XEdge: Clustering Homogeneous and Heterogeneous XML Documents Using Edge Summaries

Panagiotis Antonellis Christos Makris Nikos Tsirakis
Computer Engineering and Informatics Department, University of Patras, Greece
Rio, Patras
{adonel, makri, tsirakis}@ceid.upatras.gr

ABSTRACT

In this paper we propose a unified clustering algorithm for both homogeneous and heterogeneous XML documents. Depending on the type of the XML documents, the proposed algorithm modifies its distance metric in order to properly adapt to the special structural characteristics of homogeneous and heterogeneous XML documents. We compare the quality of the formed clusters with those of one of the latest XML clustering algorithms and show that our algorithm outperforms it in the case of both homogeneous and heterogeneous XML documents.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval – *clustering, search process.*

General Terms

Algorithms

Keywords

XML, data mining, clustering, querying

1. INTRODUCTION

Data Clustering [6] is a challenging field in the area of data management. There are various forms of clustering that are required in a wide range of applications. Moreover there are many different types of data that can be clustered. Lately data written in a more sophisticated markup language such as XML have made great strides in many domains. Processing and management of XML documents have already become popular research issues [1], with the main problem in this area being the need to optimally index them for storage and retrieval purposes. For this reason we need to know how to identify related data. Clustering of XML documents brings new challenges, since an XML file encodes not only data but also structure, in one entity and this affects the efficient application of traditional clustering algorithms in order to detect groups of XML documents that share similar characteristics. The estimation of similarity is closely related to the distance metric exploited by the clustering algorithm.

In the literature, trees are commonly used for modeling XML documents without reference elements and the structural similarity between a pair of XML documents can be defined as some edit distance between the corresponding labeled trees. However, in XML, a strong hierarchical relationship exists between the elements of the documents. Documents are considered similar for not only having elements similar in concepts, but also for having similar structural relationships between them. A few methods of XML clustering do not consider semantic measures to form the grouping. In the following section we highlight some notable methods.

1.1 Background and Related Work

There have appeared in the literature many research works that attempted to solve the XML clustering problem by proposing several techniques and algorithms. Generally the clustering of XML documents as a problem has two dimensions: content and structure. The content dimension needs distances that estimate similarity in terms of the textual content inside elements, while the structure dimension needs distances that estimate similarity in terms of the structural relationships of the elements. Taken these two dimensions into consideration, the authors in [9] address the problem of clustering XML data according to structure and content features enriched with lexical ontology knowledge. Also Ma & Chbeir [7] have studied the similarity between XML-Based data and have proposed an approach able to consider both the data structure and the content based on the same schema and by using a prototype to validate and evaluate their results. In [3] the authors proposed a methology for clustering XML documents on the basis of their structural similarities which is based on the notion of XML cluster representatives. In this way they exploited the tree nature of XML documents and provided techniques for tree matching, merging and pruning. Another work example of Dalamagas [4] explores the application of clustering methods for grouping structurally similar XML documents. By modeling the XML documents as rooted ordered labeled trees, they apply clustering algorithms using the tree-edit distance between these trees in terms of the hierarchical relationship of their nodes. Another interesting work in [5], deals with clustering homogeneous collections of text-centric XML documents. By using the classic k-means clustering algorithm they combine structural similarities and content similarity in order to improve the clustering quality. One of the most recent approaches is the work presented in [8]. The authors propose a compact level structure representation of each XML document based on node summaries per level of the XML document. Based on this representation, they define an appropriate distance metric for heterogeneous XML documents and they apply a hierarchical clustering algorithm in the set of level structure representations of the documents.

The main disadvantage of the previously described methods is the lack of a unified clustering framework that can be applied efficiently both in *homogeneous* and *heterogeneous* XML documents. Heterogeneous XML documents are derived from different Document Type Definitions (DTDs) and represent semantically different information. Such XML documents in most cases do not share the same node tags, as they belong to different semantic categories. However, in some cases such as in a movie DTD and in an actor DTD, the XML documents may share some node tags but can contain different parent/child relationships between such nodes. On the other hand, homogeneous XML documents are usually derived from sub-DTD's of the same DTD. Those documents usually share the same set of nodes per level and their differences lie in the presence or absence of certain edges between nodes of consecutive levels. Thus, due to different structural characteristics of homogeneous and heterogeneous XML documents, a unified clustering framework should distinguish between them and treat them properly, in order to improve its clustering results.

1.2 Paper Motivation and Contribution

As has already been mentioned, in order to apply a clustering process in a set of XML documents, it is vital to define a suitable representation of an XML document and a corresponding distance metric between the representations of a pair of XML documents. Some of the previously described clustering methods either utilize a complex and difficult to calculate distance metric [4] or they propose a distance metric which misses important structural or semantic information from the XML documents [5], [8]. Additionally, to our knowledge, none of the existing XML document representations and distance metrics distinguishes between homogeneous and heterogeneous XML documents. However, in order to achieve optimal results, a clustering framework for XML documents should treat homogeneous XML documents in a different way than heterogeneous XML documents due to their different characteristics as explained previously.

In this paper we extend and generalize the clustering process proposed in [8]. The proposed level structure representation in that paper, although compact and relatively small in size, ignores important structural information such as the relationships between the nodes of every level. This may result in totally erroneous clustering results in case of homogeneous XML documents sharing the same set of node tags or in the case of heterogeneous XML documents sharing a subset of same node tags. Finally, the proposed metric is mainly addressed for heterogeneous XML documents. However, the distance metric should also be considered and properly revised for the case of homogeneous XML documents, which usually requires a different approach. Regarding the clustering process, the authors propose the usage of a hierarchical clustering algorithm, but they neither mention how a cluster representative is defined nor take into consideration that the popularity of real XML datasets grows rapidly resulting into exponentially increasing clustering time.

In order to address the above mentioned problems, we propose *LevelEdge*, a novel structured representation of XML documents based on edges summaries and an appropriate distance metric between two XML documents based on the *LevelEdge* representations of the XML documents.

The proposed structure representation summarizes per level the most important structure elements of an XML document: its edges, instead of nodes and it can be adapted appropriately in the case of homogeneous XML documents. Additionally, in order to provide a more efficient clustering process and results, we propose the utilization of a partitional clustering algorithm along with the definition of an appropriate cluster representative structure. XML documents are represented by their LevelEdge's and their distances are computed using the proposed distance metric.

Finally, we propose an indexing technique for the clustering results, in order to improve the efficiency of existing XML querying algorithms over a set of heterogeneous/homogeneous XML documents.

The contribution of our paper can be summarized as follows:

- A compact representation of XML documents is proposed that preserves most of the structural information of an XML document by summarizing the distinct edges for each level of the XML document.
- A distinction is made between homogeneous and heterogeneous XML documents by properly adjusting the proposed distance metric.
- Efficient clustering of a set of XML documents is performed using a partitional clustering algorithm.
- Every cluster is represented by a compact cluster representative structure that summarizes the properties and characteristics of the XML documents included in the appropriate cluster.
- An efficient utilization of the clustering results for boosting the querying process over the set of the XML documents.

The rest of the paper is structured as follows: Section 2 introduces the LevelEdge structure of an XML document and describes the corresponding distance metric; section 3 discusses analytically the clustering process; section 4 discusses the experimental results; section 5 presents our conclusions and section 6 lists our references.

2. LevelEdge Structure and Distance Metric

2.1 LevelStructure and its Drawback

The authors in [8] introduced the *LevelStructure*: a compact structured summary of an XML document. The LevelStructure groups the distinct XML nodes for each level in the document, thus it is organized as a vector of levels, where every level contains a list of distinct XML nodes. An example of an XML document and its LevelStructure is presented in Figure 1(a) and Figure 1(b) respectively. The integer number below a node's tag in the Figure 1(a) is its corresponding integer encoding. Those integers are used for representing the nodes in the LevelStructure of Figure 1(b). Although compact and relatively small in size, the LevelStructure has a main drawback which can result in totally erroneous results during the clustering process of a set of XML documents represented by their LevelStructrues: it misses information about the structural relationships (parent/child, ancestor/descendant) between nodes. The only structural information contained in LevelStructure is which nodes are presented in each level of the XML document, while the relationships between nodes of different levels are missing. Thus, it is possible that the same LevelStructure corresponds to two structurally-different XML documents, as shown in Figure 2. Figure 2 depicts an example of two XML documents containing the same distinct nodes in each level, but the parent/child relationships between nodes are different in each document. Howbeit, the two XML documents are summarized by the same LevelStructure. This case is very common in cases of

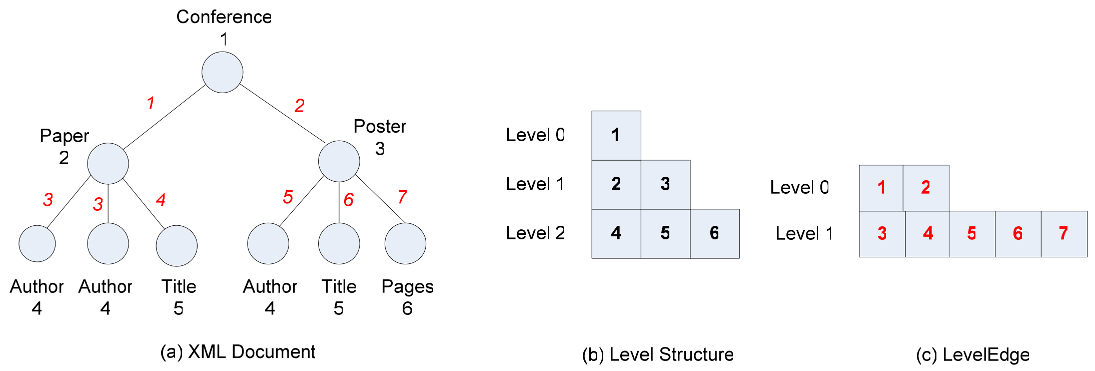


Figure 1. Example of Level Structure and LevelEdge

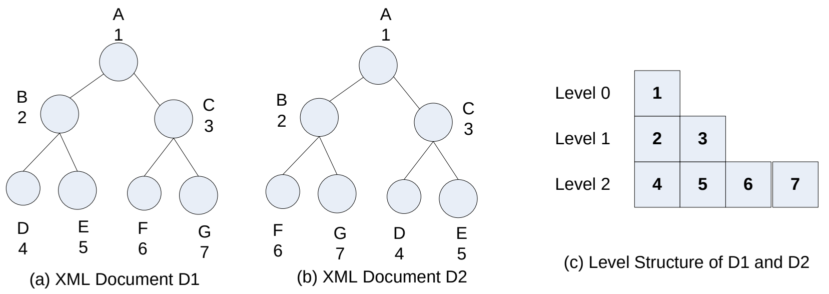


Figure 2. Example of two different XML documents represented by the same LevelStructure

homogeneous XML documents, derived from sub DTDs of the same DTD, or in cases of heterogeneous XML documents sharing the same node tags but utilizing different structural relationships between their nodes.

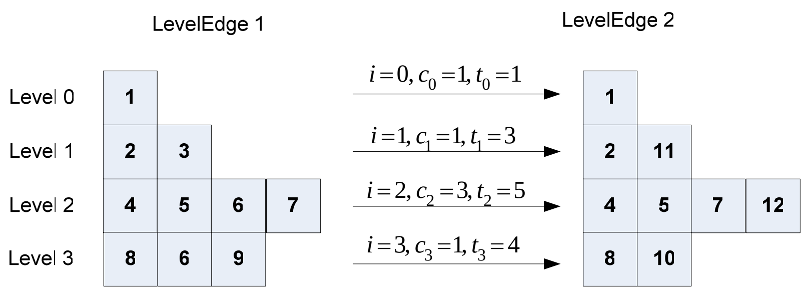
As a result, the LevelStructure representation of an XML document is insufficient for being utilized in a clustering process of either heterogeneous or homogeneous XML documents.

2.2 LevelEdge: Summarizing Edges per Level

In order to address the problems of LevelStructure representation, we propose a unique structure representation of XML documents called *LevelEdge* which is based on edge summaries.

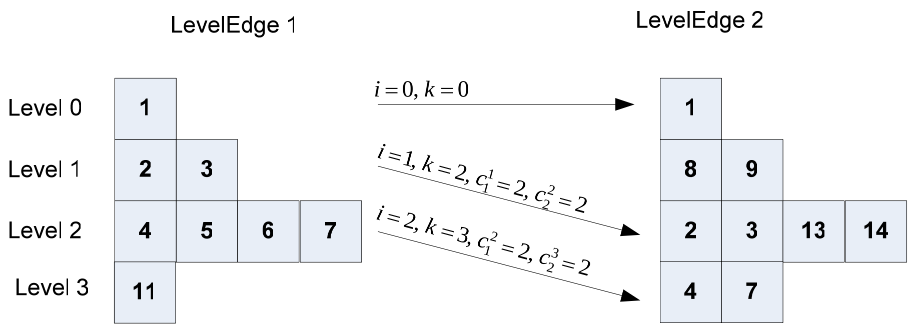
The LevelEdge structure groups the distinct edges for each level in the XML document. It is organized as a vector of levels, where each level contains a list of distinct edges. Each distinct edge is uniquely defined by its two distinct point-nodes. The distinct edges are first encoded as integers and those integers are used in order to construct the LevelEdge representation of an XML document. Figure 1(c) presents the LevelEdge representation of the XML document in Figure 1(a). The integer numbers in the side of each edge in the XML document are the encodings of the corresponding edges. For example, all the *Paper-Author* edges are encoded as *3*, while the *Poster-Author* edge is encoded as *5*. As we can see in Figure 1(c), the LevelEdge structure consists of only two levels, as no edges begin from level 3. The Level0 contains the edges *1, 2* which correspond to the outcoming edges of the *Conference* node, while the Level1 contains the edges *3, 4, 5, 6* and *7* which correspond to the outcoming edges of the nodes in the level 1 of the XML document.

The main advantage of the LevelEdge representation of an XML document in relation with the LevelStructure representation is the preservation of the structural relationships between nodes of consecutive levels of the XML documents in the form of edges. Each edge represents a parent/child relationship between the nodes corresponding to its two points. Thus, the LevelEdge representation summarizes all distinct parent/child relationships in each level of the XML document, instead of simply summarizing the distinct nodes as LevelStructure does. As a result, in all cases of real XML documents, either heterogeneous or homogeneous, the summarized edge information for each level is enough in order to distinguish between semantically and structurally different XML documents. Consider for example a set of heterogeneous XML documents derived from different DTDs which do not share the same node tags. In such a case both the LevelEdge and LevelStructure representations can be utilized in order to distinguish between two different XML documents. However, there is a possibility that some of the documents in the set share a subset of the same node tags but contain different parent/child relationships between such nodes. In such a case, the LevelEdge representations of two such documents will be very different because they encode edges, not nodes. On the other hand, the corresponding LevelStructure representations of the two documents may be similar enough to be considered as homogeneous XML documents. Regarding a set of homogeneous XML documents the LevelStructure representation totally fails to distinguish between two XML documents derived from different sub-DTDs as they share the same set of node tags in each level. On the contrary, the LevelEdge structure can distinguish between two such documents, as they usually differ in the absence or existence of extra edges in some levels (e.g. optional attributes and/or optional children nodes). This important property of the LevelEdge representation can be utilized in order to define an appropriate distance metric between two LevelEdge representations which in turn can be adopted for applying a clustering process over a set of heterogeneous or homogeneous XML documents.



$$Sim_{L_1,L_2} = \frac{1\times 2^3 + 1\times 2^2 + 3\times 2^1 + 1\times 2^0}{1\times 2^3 + 3\times 2^2 + 5\times 2^1 + 4\times 2^0} = 0,41$$

Figure 3. Example of similarity measure between homogeneous XML documents



$$Sim_{L_1,L_2} = \frac{0,5\times(1\times 2^3+2\times 2^2+2\times 2^1+0\times 2^0)+0,5\times\left(1\times 2^3+0\times 2^2+2\times 2^1+2\times 2^0\right)}{1\times 2^3+4\times 2^2+8\times 2^1+3\times 2^0}$$

Figure 4. Example of similarity measure between heterogeneous XML documents

2.3 Distance metric between LevelEdge representations

In order to apply a clustering process over a set of XML documents represented by their LevelEdge's, we need to define an appropriate distance metric between two LevelEdge representations. This distance metric should approximate the real semantic and structural distance between the two XML documents. Finally, as explained previously, the distance metric should distinguish between homogeneous and heterogeneous XML documents and adapt itself accordingly.

Based on the above notes, we first propose a similarity measure between two LevelEdge representations and then we define the corresponding distance metric based on the proposed similarity measure. Consider two LevelEdge representations L_1 and L_2 with N_1 and N_2 levels respectively and a positive integer a >0. Let $m = \min(N_1, N_2)$ and $M = \max(N_1, N_2)$ .Then, we define the similarity measure between L_1 and L_2 as follows:

$$Sim_{L_1,L_2} = \frac{\sum_{i=0}^{m-1} c_i \times a^{m-i-1}}{\sum_{j=0}^{M-1} t_j \times a^{M-j-1}} \qquad (1)$$

, where c_i denotes the number of common distinct edges in the level *i* of L_1 and L_2 , while t_j denotes the total number of distinct edges in the level *j* of both L_1 and L_2 . The integer *a* is a user-defined weight factor which is used to indicate that higher level edges are semantically more important in XML documents. The denominator in Equation 1 denotes the total weight of all distinct edges in both trees, while the numerator denotes the total weight of the common edges in levels of L_1 and L_2 . Thus, the more common edges in high levels the two LevelEdge representations have, the more similar they are. The intuition behind weighting the number of common edges accordingly to the corresponding level is that edges in higher levels of the XML document contribute in the semantics of the XML document more than edges in lower levels.

The proposed similarity value varies between 0 and 1; 0 indicates completely structural-different XML documents and 1 indicates structural-similar XML documents. The previously defined similarity measure is used *as-is* in the case of homogeneous XML documents. As explained before, homogeneous XML documents are derived from sub-DTDs of the same DTD, so we except that common edges occur in the same levels in both documents. That's why the proposed similarity measure implies a strict matching procedure, where common edges are searched only in the same level of both the XML documents. Figure 3 depicts an example of calculating the similarity between two LevelEdge representations of homogeneous XML documents, using a weight factor of 2.

However, two heterogeneous XML documents may share common edges in different levels of their structure, e.g. a document 1 may have an edge *e1* at level *i*, while a document 2 may have an edge *e1* at level *j*. Such matchings at different levels are the most common cases in heterogeneous XML documents, due to different DTDs. Thus the similarity measure should be modified accordingly in order to loose the matching procedure's criteria. Thus, instead of checking the common edges only in the same level of the two documents, we loose this constraint and we permit matching common edges in different level in each document. That way, we are able to "catch" partial matchings between two documents, even in different levels of each document. We first give the definition of the new similarity measure between two heterogeneous XML documents and then we describe the exact matching procedure for calculating the similarity measure.

The similarity measure between two LevelEdge representations L_1 and L_2 of heterogeneous XML documents is defined as follows:

$$Sim_{L_1,L_2} = \frac{0,5 \times \sum_{i=0}^{L_1-1} c_i^1 \times a^{L_1-i-1} + 0,5 \times \sum_{k=0}^{L_2-1} c_k^2 \times a^{L_2-k-1}}{\sum_{j=0}^{M-1} t_j \times a^{M-j-1}} \quad (2),$$

where c_i^1 denotes the number of common edges found between the level i of L_1 and some level of L_2 during the matching procedure, c_k^2 denotes the number of common edges found between the level k of L_2 and some level of L_1 during the matching procedure. The semantics of the numerator and denominator in Equation 2 are the same with Equation 1. The only difference is the way of defining the common edges and calculating their corresponding weight.

The c_i^1 and c_k^2 values are defined by a matching procedure that tries to match each level *i* of L_1 with a level *j* of L_2 in such a way that the two corresponding levels contain at least one common edge and if the level *i-1* of L_1 has been matched with the level *k* of L_2, then the level *i* of L_1 can only be matched with a level *j* of L_2 if $j > k$.

The matching procedure based on Equation 2 is defined as follows:

1. Start searching for common edges in Level 0 of L_1 and L_2. If at least one common edge is found, set c_0^1 and c_0^2 equal with the number of common edges found and go to step 2. Otherwise go to step 3.
2. Move both L_1 and L_2 to the next level (increase *i*, *k* by 1) and search for common edges. If at least one common edge is found, set c_i^1 and c_k^2 equal to the number of common edges found and go to step 2. Otherwise go to step 3.
3. Only move L_2 to the next level (increase *k* by 1) and then search for common edges in the level *i* of L_1 and the new level *k* of L_2. If at least one common element is found, set c_i^1 and c_k^2 equal to the number of common edges found and go to step 2. Otherwise go to step 3.
4. Repeat until all the levels in either object have been checked.

Figure 4 depicts an example of calculating the similarity between two LevelEdge representations of heterogeneous XML documents, using a weight factor of 2. For example, during the matching procedure, no common edge was found at level 1 of L_1 and L_2. Thus, we proceeded at the next level (level 2) of L_2. At this point, we found two common edges (edge 2 and edge 3) between the level 1 of L_1 and the level 2 of L_2, resulting in $c_1^1 = 2$ and $c_2^2 = 2$.

Again, the proposed similarity's (Equation 2) value varies between 0 and 1; 0 indicates completely structural-different XML documents and 1 indicates structural-similar XML documents.

After defining the similarity between two LevelEdge representations of homogeneous or heterogeneous XML documents, we define the distance metric between two LevelEdge representations as:

$$Dis_{L_1,L_2} = 1 - Sim_{L_1,L_2} \quad (3)$$

The distance metric's value varies between 0 and 1; 0 indicates structural-similar XML documents and 1 indicates completely structural-different XML documents. The proposed distance metric will be utilized as the distance metric of the *XEdge* clustering algorithm, described in the next section.

3. XEdge Clustering Algorithm

With the increasing volume of information stored in XML documents, the number of XML documents grows rapidly. Thus, XML management systems should be able to deal and cluster efficiently a large number of XML documents. Based on this notion, we propose *XEdge*: a partitional clustering algorithm based on kMeans [6] as it incorporates the main advantages of kMeans:

- It is generic, as it can work for any distance desired and requires no training phase.
- Its speed is very appealing in practice, especially in the case of large numbers of items.

XEdge is a modified version of k-Means where each XML document is represented by its LevelEdge and which utilizes the previously described distance metric in order to calculate the distance between two LevelEdge representations. Additionally, instead of assigning random initial centroids to the clusters, the algorithm utilizes the method described in [2] to calculate the initial centroid for each cluster. Finally, for every cluster we define its *cluster representative*. A cluster representative is a LevelEdge representation that summarizes all the LevelEdge representations of the XML documents belonging to the

```
XEdge Algorithm
/*Input Parameters*/
Number of clusters: k
A set of m LeveEdge representations $L_1, L_2, \ldots, L_m$
/*Initialization Phase*/
(1)    Form k empty clusters
(2)    Calculate the initial centroid for each cluster utilizing e.g. the method described in Error! Reference source not found..
/*Main Phase*/
Repeat
(6)    Assign each LevelEdge representation to the cluster that has the minimum $Dis_{L_j, C_i}$;
(7)    When all LevelEdge representations have been assigned, recalculate new cluster representatives;
Until no cluster representative is changed
```

Figure 5. XEdge clustering algorithm's overview

corresponding cluster. More precisely, each level of the cluster representative contains all the distinct edges in that level of all the cluster's LevelEdge representations.

Figure 5 outlines the proposed clustering algorithm that consists of the initialization phase and the main phase. In the initialization phase, *k* clusters are formed and the initial centroid for each cluster is calculated based on the method described in [2]. Due to space limitations, we will not describe further the technique for calculating the initial centroids.

During the main phase, every LevelEdge representation is checked again each cluster and is assigned to the closest cluster. The distance between a LevelEdge representation L_j and a cluster C_i is defined as the distance between L_j and the cluster's representative. After assigning all the LevelEdge representations, the cluster representatives are recalculated. The main phase is repeated until no cluster representative is changed.

The output of the XEdge algorithm is a set of *k* clusters containing the input LevelEdge representations of the XML documents. The resulted clusters, except for providing a condensed overview of the grouping of XML documents, can be utilized in order to boost existing XML querying algorithms. As mentioned before, every cluster is well-defined by its cluster representative: a LevelEdge representation summarizing all the distinct edges per level of the XML documents belonging to the corresponding cluster. A naïve solution for boosting the processing of XML queries, is to build an index with key the integer code of an edge and value a list of tuples $< clNum, Level >$. The *clNum* of a tuple represents the number of the cluster which contains the edge-key and the *Level* represents the edge's level in this cluster's representative. If the corresponding edge is contained in more than one level in the cluster's representative, then the list contains one tuple for every edge's level on the corresponding cluster representative.

The resulted index can be utilized by an XML querying algorithm in order to boost the processing of XML queries, by reducing the query space. In this paper we consider simple path queries containing only parent/child relationships. Every such query can be split into a sequence of edges, starting from the root node of the query. Starting from the first edge, we query the formed index, gain the corresponding list of tuples and store it. In every step, we check the newly gained list of tuples with the stored tuples of the previous steps. For every new tuple <*cId_n, lev_n*>, we check whether there exists a stored tuple <*cId_s, lev_s*> with *cId_s* = *cId_n* and *lev_n* = *lev_s+1*. If such a stored tuple exists, then the corresponding cluster's representative contains the portion of the query comprised by the previous and current query edge. Thus, we store the corresponding new tuple, otherwise we discard it. After querying all the edges, we have a stored list of tuples. The set of cluster numbers in that list is the clusters that may contain XML documents that match the original query. This is true, as a cluster representative summarizes all the distinct edges per level of its XML documents. Thus, we apply the querying algorithm only in the XML documents of the corresponding clusters, reducing that way the querying space and boosting the processing of the XML query. This technique can be utilized with any existing XML querying algorithm as it can be considered as a preprocessing step of the querying algorithm.

4. Experimental Results

We compared XEdge against XCLS [8] in order to test the efficiency of our LevelEdge representation along with the utilization of the previously described distance metric. We chose XCLS because it is one of the most recently proposed XML clustering algorithms, it utilizes a technically similar clustering approach with XEdge and it is mainly used for heterogeneous XML documents. As mentioned before, XEdge is designed to work efficiently both in case of heterogeneous and homogeneous XML documents. In order to compare the efficiency and performance of XEdge with XCLS, we performed three different experiments.

In the first experiment, we used 323 real heterogeneous XML documents downloaded from the Wisconisn's XML data bank (www.cs.wisc.edu/niagara/data.html) and the XML data repository (www.cs.washington.edu/research/xmldatasets). The downloaded XML documents can be categorized in 7 different domains: Actors, Bibliography, Club, Company, Department, Movies and Sigmod Records. The main feature of this dataset is that most of the different domains utilize different node tags and edges/relationships, except of the Sigmod Record and the Bibliography domains that share the same node tags but use different edges/relationships between those shared node tags.

Table 1. Results of experiment 1

XML Domain	# Documents	XEdge			XCLS		
		Recall	Precision	F-measure	Recall	Precision	F-measure
Actors	75	1	1	1	1	1	1
Bibliography	16	1	1	1	1	1	1
Club	12	1	1	1	0	0	0
Company	20	1	1	1	1	1	1
Department	19	1	1	1	1	0,61	0,75
Sigmod	51	1	1	1	1	0,92	0,96
Movies	130	1	1	1	1	1	1

Table 2. Results of experiment 2

Sub-DTD	# Documents	XEdge			XCLS		
		Recall	Precision	F-measure	Recall	Precision	F-measure
Dblp_1	17	1	1	1	0	0	0
Dblp_2	16	1	1	1	0,21	1	0,35
Dblp_3	24	1	1	1	0,58	1	0,73
Dblp_4	110	1	1	1	1	0,47	0,64

Table 3. Results of experiment 3

Sub-DTD	# Documents	XEdge			XCLS		
		Recall	Precision	F-measure	Recall	Precision	F-measure
Movie_1	17	0,58	1	0,73	1	1	1
Movie_2	25	0,44	0,35	0,39	0,96	0,65	0,78
Movie_3	15	0,80	0,46	0,46	0,66	0,13	0,22

With this experiment we wanted to compare the efficiency of XEdge and XCLS in the case of heterogeneous XML documents.

In the second experiment, we used the dblp DTD (http://www.informatik.uni-trier.de/~ley/db/about/dblp.dtd) to create 4 sub-DTDs. We created those sub-DTDs in such a way that the derived synthetic XML documents would share the same node tags in each level, but contain some different edges/relationships in most of the levels. Thus, with this experiment we wanted to test the performance of XEdge and XCLS in the case of homogeneous XML documents sharing the same node tags but different edges. We created a total number of 167 homogeneous XML documents: 17 derived from the first sub-DTD, 16 derived from the second sub-DTD, 24 derived from the third sub-DTD and 110 derived from the fourth sub-DTD.

In the third experiment, we used the movies DTD (http://www.cs.wisc.edu/niagara/data.html) to create 3 sub-DTDs. We created those sub-DTDs in such a way that the derived synthetic XML documents would share the same node tags and edges/relationships in each level. Thus, with this experiment we wanted to test the performance of XEdge and XCLS in the case of homogeneous XML documents sharing the same node tags and edges. We created a total number of 57 homogeneous XML documents: 17 derived from the first sub-DTD, 15 derived from the second sub-DTD and 15 derived from the third sub-DTD.

In each experiment, we used the recall, precision and F-measure external metrics in order to evaluate the quality of the formed clusters by XEdge and XCLS. Precision and recall are external cluster quality metrics based on the comparison of the formed clusters to known external classes (e.g. different domains or sub-DTDs). Given a class Z_j of XML documents with a number n_j of XML documents, a cluster C_i, formed by either XEdge or XCLS, with n_i XML documents, let n_i^j be the number of documents in C_i belonging to Z_j. Then, the precision, recall and F-measure are defined as follows:

$$\text{precision}(\mathrm{C_i}, Z_j) = \frac{n_i^j}{n_i}, \quad \text{recall}(\mathrm{C_i}, Z_j) = \frac{n_i^j}{n_j}$$

$$\text{F-measure}(\mathrm{C_i}, Z_j) = \frac{\text{precision}(\mathrm{C_i}, Z_j) \times \text{recall}(\mathrm{C_i}, Z_j)}{\text{precision}(\mathrm{C_i}, Z_j) + \text{recall}(\mathrm{C_i}, Z_j)}$$

4.1 Experiment 1

The dataset of experiment 1 was a set of 323 heterogeneous XML documents, categorized in 7 domains as described before. The results of that experiment are presented in Table 1. As it can be easily seen, XEdge performs excellent and succeeds to correctly categorize the XML documents in the appropriate clusters. On the other hand, XCLS although succeeding in determining most of the clusters, it fails to correctly identify the Club cluster as well as some documents belonging to Sigmod and Department domains. These results are expected because some XML documents, although belonging in different domains (such as Sigmod and Bibliography) share the same node tags. This property results in erroneous formed clusters by XCLS, because it is based on node summaries and doesn't take into consideration the relationships between nodes (which are different in Sigmod and Bibliography). On the contrary, XEdge performs excellent as it is based on edge summaries, thus it can distinguish between XML documents belonging to different domains as they have totally different edges.

4.2 Experiment 2

The dataset of experiment 2 was a set of 167 homogeneous XML documents, derived from 4 sub-DTDs of the dblp DTD. The main property of those documents was that they shared the same node tags, but different edges. The results of that experiment are presented in Table 2. As it can easily be seen, XEdge performs excellent and succeeds to correctly categorize the XML documents in the appropriate clusters, as it is based on edges summaries and not node summaries. On the other hand, XCLS fails to distinguish documents derived from different DTDs and achieves an average F-measure of only 0,43. This is due to the fact that it considers only nodes and not edges, thus because of the previously mentioned property of the XML documents fails to cluster them properly.

4.3 Experiment 3

The dataset of experiment 3 was a set of 57 homogeneous XML documents, derived from 3 sub-DTDs of the movies DTD as described before. The main property of those documents was that they shared the same node tags and edges. The results of that experiment are presented in Table 3. As it can easily be seen, both XEdge and XCLS fail to cluster properly the XML documents and achieve an average F-measure of 0,57 and 0,66 respectively. These results were expected due to the fact that the XML documents share the same node tags and edges, thus XEdge which is based on edges summaries and XCLS which is based on node summaries are not able to distinguish between XML documents derived from different DTDs. However, the need of clustering real homogeneous XML documents with the above mentioned property is very rare, because such documents usually are considered structural-similar, so there is no need to categorize them. We performed this experiment for evaluation purposes and in order to investigate the drawbacks of XEdge and XCLS.

5. Conclusions and Future Work

In this paper, we proposed XEdge, a unified clustering algorithm for homogeneous and heterogeneous XML documents. At first, each XML document was represented by a LevelEdge, a structure that summarizes the distinct edges in each level of the document. Based on this representation, we next proposed two different distance metrics, depending on the type of XML documents to be clustered. The distance metric adapted to the special structural characteristics of homogeneous and heterogeneous XML documents in order to provide a realistic measure of the structure distance between two XML documents. The next step was the application of XEdge, a partitional clustering algorithm that utilizes the proposed structured representation and distance metric. For improving the clustering results, a preprocessing step for refining the initial points for the formed clusters was applied and an appropriate definition for the cluster representative was presented. The experimental results showed that XEdge outperforms XCLS both in case of homogeneous and heterogeneous XML documents. Finally, we proposed the construction of an index structure over the formed clusters of XML documents in order to boost the performance of existing XML querying algorithms. In the future, we intend to investigate a more advanced and efficient method for exploiting the formed clusters in order to boost the performance of XML querying algorithms. Additionally, we want to compare XEdge with more existing XML clustering algorithms.

6. References

[1] Abiteboul, S., Buneman, P. and Suciu, D. Data on the Web. *Morgan Kaufmann*, 2000.

[2] Bradley, P. S. and Fayyad U.M. Refining Initial Points for k-Means Clustering. In *Proceedings of the 15th International Conference in Machine Learning (ICML '98)* (San Francisco, 1998). 1998, pp. 91-99.

[3] Costa, G., Manco, G., Ortale, R. and Tagarelli, A. A Tree-Based Approach to Clustering XML Documents by Structure. In *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD '04)* (Sydney, Australia, May 26-28, 2004), 2004, pp. 137-148.

[4] Dalamagas, T., Cheng, T., Winkel, K. and Sellis, T.K. A methodology for clustering XML documents by structure. In *Information Systems Journal*, 31(3), 2006, pp.: 187-228.

[5] Doucet, A. and Ahonen-Myka, H. Naïve Clustering of a large XML Document Collection. In *Proceedings of the 2002 Initiative for the Evaluation of XML Retrieval Workshop (INEX '02)*, 2002, pp. 81-87.

[6] Han, J. and Kamber, M. Data Mining: Concepts and Techniques. *Academic Press*, 2001.

[7] Ma, Y. and Chbeir, R. Content and Structure Based Approach For XML Similarity. In *Proceedings of the 2005 Conference on Instructional Technologies (CIT '05)* (Binghamton, Canada, 2005), 2005, pp. 136-140.

[8] Nayak, R. and Xu, S. XCLS: A Fast and Effective Clustering Algorithm for Heterogeneous XML Documents. In *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD '06)* (The Singapore, April 9-12, 2006). 2006, pp. 292-302.

[9] Tagarelli, A. and Greco, S. Toward Semantic XML Clustering. In *Proceedings of the 2006 Siam Conference on Data Mining (SDM '06)* (Maryland, USA, 2006). 2006, pp. 188-199.