag of the roots of the XML document
Output: DOM Tree.
1) Initialize single node in XML document.
2) Check whether that node have child node.
3) Get all child nodes of the node.
4) for each node from the xml node list.
5) Get child node of the node.
6) Get or sets concatenated values of the node and its entire child node.
7) DOMTreeConstruction(X, P)
8) Returns DOM tree
```

B. Construction of Bayesian Network

An XMLDup approach for XML duplicate detection is based on one basic assumption: The fact that two XML nodes are duplicates depends only on the fact that their values are duplicates and that their children nodes are duplicates. Thus, we say that two XML trees are duplicates if their root nodes are duplicates. To illustrate this idea, consider the goal of detecting that both publications represented in Fig.1.are duplicates. This means that the two publication objects, represented by nodes tagged pub, are duplicates depending on whether or not their children nodes (tagged ven and aut) and their values for attributes year and title are duplicates. The nodes tagged aut and ven are duplicates depending on whether or not their children nodes are duplicates. This process goes on recursively until the leaf nodes are reached. If we consider trees U and U’ of Fig.1, this process can be represented by the Bayesian Network of Fig.2. Let us first consider the XML nodes tagged pub. As illustrated in Fig.2, the BN will have a node labeled pub11 representing the possibility of node pub1 in the XML tree U being a duplicate of node pub1 in the XML tree U’. Node pub11 is assigned a binary random variable. This variable takes the value1 (active) to represent the fact that the XML pub nodes in trees U and U’ are duplicates. It takes the value 0 (inactive) to represent the fact that the nodes are not duplicates. In accord with assumption, the probability of the two XML nodes being duplicates depends on 1) whether or not their values are duplicates, and 2) whether or not their children are duplicates. Thus, node pub11 in the BN has two parent nodes, as shown in Fig.2. Node Vpub11 represents the possibility of the values in the pub nodes being duplicates.

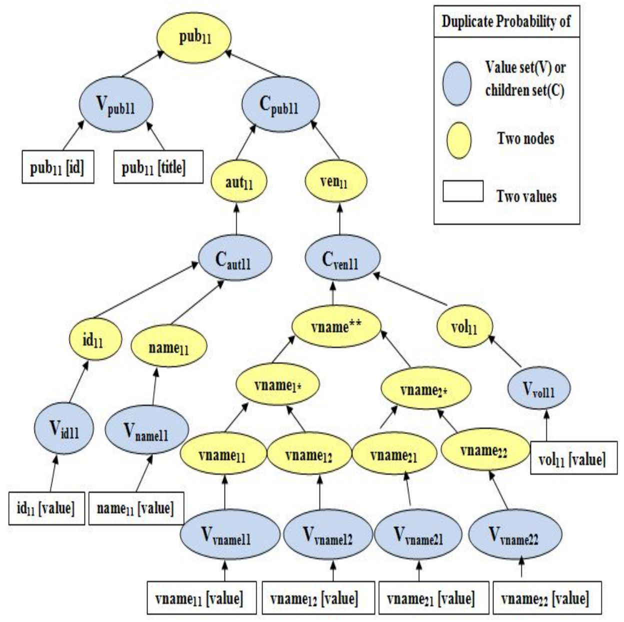


Fig 2: Bayesian Network (BN).

Node Cpub11 represents the possibility of the children of the pub nodes being duplicates. As before, a binary random variable, that can be active or inactive, is assigned to these nodes, representing the fact that the values and children nodes are duplicates or non duplicates, respectively. It is assumed that the probability of the XML node values being duplicates depends on each attribute independently. This is represented in the network by adding new nodes for the attributes as parents of node Vpub11, represented as rectangles in Fig.2. In this case, these new nodes represent the possibility of the year values in the pub nodes being duplicates and of the title values in the pub nodes being duplicates. Similarly, the probability of the children of the pub nodes being duplicates depends on the probability of each pair of children nodes being duplicates. Thus, two more nodes are added as parents of node Cpub11: node aut11 represents the possibility of node aut1 in tree U being a duplicate of the node aut1 in tree U'; node ven11 represents the possibility of node ven1 in tree U being a duplicate of node ven1 in tree U'. Whole process is repeated for these two nodes. However, a slightly different procedure is taken when representing multiple nodes of the same type, as is the cases for the XML nodes labeled vname. In this case, the full set of nodes is compared, instead of each node independently. Thus, the set of act nodes being duplicate depends on each vname node in tree U being a duplicate of any ven node in tree U'. This is represented by nodes ven, ven1, and ven2 in the BN of Fig.2.This approach is not symmetrical, i.e., the network obtained for U and U' is not always the same as that obtained for U' and U. However, it would be difficult to satisfy this feature without a significant decrease in efficiency, while we would not expect a high increase in effectiveness.

C. Probabilities calculation

As a binary random variable is assigned to each node, which takes the value 1 to represent the fact that the corresponding data in trees U and U' are duplicates, and the value 0 to represent the opposite. Thus, to decide if two XML trees are duplicates, the algorithm has to compute the probability of the root nodes being duplicates.

1) Prior Probabilities: In the network of Fig.2, we need to define the prior probabilities of values being duplicates in the context of their parent XML node, i.e., P(pub11[id]) P(pub11[title]) P(vol11[value]) and P(vnameij[value]) . These probabilities can be defined based on a similarity function sim( ) between the values, normalized to fit between 0 and 1. However, it is sometimes not possible, or not efficient, to measure the similarity between two attribute values. In this case, we define the probability as a small constant ka, named the default probability, representing the possibility of any two values being duplicates before we observe them. Thus, we define P (tij [a]) = sim (vi[a], vj [a]) if the similarity was measured, and P (tij [a]) =ka if otherwise, where Vi[a] is the value of attribute a of the ith node with tag t in the XML tree. For instance, for the name attribute in the pub nodes, we can define sim (n1, n2) = 1-ned (n1, n2), where ned( ) is the normalized edit distance between the strings. The default probability ka can be derived from the distribution of attribute values in the database, or simply be set to a small number.

2) Conditional Probabilities:

- CP1:

The probability of the values of the nodes is duplicates, given that each single pair of values contains duplicates.

- CP2:

The probability of the children nodes is duplicates, given that each single pair of children are duplicates. The more child nodes in both trees are duplicates, the higher the probability that the parent nodes are duplicates.

- CP3:

The probability of two nodes is duplicates given that their values and their children are duplicates.

- CP4:

The probability of a set of nodes of the same type being duplicates given that each pair of individual nodes in the set are duplicates.

D. BN Evaluation Algorithm

In order to improve the BN evaluation time, a lossless pruning strategy is used. 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. Network evaluation is performed by doing a propagation of the prior probabilities, in a bottom up manner, 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 pruning technique lies in avoiding the calculation of prior probabilities, unless they are strictly necessary. The strategy makes assumption that, before comparing two objects, all the similarities are assumed to be 1.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 evaluated, 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. The process is presented in detail in Algorithm1.

```
Algorithm 1. NetworkPruning(N,T)
Input: N-node which we intend to compute the probability
score; T- threshold value below which the XML nodes are
considered non-duplicates
Output: Probability score of the XML nodes represented by
N
1) Get the ordered list of parents
2) Set Maximum probability of each parent node as 1
3) currentScore 0
4) for each node n in List do Compute the duplicate
probability
5) if n is a value node then
6) For value nodes, compute the similarities
7) else
8) Get new threshold
9) NetworkPruning(n, newThreshold)
10) end if
11) parentScore[n]   score
12) currentScore   computeProbability(parentScore)
13) if currentScore < T then
14) End network evaluation
15) end if
16) end for
17) return currentScore
```

The algorithm takes as input a node N from the BN and a user defined threshold T. It starts by gathering a list of all the parent nodes of N and assuming that their duplicate probability score is 1 .It then proceeds to compute the actual probability value of each of the parents of N. If a given parent node n is a value node, its probability score is simply the similarity of the values it represents. If, on the other hand, n also has parent nodes, its probability score depends on the scope of its own parents, which is computed recursively .However, the algorithm should now be called with a different threshold value that depends on the equation used to combine the probabilities for node N. Once the score for node n is computed, the algorithm check if the total score for N is still above T, and decides whether to continue computing or to stop the network evaluation.

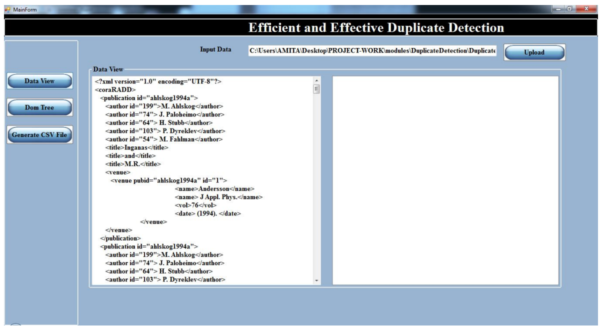
IV. Current State of Result Analysis

The Proposed algorithm known as DOM Tree Construction algorithm is applied to XML input dataset. Here we used Cora dataset for evaluation.

Modules for Implementation:-

- DOM Tree Construction
- Bayesian Network Construction
- Probabilities Calculation
- Network Pruning Algorithm

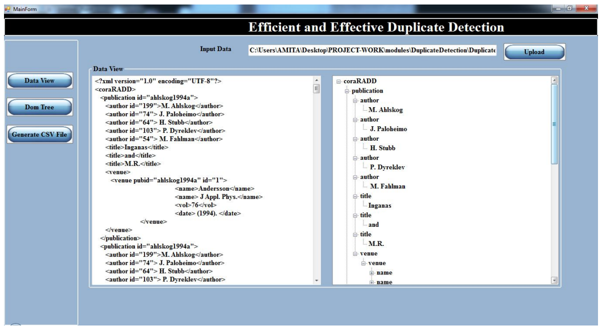
The following are the screenshot of how input XML dataset is converted into DOM tree.



C:\Users\AMITA\Desktop\PROJECT-WORK\modules\DuplicateDetection\Duplicate

```
<?xml version="1.0" encoding="UTF-8"?>
<coraRADD>
  <publication id="ahlskog1994a">
    <author id="199">M. Ahlskog</author>
    <author id="74"> J. Paloheimo</author>
    <author id="64"> H. Stubb</author>
    <author id="103"> P. Dyreklev</author>
    <author id="54"> M. Fahlman</author>
    <title>Inganas</title>
    <title>and</title>
    <title>M.R.</title>
    <venue>
      <venue pubid="ahlskog1994a" id="1">
                          <name>Andersson</name>
                          <name> J Appl. Phys.</name>
                          <vol>76</vol>
                          <date> (1994). </date>
              </venue>
    </venue>
  </publication>
  <publication id="ahlskog1994a">
    <author id="199">M. Ahlskog</author>
    <author id="74"> J. Paloheimo</author>
    <author id="64"> H. Stubb</author>
    <author id="103"> P. Dyreklev</author>
```

Fig 3: XML Data View of the uploaded XML input



C:\Users\AMITA\Desktop\PROJECT-WORK\modules\DuplicateDetection\Duplicate

```
<?xml version="1.0" encoding="UTF-8"?>
<coraRADD>
  <publication id="ahlskog1994a">
    <author id="199">M. Ahlskog</author>
    <author id="74"> J. Paloheimo</author>
    <author id="64"> H. Stubb</author>
    <author id="103"> P. Dyreklev</author>
    <author id="54"> M. Fahlman</author>
    <title>Inganas</title>
    <title>and</title>
    <title>M.R.</title>
    <venue>
      <venue pubid="ahlskog1994a" id="1">
                          <name>Andersson</name>
                          <name> J Appl. Phys.</name>
                          <vol>76</vol>
                          <date> (1994). </date>
              </venue>
    </venue>
  </publication>
  <publication id="ahlskog1994a">
    <author id="199">M. Ahlskog</author>
    <author id="74"> J. Paloheimo</author>
    <author id="64"> H. Stubb</author>
    <author id="103"> P. Dyreklev</author>
```

```
coraRADD
  publication
    author
      M. Ahlskog
    author
      J. Paloheimo
    author
      H. Stubb
    author
      P. Dyreklev
    author
      M. Fahlman
    title
      Inganas
    title
      and
    title
      M.R.
    venue
      venue
        name
        name
```

Fig 4: DOM tree of XML data

V. Conclusion

A DOM tree construction algorithm is proposed to construct the tree from XML data. A novel method known as XMLDup is used for XML duplicate detection. Bayesian network model is used for the comparison of the objects. For improving runtime efficiency a network pruning strategy is used. Estimating the probability using machine learning increases the rate of duplicate detection. The proposed method achieves an improvement in the value of precision on different structured data.

Acknowledgment

The authors wish to thank MET Institute of Engineering Bhujbal knowledge city Nashik, India for providing lab facilities. Authors are also thankful to Luis Leita o, Pavel Calado, and Melanie Herschel, “Efficient and Effective Duplicate Detection in Hierachical data,” for their work in this area. The mentioned paper is major guideline in this work.

.

References

[1] Luis Leita o, Pavel Calado, and Melanie Herschel, ”Efficient and Effective Duplicate Detection in Hierarchical data”,IEEE Transactions on Knowledge and Data Engineering, VOL. 25, NO. 5, MAY 2013.

[2] E. Rahm and H. H. Do, ”Data cleaning: Problems and current approaches,”
IEEE Data Engineering Bulletin, vol. 23, pp. 3-13, 2000.

[3] 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.

[4] S. Guha, H. V. Jagadish, N. Koudas, D. Srivastava, and T. Yu" Approximate XML joins", in Conference on the Management of Data (SIGMOD), 2002.

[5] R. Ananthakrishna, S. Chaudhuri, and V. Ganti, "Eliminating fuzzy duplicates in data warehouses", in Conference on Very Large Databases (VLDB),Hong Kong, China, 2002, pp. 586-597.

[6] D. Milano, M. Scannapieco, and T. Catarci, "Structure aware XML object identification", in VLDB Workshop on Clean Databases (CleanDB),Seoul, Korea, 2006.

[7] M. Weis and F. Naumann, "Dogmatix tracks down duplicatesin XML", in Conference on the Management of Data (SIGMOD),Baltimore, MD, 2005, pp. 431-442.

[8] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of plausible inference, ed. Morgan Kaufmann Publishers, 1988.

[9] 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.

[10] A. M. Kade and C. A. Heuser, "Matching XML documents in highly dynamic applications", ACM Symposium on Document Engineering (DocEng),2008, pp. 191-198.

[11] S. Puhlma nn, M. Weis, and F. Naumann, "XML Duplicate Detection Using Sorted Neighborhoods", Proc. Conf. Extending Database Technology (EDBT),pp. 773-791, 2006.