Organizing XML Documents on a Peer-to-Peer Network by Collaborative Clustering

Francesco Gullo
University of Calabria, Italy
Giovanni Ponti
ENEA, Italy
Sergio Greco
University of Calabria, Italy

ABSTRACT

XML and peer–to–peer (P2P) technologies are nowadays increasingly being combined in many application scenarios. Indeed, while XML is the standard for representing and exchanging data, P2P networks offer innovative opportunities to users to share and locate distributed data over the Internet. This synergistic coupling of XML and P2P networks becomes particularly appealing in mining contexts, such as document clustering. Clustering XML documents is extensively used to discover meaningful groups in large collections of XML documents according to structure and/or content information. However, despite the growing availability of distributed XML sources and the variety of high-demand environments, existing methods for clustering XML documents work only in a centralized way.

In this chapter we address the problem of clustering XML documents in a collaborative distributed environment. We developed a clustering framework for XML sources distributed on a P2P network. XML documents are modeled based on a transactional representation which uses both XML structure and content information. The clustering method employs a centroid-based partitional scheme suitably adapted to work on a P2P network. Each peer is enabled to compute a clustering solution over its local repository and to exchange the resulting cluster representatives with the other peers. The exchanged cluster representatives are hence used to compute the global clustering solution in a collaborative way. Effectiveness and efficiency of the framework were evaluated on real XML document collections varying the number of peers. Experimental results have shown significant improvements of our collaborative distributed algorithm with respect to the centralized clustering setting in terms of execution time, although achieving clustering solutions that still remain accurate with a moderately low number of nodes in the network.

INTRODUCTION

The extensibility of its markup functionalities along with its natural capability of representing complex real-world objects and their relationships are the keys of success of XML in enabling the development of domain-specific markup languages. This has had a strong impact on the role of XML in the Internet, where XML languages have been developed for a large variety of domain applications, ranging from multimedia and networking to Web content syndication and rendering, from scientific data representation and literature to business processes.

In recent years, the use of XML for data representation and exchange has become central in *high-demand environments*. On the one hand, the growing availability of large XML document repositories has raised the need for fast and accurate organization of such data. In this respect, research on *XML document clustering* has produced a variety of approaches and methods, with different focuses on aspects such as the structure and/or content type of XML features, the XML data representation and summarization model, the XML similarity measures, and the strategy of clustering that was able to such special requirements as dealing with large document collections and high dimensionality, ease for browsing, meaningfulness of cluster

descriptions (Candillier, Tellier, & Torre, 2005; Denoyer & Gallinari, 2008; Doucet & Lehtonen, 2006; Kutty, Tran, Nayak, & Li, 2008; Lian, Cheung, Mamoulis, & Yiu, 2004; Nayak & Xu, 2006; Tran, Nayak, Bruza, 2008; Tagarelli & Greco, 2010).

On the other hand, the inherently distributed nature of XML repositories is also calling for adequate distribute processing techniques that can aid the efficient management and mining of XML data. As an example, think of some Web news services that are in charge of very frequently gathering up-to-date information spanning over thousands of news sources: if such services aim to highlight (new) hot topics through the news channels or provide the users with a (personalized) view on the news headlines, they might be required to apply clustering algorithms to the news articles with a frequency of few minutes.

Clustering XML documents in such high-demand environments is hence challenging as the algorithms developed are to be able to face tight requirements on both processing power and space resources. Existing methods for clustering XML documents are instead designed as centralized systems, which is mainly due to the difficulty of decentralizing most clustering strategies and, additionally, to a number of issues arising in the development of a convenient yet effective summarization of both XML structure and content information in XML documents.

XML distributed applications are increasingly being demanded in several domains, such as software and multimedia sharing, product rating, personal profiling, and many others. A great merit of such an extensive use of XML in distributed applications is due to the popularity of *peer–to–peer* (P2P) networks. A P2P network is a distributed system with the following main properties (Rodrigues & Druschel, 2010). It has a high degree of decentralization, since processing power, bandwidth and space resources are contributed by the peers, which implement both client and server functionalities of the system. In general, there can be a high heterogeneity of resources, in terms of hardware and software architecture, power supply, geographic location. A P2P is mostly self-organizing, since any newly introduced peer node requires little or no manual configuration for the system maintenance. Multiple administrative domains usually characterize the system as the peers are not owned or controlled by a single organization. The deployment costs of a P2P system are typically lower than client-server systems, thanks to its independence of dedicated infrastructure, while the upgrade of the system components is made easier. Because there are few if any peers with centralized state, the P2P system is also more resilient to faults and attacks.

XML is being coupled with P2P networks (Koloniari & Pitoura, 2005; Abiteboul, Manolescu, & Taropa, 2006; Rao & Moon, 2009; Antonellis, Makris, & Tsirakis, 2009), thanks to the merits of P2P in enabling innovative services. Hence, devising a distributed approach to XML clustering based on a P2P system arises as a new challenge in efficient XML data management and mining. Ideally, a distributed approach to efficiently clustering XML documents might satisfy at least the following special requirements:

- *High level of resource distribution* – the decentralization should involve both the data objects and the other data needed to perform clustering.
- *Collaborativeness* – this should be a major advantageous aspect resulting from the establishment of the clustering approach on a P2P network, in order to develop a distributed system that is reliable, resource sharing-aware, and able to effectively and efficiently exchange the processed information.
- *Limited network load* – a summarized data structure should be devised to suitably represent structure and content information in XML data, and to ensure an efficient exchange of information.
- *Ease of implementation* – because each peer in the network is easy to maintain, the clustering strategy adopted by the network nodes and the processing information exchanged between the nodes needs to be feasible on a large-scale system.

This chapter addresses the problem of clustering XML documents on a distributed, peer-to-peer network. A framework recently proposed in (Greco, Gullo, Ponti, & Tagarelli, 2009) is presented. Such a framework exploits the approach in (Tagarelli & Greco, 2010; Tagarelli & Greco, 2006) for modeling and clustering XML documents. That approach allows for transforming XML documents into *transactional data* based on the notion of *tree tuple*, which aims to model any XML document according to a flat, relational-like representation. XML tree tuples are well-suited for clustering XML documents according to both structure and content information.

The distributed environment underlying the framework assumes that only a portion of the input collection of XML documents to be clustered can be accessed by each node in the network. The clustering task is performed according to a collaborative model, where each node communicates with all the other ones. In particular, according to the well-known paradigm of *centroid-based partitional clustering* (Jain & Dubes, 1988), the information exchanged among the nodes has the form of *cluster centroids*. Summarizing the

clustered data based on cluster centroids is highly desirable, especially in scenarios where the input data is spread across a network. Indeed, cluster centroids provide an accurate as well as compact representation of the information to be exchanged among the various nodes in the network. In this way, the exchanged data is highly representative and, at the same time, the network load is maintained relatively small.

The collaborative clustering approach in the distributed framework essentially acts as follows. Each node in the network is responsible for computing a local clustering solution (a partition of its own set of XML data), whose clusters are summarized by "local" centroids that are sent to the other nodes in the network. The nodes are also responsible of computing a subset of the "global" centroids by taking into account the information stored in the corresponding "local" centroids received from the other nodes. The global centroids are eventually sent back to all the nodes to be used for updating the local clustering solution. The process ends when no global centroid changes with respect to the previous iteration.

Experiments were conducted on large, real-world collections of XML documents, whose features are particularly suitable for assessing the validity of distributed collaborative clustering of XML documents. The experiments were performed by varying the number of nodes in the network while keeping the communication among nodes minimized. Results have shown that, although accuracy is typically lower than in the centralized case, a relatively small number of nodes is sufficient for drastically improving the efficiency of the clustering task.

This chapter is organized as follows. Section "Background" briefly discusses related work. Section "XML Transactional Representation" introduces the transaction model used for representing structure and content information from XML documents. Section "Collaborative Clustering of XML Documents" describes the XML transactional similarity measure and the collaborative clustering algorithm. Section "Experimental Evaluation" reports experimental results achieved by our approach from both effectiveness and efficiency viewpoint. Section "Conclusion and Future Research" contains concluding remarks and our pointers for future research directions.

BACKGROUND

In the past few years, there has been a great interest in XML data management in distributed environments. In this respect, early research studies focused on efficient distributed query processing and optimization. (Abiteboul, Manolescu, & Taropa, 2006) describes an extension of the Active XML (AXML) language used to embed service call specification in XML documents, by which XML applications can easily be specified and deployed in complex distributed systems. Antonellis, Makris, and Tsirakis (2009) propose a distributed clustering-based approach to efficiently locate XML documents over a P2P network for the purpose of improving time performance of querying systems. This problem is also studied in (Rao & Moon, 2009), where XML documents and XML query patterns are mapped into algebraic signatures, and distributed hash tables are used to index the document signatures.

XML allows for publishing information that can be edited and reorganized in a distributed collaborative way. (Ignat & Oster, 2008) investigates the problem of XML document reconciliation in a P2P network by proposing an environment in which users perform an off-line editing of XML documents, and their edits are committed and synchronized with the ones performed by the other users as they reconnect to the network. In order to guarantee consistency, a tombstone operational transformation based approach has been employed to merge XML structures of the edited XML document.

Data mining algorithms are traditionally designed to work with data located in a centralized system and/or memory. However, with the development of large-scale systems, there has been an increasing demand for data mining methods that can work in parallel and distributed environments. This is challenging for two main reasons: data mining algorithms usually require large resources in terms of storage systems, and it is highly inefficient to transfer the huge amount of data stored in several (distributed) databases and analyze them in a single centralized site. Early proposals on distributed data mining exploited agent-based architectures. Kargupta, Hamzaoglu, and Stafford (1997) design a distributed system in which each agent is able to model its local world and the global solution is cooperatively obtained by exploiting agent communication exchange. Dhillon and Modha (2000) define a parallel implementation of the k-Means algorithm based on the message passing model. Cesario and Talia (2008) focus on distributed data mining as WSRF services by exploiting the Grid infrastructure. Other examples of distributed data mining can be found in (Cannataro, Congiusta, Pugliese, Talia, & Trunfio, 2004; Kargupta, Park, Hershberger, & Johnson, 1999; Prodromidis, Chan, & Stolfo, 2000).

As regards distributed document clustering in P2P networks, (Eisenhardt, Muller, & Henrich, 2003) discusses the efficiency of partitional clustering approaches like k-Means based on centroids in the task of distributed clustering of documents. Here, local clustering solutions are improved taking into account recommendations exchanged among the peers, whereas clusters are summarized as keyphrases. A very recent P2P approach to distributed document clustering is presented in (Papapetrou, Siberski, & Fuhr, 2010). Here the clustering strategy is again based on a k-Means-like scheme, whereas the main novelty lies in the cluster summary indexing and in the 2-step document-to-cluster assignment, which are probabilistically controlled. Despite the P2P scenario, this approach however ignores any notion of "collaborativeness" in the computation and/or exchange of cluster centroids/summaries, which might improve the efficiency (and parallelism) of the distributed clustering process.

To the best of our knowledge, Hammouda and Kamel are the only authors that have addressed the problem of distributed clustering of (plain) documents from a *collaborative* fashion (Hammouda & Kamel, 2006). They had the intuition that recommendations exchanged between the peers in the network may improve the accuracy of document clustering in a distributed environment. In their work, document cluster summaries in the form keyphrases are also exploited. Our proposed work also exploits a notion of collaborativeness in a distributed environment; however, besides dealing with the more complex case of semistructured (XML) documents, in our approach we devise different strategies for summarizing clusters and exchanging information between the peers. In fact, our XML cluster summaries are defined as transactions that are comprised of items containing highly representative structure and content information present in that cluster of XML documents. Moreover, XML information exchanged among peers is not supplied in the form of recommendations, but in a simpler way that exploits "meta-representatives", which guide the construction of the global clustering solution.

XML TRANSACTIONAL REPRESENTATION

Tagarelli and Greco (2010, 2006) originally introduced an XML representation model that allows for mapping XML document trees into *transactional data*. In a generic application domain, a transaction dataset is a multi-set of variable-length sequences of objects with categorical attributes; in the XML domain, a transaction is devised as a set of items, each of which embeds a distinct combination of structure and content features from the original XML data. Within this view, XML documents are not directly transformed to transactional data, rather they are initially decomposed on the basis of the notion of *tree tuple*. Intuitively, given any XML document, a tree tuple is a tree representation of a complete set of distinct concepts that are correlated according to the structure semantics of the original document tree. Tree tuples extracted from the same tree maintain similar or identical structure while reflect different ways of associating content with structure as they can be naturally inferred from the original tree.

XML tree tuple extraction

Tree tuple resembles the notion of tuple in relational databases and has been proposed to extend functional dependencies to the XML setting (Arenas & Libkin, 2004; Flesca, Furfaro, Greco, & Zumpano, 2003). In a relational database, a tuple is a function assigning each attribute with a value from the corresponding domain.

Given an XML tree *XT*, an *XML tree tuple* τ derived from *XT* is a maximal subtree of *XT* such that, for each (tag or complete) path *p* in *XT*, the size of the answer of *p* on τ is not greater than 1. We denote with T_{XT} and *T* the set of tree tuples that can be derived from any given tree *XT* and from the collection $\boldsymbol{XT}$, respectively. Also, we use P_τ to denote the set of complete paths in a tree tuple τ. Intuitively, a tree tuple is a (sub)tree representation of a set of distinct concepts that are correlated according to the structure semantics of the original tree. Tree tuples extracted from the same tree maintain identical structure while reflect different ways of associating content with structure as they can be naturally inferred from the original tree.

Consider the XML document tree shown in Figure 1, which represents two software tools. Any internal node has a unique label denoting a tag name, whereas each leaf node is labeled with either name and value of an attribute, or symbol *S* and a string corresponding to the #PCDATA content model. Path answers can be easily computed: for example, the path software.tool.name yields the set of node identifiers $\{n_8, n_{20}\}$,

and the path software.tool.developer.*S* yields the set of strings {*"software's developer 1"*, *"software's developer 2"*}.

```
<software>
   <tool>
      <name>software's name 1</name>
      <developer>software's developer 1</developer>
      <developer>software's developer 2</developer>
      <description>software's description 1</description>
      <review>
         <comments>any user's comments 1</comments>
      </review>
   </tool>
   <tool>
      <name>software's name 2</name>
      <developer>software's developer 1</developer>
      <description>software's description 1</description>
      <review>
         <comments>any user's comments 2</comments>
      </review>
      <review>
         <comments>any user's comments 3</comments>
         <recommendation>software's recommendation</recommendation>
      </review>
   </tool>
</software>
```

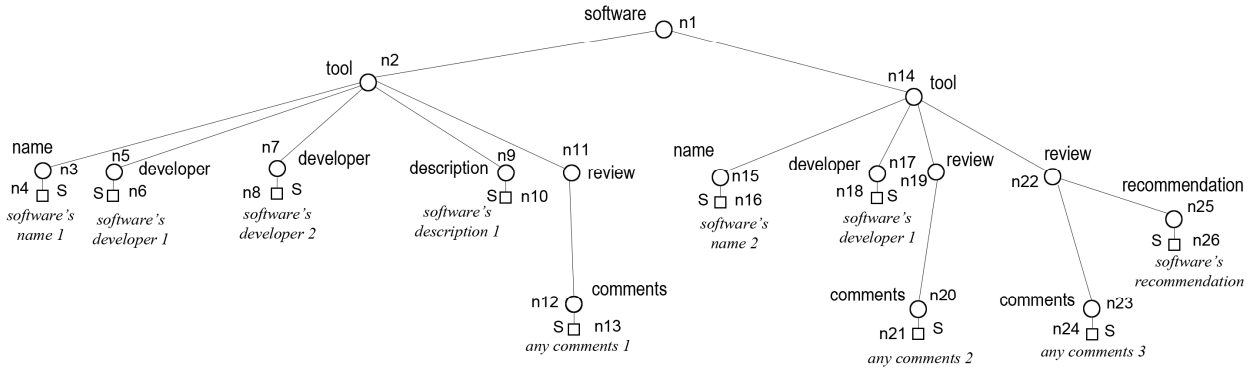
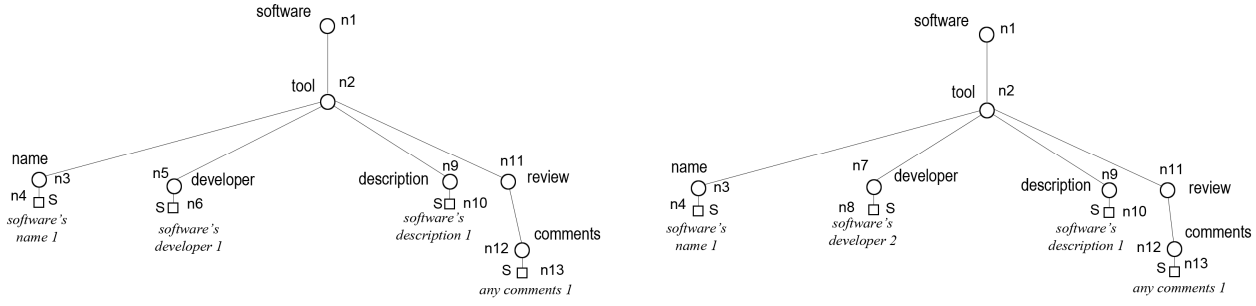


Figure 1. Example XML document and its tree XML

Four tree tuples can be extracted from the example tree; in Figure 2, nodes belonging to the same tree tuple are shaded the same. Two distinct tree tuple are extracted starting from the left subtree rooted in the software element, as here there are two paths software.tool.developer. Two other distinct tree tuples are instead extracted starting from the right subtree rooted in software, as here there are two paths software.tool.review, each of which yields a distinct path answer corresponding to a tool.



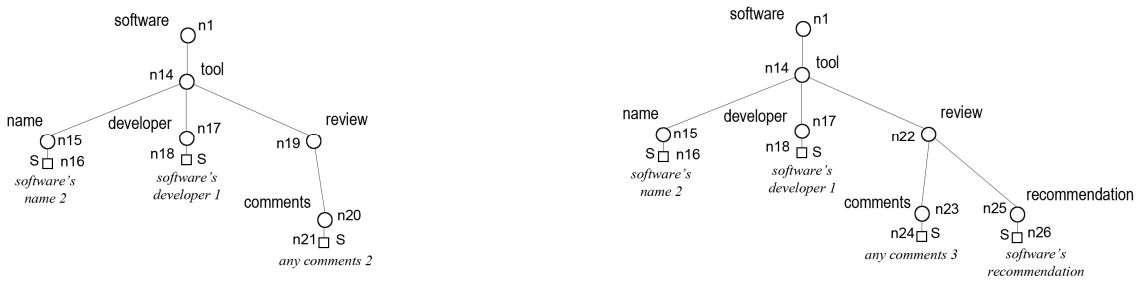


Figure 2. The tree tuples extracted from the XML tree of Figure 1

Transactional modeling of XML tree tuples

In the huge amount of available structured data, a relevant portion is represented by *transactional data*, that is, variable-length sequences of objects with categorical attributes. Given a set $I = \{e_1, \ldots, e_m\}$ of distinct categorical values, or *items*, a transactional database is a multi-set of transactions $tr \subseteq I$. The item domain is built over all the leaf elements in a given collection of XML tree tuples, that is the set of distinct answers of complete paths applied to the tree tuples. A transaction is then modeled with the set of items associated to the leaf elements of a specific tree tuple. The intuition behind such a model lies mainly on the definition of XML tree tuple itself: each path applied to a tree tuple yields a unique answer, thus each item in a transaction indicates information on a concept that is distinct from that of other items in the same transaction.

Formally, given an XML tree tuple τ and a path $p \in P_\tau$, an *XML tree tuple item* in τ is a pair $e = <p, A_\tau(p)>$, where $A_\tau(p)$ denotes the answer of p on τ. The *XML transaction* corresponding to τ is the set of XML tree tuple items of τ.

COLLABORATIVE CLUSTERING OF XML DOCUMENTS

XML tree tuple item similarity

XML features are represented by tree tuple items. Therefore, in order to compare XML data in our transactional domain, we define a measure of similarity between tree tuple items according to their structure and content features. Given two tree tuple items e_i and e_j, the *tree tuple item similarity* function is defined as:

$$sim(e_i, e_j) = f \times sim_S(e_i, e_j) + (1 - f) \times sim_C(e_i, e_j)$$

where sim_S (resp. sim_C) denotes the structural (resp. content) similarity between the items, and $f \in [0, 1]$ is a factor that tunes the influence of the structural part to the overall similarity. Since the combination of structure and content information characterizes an XML tree tuple item, it is advisable to take tolerance on computing similarity between XML tree tuple items. For this purpose, we introduce a similarity threshold $\gamma \in [0, 1]$ that represents the minimum similarity value for considering two XML tree tuple items as similar. According with this, two XML tree tuple items e_i and e_j are said to be *γ-matched* if $sim(e_i, e_j) \geq \gamma$.

Similarity by structure

Structural similarity between two tree tuple items e_i and e_j is evaluated by comparing their respective tag paths. Computing the similarity between any two paths is essentially accomplished by referring to it as a simple case of string matching of their respective element names, and finally averaging the (weighted) matchings. To this end, given any two tags t_1 and t_2, the Dirichlet function (Δ) is applied in such a way that $\Delta(t_1, t_2)$ is equal to one if the tags match, otherwise $\Delta(t_1, t_2)$ is equal to zero.

Given two XML tree tuple items e_i and e_j, let $p_i = t_{i1}.t_{i2}. \ldots .t_{in}$ and $p_j = t_{j1}.t_{j2}. \ldots .t_{jm}$ be their respective tag paths. The *structural similarity* between e_i and e_j is defined as

$$sim_S(e_i, e_j) = \frac{1}{n+m}\left(\sum_{h=1}^{n} s(t_{ih}, p_j, h) + \sum_{k=1}^{m} s(t_{jk}, p_i, k)\right)$$

such that, for each tag t and path $p = t_1.t_2. \ldots .t_L$, $s(t, p, a) = \max_{\{l = 1..L\}} (1+|a-l|)^{-1} \times \Delta(t, t_l)$.

Above, the tag matchings are corrected by a factor which is inversely proportional to the absolute difference of location of the tags in their respective paths. Essentially, this factor penalizes the similarity of two paths that have the same tags but are differently located.

Similarity by content

A *textual content unit* (for short, TCU) is referred to as the preprocessed text of a tree tuple item, that is, a #PCDATA element content or an attribute value. To weight the relevance of terms in TCUs, an adaptation of the popular *tf.idf* (term frequency - inverse document frequency) to the XML transactional domain is defined.

Given a collection $\boldsymbol{XT}$ of XML trees, let w_j be an index term occurring in a TCU u_i of a tree tuple τ belonging to T extracted from a tree XT belonging to $\boldsymbol{XT}$. The *ttf.itf* (*Tree tuple Term Frequency - Inverse Tree tuple Frequency*) weight of w_j in u_i with respect to τ is defined as

$$ttf.itf(w_j, u_i|_\tau) = tf(w_j, u_i) \times \exp\left(\frac{n_{j,\tau}}{N_\tau}\right) \times \frac{n_{j,XT}}{N_{XT}} \times \ln\left(\frac{N_T}{n_{j,T}}\right)$$

where $tf(w_j, u_i)$ denotes the number of occurrences of w_j in u_i, and the other symbols denote the number of TCUs appearing in τ (N_τ) and in the portion containing w_j ($n_{j,\tau}$), in XT (N_{XT}) and in the portion containing w_j ($n_{j,XT}$), in T (N_T) and in the portion containing w_j ($n_{j,T}$). Note that the *ttf.itf* weight increases by increasing each of the factors in the function, namely the term frequency within the specific TCU, the term popularity across the TCUs of the same XML transaction and across the TCUs of the same document tree, and the term rarity across the whole collection of TCUs.

Content similarity between two tree tuple items is computed by measuring the text similarity of their respective TCUs. A vector-space model to represent the TCUs is adopted, therefore any TCU u_i is modeled with a vector whose j-th component corresponds to an index term w_j and contains the *ttf.itf* relevance weight. To measure the similarity between TCU vectors, the well-known *cosine similarity* (Strehl, Ghosh, & Mooney, 2000) is used.

The centralized approach: XK-means algorithm

XML tree tuples modeled as transactions can be efficiently clustered by applying a centralized partitional algorithm devised for the XML transactional domain. A partitional clustering problem consists in partitioning a set $\{x_1, \ldots, x_n\}$ of objects in k non-empty groups each containing a homogeneous subset of objects.

In (Tagarelli & Greco, 2010; Tagarelli & Greco, 2006), a centroid-based partitional clustering algorithm, called XK-means, was developed as a variant of the *k*-Means algorithm for the XML transactional domain. XK-means embeds novel notions of proximity among XML transactions, and XML cluster representative (centroid). The basic idea underlying the algorithm is to search for the "shared" items when comparing any two XML transactions. Following one of the commonly used measures for categorical data, the similarity of two XML transactions could be computed as directly proportional to the number of common items and inversely proportional to the number of different items. However, computing exact intersection between XML transactions will likely lead to very poor results, since two XML (tree tuple) items may share relevant structural or content information even though they are not identical. Therefore, the notion of standard intersection between sets of items was enhanced with one able to capture similarities, rather than strictly exact matching, from content and structure features of XML elements.

Let tr_1 and tr_2 be two transactions, and $\gamma \in [0,1]$ be a similarity threshold. The set of *γ-shared items* between tr_1 and tr_2 is defined as

$$match^{\gamma}(tr_1, tr_2) = match^{\gamma}(tr_1 \rightarrow tr_2) \cup match^{\gamma}(tr_2 \rightarrow tr_1)$$

where $match^{\gamma}(tr_i \rightarrow tr_j) = \{e \in tr_i \mid \exists e_h \in tr_j, sim(e, e_h) \geq \gamma, \neg\exists e' \in tr_i, sim(e', e_h) > sim(e, e_h)\}$. The set of γ-shared items resembles the intersection between transactions at a degree greater than or equal to a similarity threshold γ. This notion of (enhanced) intersection is also at the basis of the *XML transaction similarity* function:

$$sim_J^{\gamma}(tr_1, tr_2) = \frac{\left|match^{\gamma}(tr_1, tr_2)\right|}{\left|tr_1 \cup tr_2\right|}$$

The above function is used in *XK-means* to measure the proximity among transactions. Moreover, given a cluster *C*, its centroid *c* is computed by starting from the set of γ-shared items among all the transactions within C. More precisely, for each transaction in *C*, the union of the γ-shared item sets with respect to all the other transactions in *C* is obtained; this guarantees no dependence of the order of examination of the transactions. Then, the set of γ-shared items is involved to compute a centroid having the form of a tree tuple. According to such a function, a raw centroid is firstly defined by selecting the items from these union sets with the highest frequency: the raw centroid, however, may not have the form of a tree tuple, as some items therein may refer to the same path but with different answers. Any raw centroid is transformed into a tree tuple. This step involves a set *I* of items and yields a tree tuple composed by all the distinct paths *p* involved into the items in *I*; the content associated to each path *p* is the union of the contents of the items in *I* having *p* as a path. A greedy heuristic refines the current centroid by iteratively adding the remaining most frequent items until the sum of pair-wise similarities between transactions and centroid cannot be further maximized. Performing again this step, any refinement guarantees that the resulting centroid satisfies the requirements presented above.

The distributed approach: CXK-means algorithm

The main novelties and intuitions introduced in *XK-means*, such as XML transaction proximity measure and cluster centroid for a set of XML transaction, are exploited by the *CXK-means* XML transactional clustering algorithm for a distributed collaborative environment (Greco, Gullo, Ponti, & Tagarelli, 2009). The pseudo-code of *CXK-means* is sketched in Figure 3.

Algorithm: CXK-MEANS

Input: set *S* of XML transactions distributed over *m* nodes,
number *k* of clusters, similarity threshold γ
Output: partition ***C*** of *S* in *k* clusters distributed over *m* nodes

Process N_0
1: define a partition of {1, …, *k*} into *m* subsets $\{Z_1, \ldots, Z_m\}$
// Z_i is the responsibility assignment of node N_i
2: **send** the chosen *responsibility assignments* to each node

Process N_i
3: select a global centroid g_j for each $j \in Z_i$
4: **repeat**
5: **send** global centroids g_j for each $j \in Z_i$ to all other nodes
6: **receive** the remaining global centroids from all the other nodes
7: set the initial local centroids l^i_j as equal to the global centroids
8: assign each transaction to the cluster with the closest local centroid (according to the sim^{γ}_J function)
9: re-compute local centroids l^i_j of each cluster C^i_j

```
10:   if each l_j does not change then
11:     send a termination signal to all other nodes
12:   else
13:     send a continuation signal and local centroids to all the other nodes
14:   receive local centroids from all the other nodes
15:   if there exists a node that does not terminate then
16:     compute global centroids g_i
          for each j ∈ Z_i employing local ones received by the other nodes
17: until each node terminates
```

Figure 3. The CXK-means clustering algorithm

In the *CXK-means* algorithm, document data are distributed over m nodes and each node communicates with all the other ones sending "local" centroids and receiving "global" centroids. An initial process corresponding to a node N_0 defines a partition of the set $\{1, \ldots, k\}$ of cluster identifiers into m subsets. Each set contains the identifiers of the clusters for which the node N_j has the responsibility of computing the global centroids (i.e., the *responsibility assignments*). It should be noted that the presence of node N_0 does not contrast the collaborative nature of the *CXK-means* algorithm. Indeed, N_0 is not responsible of summarizing the information coming from the various peers (nodes $N_1, \ldots, N_m$) and, therefore, does not act as a coordinator; rather, N_0 performs only trivial startup operations which, in principle, can be performed by any other peer node $N_1, \ldots, N_m$.

Each node N_i is in charge of computing local clusters $C^i_1, \ldots, C^i_k$ and local centroids $l^i_1, \ldots, l^i_k$, but also a subset g_j of the global centroids; this is performed by looking at its own responsibility assignment and the local centroids computed by all nodes. Each node has a process that executes a classic k-Means-like partitional clustering scheme on its local data in S^i (S^i is the subset of the whole set S of transactions available from the original distributed dataset stored into node N_i). The clustering process employs global centroids received from each other node in the network and terminates when transaction assignments to local clusters do not change. For each node N_i, the local centroid of a cluster C^i_j is computed by starting from the set of γ-shared items among all the transactions within C^i_j. The procedure followed is the one employed by *XK-means*.

The global centroid g_j of a cluster C_j is computed in a way similar to that employed for local centroids. A major difference is that global centroids are computed by considering the m local centroids $l^1_j, \ldots, l^m_j$ (computed by each node $N_1, \ldots, N_m$) along with their respective weights $|C^1_j|, \ldots, |C^m_j|$, which are needed for taking also into account the size of the cluster summarized by each node. The rationale in this respect is that the greater the weight $|C^i_j|$ (i.e., the greater the number of transactions belonging to the cluster C_j stored into the local repository S^i at node i), the more reliable the local centroid l^i_j in summarizing cluster C_j is – in other terms, l^i_j is more reliable than any other local centroid $l^{i'}_j$ outputted by a node $i' \neq i$ such that $|C^{i'}_j| < |C^i_j|$.

Nodes communicate their local state by sending a flag to other nodes in the network. In particular, a node sends a termination signal if, at the end of its local clustering process, all its local cluster centroids do not change with respect to the ones computed in the previous execution. In this way, the collaborative clustering process continues until each node in the network reaches a stable clustering solution.

EXPERIMENTAL EVALUATION

Experimental setting

The experimental evaluation was mainly conceived to evaluate the performance of *CXK-means* with respect to *XK-means* in terms of both efficiency and effectiveness. In this respect, the number of nodes in the peer–to–peer distributed environment was varied from 1 to a maximum of 19, where a network size equal to 1 clearly refers to the baseline case represented by the centralized case (i.e., *XK-means* algorithm).

Data description

Two real-world document collections were involved into the evaluation. Figure 2 illustrates the XML DTDs that can be inferred from each of the evaluation datasets. Note that these schemas are presented only

for purposes of description of the datasets involved into the experiments, that is, they were not used during the evaluation since the approach at hand does not require the availability of XML schemas.

The *DBLP* dataset is a subset of the DBLP archive,[1] a digital bibliography on computer science which contains citations on journal articles, conference papers, books, book chapters, and theses. *DBLP* is comprised of 3,000 documents which correspond to 5,884 transactions and 8,231 items. *DBLP* is characterized by a small average depth (3), whereas the number of leaf nodes is 13,209 and the maximum fan out is 20. According to its element type definition (Fig. 2(a)), exhibits short text descriptions (e.g., author names, paper titles, conference names), and a moderate structural variety which corresponds to 4 main structural categories, namely "journal articles" (*article*), "conference papers" (*inproceedings*), "books" (*book*), and "book chapters" (*incollection*). Also, 6 topical classes are identified in *DBLP*, which are "multimedia", "logic programming", "web and adaptive systems", "knowledge based systems", "software engineering", and "formal languages". If both content and structure information are taken into account, 16 classes are identified.

The *IEEE* dataset refers to the IEEE collection version 2.2, which has been used as a benchmark in the INEX document mining track 2008[2]. *IEEE* consists of 4,874 articles originally published in 23 different IEEE journals from 2002 to 2004. Such articles follow a complex schema which includes front matter, back matter, section headings, text formatting tags and mathematical formulas. We kept most of the logical structure elements and removed the stylistic markups, as shown in the DTD of Fig. 2(b). In our XML transactional domain, the *IEEE* collection has 211,909 transactions and 135,869 items. Also, the number of leaf nodes is 228,869, the maximum fan out is 43, and the average depth is about 5. In *IEEE*, the article journals determine the categories that were used to partition the collection, which strictly follow the original INEX categorization. Precisely, two structural categories correspond to "Transactions" and "non-Transactions" articles, respectively, whereas the classification by content organizes the articles by the following 8 topic-classes: "Computer", "Graphics", "Hardware", "Artificial Intelligence", "Internet", "Mobile", "Parallel", and "Security". Moreover, 14 hybrid classes are identified according to these structural and content classes.

Cluster validity measures

The quality of clustering solutions for the datasets was assessed by exploiting the availability of reference classifications for XML documents. The objective was to evaluate how well a clustering fits a predefined scheme of known classes (natural clusters). For this purpose, the evaluation employed the well-known *F-measure* (Larsen, & Aone, 1999), which is defined as the harmonic mean of values that express two notions from Information Retrieval, namely *Precision* and *Recall*. F-measure ranges within [0,1], where higher values refer to better quality results. Since tree tuple decomposition of XML documents and then transactional modeling were performed as preliminary phases, the evaluation process take into account the set S of XML transactions.

Given a set $S = \{tr_1, \ldots, tr_m\}$ of XML transactions, let $\Gamma = \{\Gamma_1, \ldots, \Gamma_H\}$ be the reference classification of the objects in S, and $\boldsymbol{C} = \{C_1, \ldots, C_K\}$ be the output partition yielded by a clustering algorithm. *Precision* of cluster C_j with respect to class Γ_i is the fraction of the objects in C_j that has been correctly classified, whereas *Recall* of cluster C_j with respect to class Γ_i is the fraction of the objects in Γ_i that has been correctly classified. Formally,

$$P_{ij} = \frac{\left|C_j \cap \Gamma_i\right|}{\left|C_j\right|} \quad R_{ij} = \frac{\left|C_j \cap \Gamma_i\right|}{\left|\Gamma_i\right|} \quad F_{ij} = \frac{2P_{ij}R_{ij}}{\left(P_{ij} + R_{ij}\right)}$$

In order to score the quality of C with respect to Γ by means of a single value, the overall F-measure $F(\boldsymbol{C}, \Gamma)$ is computed using the weighted sum of the maximum F_{ij} score for each class Γ_i.

$$F(\boldsymbol{C}, \Gamma) = \frac{1}{\left|S\right|} \sum_{i=1}^{H} \left|\Gamma_i\right| \max_{j \in [1..K]} F_{ij}$$

[1] http://dblp.uni-trier.de/xml/

[2] http://www.inex.otago.ac.nz/data/documentcollection.asp

Results

Effectiveness

Table 1(a–b) illustrates accuracy performances obtained on (a) *DBLP* an (b) *IEEE* by *CXK-means* when data are equally partitioned over the nodes. Experiments have been performed for each of the clustering type (i.e., content-, structure-, and content/structure-driven) by varying the number of nodes. For each dataset and clustering setting, results refer to multiple (10) runs of the algorithm and correspond to F-measure scores averaged over the range of f values specific of the clustering setting. Moreover, the best setting of parameter γ was found to be close to high values (typically above 0.85), for each dataset and type of clustering (Tagarelli & Greco, 2010).

Clustering type (#clusters)	**#nodes**	**avg F-measure**
$f \in$ [0, 0.3] (content-driven similarity) (6 clusters)	1	0.795
	3	0.730
	5	0.701
	7	0.639
	9	0.574
$f \in$ [0.4, 0.6] (structure/content-driven similarity) (16 clusters)	1	0.803
	3	0.750
	5	0.716
	7	0.641
	9	0.585
$f \in$ [0.7, 1] (structure-driven similarity) (4 clusters)	1	0.991
	3	0.971
	5	0.935
	7	0.855
	9	0.751

(a)

Clustering type (#clusters)	**#nodes**	**avg F-measure**
$f \in$ [0, 0.3] (content-driven similarity) (8 clusters)	1	0.629
	3	0.552
	5	0.514
	7	0.440
	9	0.396
$f \in$ [0.4, 0.6] (structure/content-driven similarity) (14 clusters)	1	0.598
	3	0.524
	5	0.478
	7	0.423
	9	0.375
$f \in$ [0.7, 1] (structure-driven similarity) (2 clusters)	1	0.655
	3	0.572
	5	0.527
	7	0.453
	9	0.406

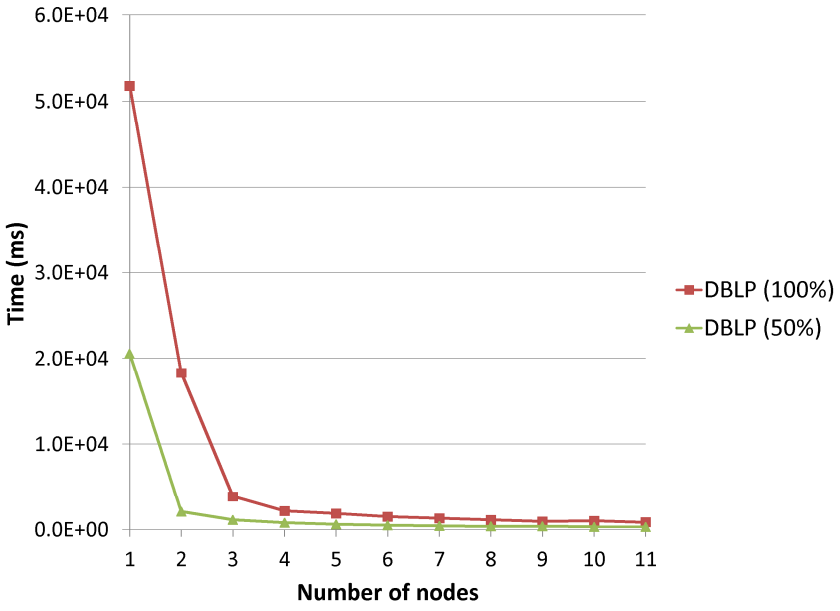
(b)

Table 1. Clustering accuracy results for data equally distributed over the nodes: (a) DBLP and (b) IEEE

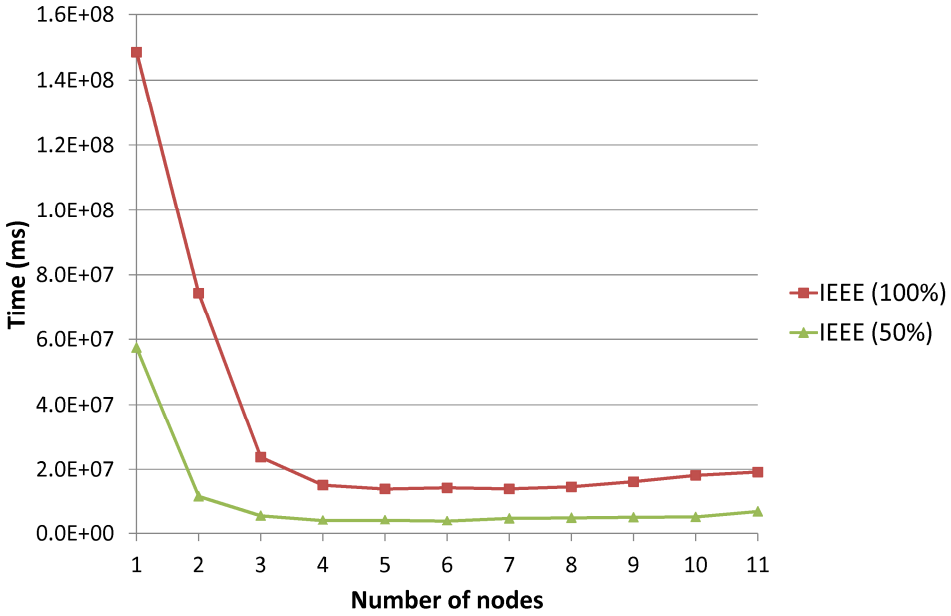
As it is reasonable to expect, the centralized case corresponds to an upper bound in terms of clustering quality for the distributed collaborative approach. While the focus here is not on the effectiveness evaluation of the centralized case, it can be noted how the clustering accuracy decreases as the number of nodes increases, regardless of the dataset and the type of clustering. However, this performance degradation remains relatively acceptable for a distributed environment, which is partly due to the model of cluster centroid in achieving good quality summaries for the clusters. Indeed, loss of accuracy of *CXK-means* with respect to *XK-means* was always lower than 0.2 in relation to the number of nodes leading to the stabilization of efficiency performance determined in the previous paragraph (i.e., 4 and 6 for *DBLP* and *IEEE*, respectively); particularly, the decrease in accuracy were roughly equal to 0.08 (*DBLP*) and 0.14 (*IEEE*).

Efficiency

Figure 4 shows time performances on the four evaluation datasets by increasing the number of nodes. The dataset size was also varied in order to consider the whole datasets and reduced-size (50%) datasets and show the scalability of *CXK-means*. Results refer to structure/content-driven clustering experiments ($f \in [0.4,0.6]$) and data equally distributed in the network.



(a)



(b)

Figure 4. Clustering time performances varying the number of nodes and the dataset size: (a) DBLP, and (b) IEEE

Here a noteworthy remark is that the performance of *CXK-means* takes major advantages with respect to *XK-means* in terms of runtime behavior. In fact, a higher number of nodes in the network leads to more

parallelism, which results in a drastic reduction of the overall time needed for the clustering task. However, when the number of nodes grows up, the data exchanged among nodes grow up as well. This fact clearly affects negatively the network traffic (i.e., the centroid exchange) which might not be negligible anymore. Indeed, as it can be noted in Fig. 3 for both the datasets, after a drastic reduction of the runtime due to the use of just a few nodes, the runtime remains roughly constant for a certain range, then it starts to slightly increase when the number of nodes becomes significantly higher. Regarding the evaluation of the stabilization (saturation) points, it can be noted that time performances on *IEEE* tend to stabilize for 6 and 4 nodes, respectively in the case of full and halved dataset; on *DBLP*, time performances tend to stabilize for a smaller number of nodes (4 and 2, respectively) which is due to a smaller size of *DBLP* with respect to *IEEE*, in terms of both transactions and vocabulary of terms.

Another important remark is that, as the dataset size is halved, the minimum number of nodes to bring down the clustering times tends to decrease. This further confirms that the advantage of the distributed collaborative approach with respect to the centralized one tends to become less significant as the dataset size is reduced.

CONCLUSION AND FUTURE RESEARCH

In this chapter we presented a collaborative clustering approach to organizing XML documents in P2P networks. The framework is based on a distributed, centroid-based partitional clustering algorithm, where cluster centroids are used to describe local and global clusters of the input document collection. Each node yields a local clustering solution, whose clusters are represented by local centroids. These centroids are exchanged with the other nodes in the network in a collaborative way; the ultimate goal is to exploit such local centroids for computing global clusters that form the overall clustering solution. Experimental results have shown that XML distributed collaborative clustering is more efficient than the corresponding centralized clustering setting, though paying a slight decrease in accuracy.

There are several directions that are worthy of investigation. Some of them involve enhancements of the framework we have proposed in this chapter. For instance, semantic information of both structural and content type from XML data could be integrated in our framework at different levels, following the study provided in (Tagarelli & Greco, 2010). Other future research directions regard in general the impact of using a distributed approach to XML document clustering in such contexts as data integration and information retrieval. Collaborative distributed clustering of XML documents can in fact efficiently support the integration and classification of heterogeneous XML sources, but the extent to which this can be useful need to be thoroughly investigated. The summarization and extraction of information from the Web and the consequent organization of selectively structured information (wrapped data) represent real-world scenarios in which collaborative distributed clustering of XML documents can help in different ways. For instance, it clustering can be used to preliminarily filter subsets of documents that are cohesive with respect to a specific domain of interest for the user who needs to summarize, extract and organize the extracted data. Also, distributed clustering can easily be applied to the result of summarization and extraction of heterogeneous Web sources without requiring any effort of centralization of resources.

REFERENCES

Abiteboul, S., Manolescu, I., & Taropa, E. (2006). A Framework for Distributed XML Data Management. In Y. E. Ioannidis, M. H. Scholl, J. W. Schmidt, F. Matthes, M. Hatzopoulos, K. Böhm, A. Kemper, T. Grust, and C. Böhm (Eds.), *Proceedings of the 10th International Conference on Extending Database Technology (EDBT): Vol. 3896. Lecture Notes in Computer Science* (pp. 1049-1058). Springer.

Antonellis, P., Makris, C., & Tsirakis, N. (2009). Utilizing XML Clustering for Efficient XML Data Management on P2P Networks. In S. S. Bhowmick, J. Küng, and R. Wagner (Eds.), *Proceedings of the 20th International Conference on Database and Expert Systems Applications (DEXA): Vol. 5690. Lecture Notes in Computer Science* (pp. 68-82). Springer.

Arenas, M., & Libkin, L. (2004). A Normal Form for XML Documents. *ACM Transactions On Database Systems*, 29(1), 195–232.

Baeza-Yates, R., & Ribeiro-Neto, B. (1999). *Modern Information Retrieval*. Addison Wesley.

Candillier, L., Tellier, I., & Torre, F. (2005). Transforming XML Trees for Efficient Classification and Clustering. In N. Fuhr, M. Lalmas, S. Malik, and G. Kazai (Eds.), *Proceedings of the 4th International Workshop of the Initiative for the Evaluation of XML Retrieval (INEX): Vol 3977. Lecture Notes in Computer Science* (pp. 469-480). Springer.

Cannataro, M., Congiusta, A., Pugliese, A., Talia, D., & Trunfio, P. (2004). Distributed Data Mining on Grids: Services, Tools, and Applications. *IEEE Transactions on Systems, Man, and Cybernetics: Part B, 34*(6), 2451-2465.

Cesario, E., & Talia, D. (2008). Distributed Data Mining Models as Services on the Grid. In *Workshops Proceedings of the 8th IEEE International Conference on Data Mining (ICDM),* (pp. 486-495). IEEE Computer Society.

Costa, G., Manco, G., Ortale, R., & Tagarelli, A. (2004). A Tree-based Approach to Clustering XML Documents by Structure. In J.-F. Boulicaut, F. Esposito, F. Giannotti, and D. Pedreschi (Eds.), *Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD): Vol. 3202. Lecture Notes in Computer Science* (pp. 137-148). Springer.

Denoyer, L., & Gallinari, P. (2008). Report on the XML Mining Track at INEX 2007: Categorization and Clustering of XML Documents. *SIGIR Forum, 42*(1), 22-28.

Dhillon, I. S., & Modha, D. S. (2000). A Data-Clustering Algorithm On Distributed Memory Multiprocessors. In M. Javeed Zaki, and C.-T. Ho (Eds.), *Proceedings of the Workshop on Large-Scale Parallel KDD Systems (KDD): Vol. 1759. Lecture Notes in Computer Science* (pp. 245-260). Springer.

Dhillon, I. S., & Modha, D. S. (2001). Concept Decompositions for Large Sparse Text Data Using Clustering. *Machine Learning, 42*(1/2), 143-175.

Doucet, A., & Lehtonen, M. (2006). Unsupervised Classification of TextCentric XML Document Collections. In *Proceedings of the 5th International Workshop of the Initiative for the Evaluation of XML Retrieval (INEX): Vol. 4518. Lecture Notes in Computer Science* (pp. 497-509). Springer.

Eisenhardt, M., Muller, W., & Henrich, A. (2003). Classifying Documents by Distributed P2P Clustering. In K. R. Dittrich, W. König, A. Oberweis, K. Rannenberg, W. Wahlster (Eds.), *Proceedings of the INFORMATIK 2003 - Innovative Informatikanwendungen (GI Jahrestagung (2)): Vol. 35. Lecture Notes in Informatics* (pp. 286-291). GI.

Flesca, S., Furfaro, F., Greco, S., & Zumpano, E. (2003). Repairs and Consistent Answers for XML Data with Functional Dependencies. In Z. Bellahsene, A. B. Chaudhri, E. Rahm, M. Rys, R. Unland (Eds.), *Proceedings of the 1st International XML Database Symposium (XSym): Vol. 2824.* Lecture Notes in Computer Science (pp. 238-253). Springer.

Greco, S., Gullo, F., Ponti, G., & Tagarelli, A. (2009). Collaborative Clustering of XML Documents. In *Proceedings of the International Workshop on Distributed XML Processing: Theory and Practice (ICPPW)* (pp. 579-586). IEEE Computer Society.

Hammouda, K., & Kamel, M. (2006). Collaborative Document Clustering. In J. Ghosh, D. Lambert, D. B. Skillicorn, and J. Srivastava (Eds.), *Proceedings of the SIAM International Conference on Data Mining (SDM),* (pp. 451-461). SIAM.

Ignat, C.-L., & Oster, G. (2008). Peer–to–Peer Collaboration over XML Documents. In *Proceedings of the 5th international conference on Cooperative Design, Visualization, and Engineering (CDVE,)* (pp. 66-73). Springer.

Jain, A., & Dubes, R. (1988). *Algorithms for Clustering Data*. Prentice-Hall.

Kargupta, H., Hamzaoglu, I., & Stafford, B. (1997). Scalable, Distributed Data Mining Using an Agent Based Architecture. In *Proceedings of the European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD),* (pp. 211-214).

Kargupta, H., Park, B. H., Hershberger, D., & Johnson, E. (1999). Collective Data Mining: A New Perspective Toward Distributed Data Mining. In *Proceedings of Advances in Distributed and Parallel Knowledge Discovery* (pp. 133-184).

Koloniari, G., & Pitoura, E. (2005). Peer–to–Peer Management of XML Data: Issues and Research Challenges. *SIGMOD Record, 34*(2), 6-17.

Kutty, S., Tran, T., Nayak, R., & Li, Y. (2008). Clustering XML Documents Using Closed Frequent Subtrees: A Structural Similarity Approach. In N. Fuhr, J. Kamps, M. Lalmas, and A. Trotman (Eds.), *Focused Access to XML Documents, 6th International Workshop of the Initiative for the Evaluation of XML Retrieval (INEX): Vol. 4862. Lecture Notes in Computer Science* (pp. 183-194). Springer.

Larsen, B., & Aone, C. (1999). Fast and Effective Text Mining Using Lineartime Document Clustering. In *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD),* (pp. 16-22). ACM.

Lian, W., Cheung, D., Mamoulis, N., & Yiu, S. (2004). An Efficient and Scalable Algorithm for Clustering XML Documents by Structure. *IEEE Transactions On Knowledge and Data Engineering, 16*(1), 82-96.

Nayak, R., & Xu, S. (2006). XCLS: A Fast and Effective Clustering Algorithm for Heterogeneous XML Documents. In W. Keong Ng, Masaru Kitsuregawa, J. Li, and K. Chang (Eds.), *Proccedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD): Vol. 3918. Lecture Notes in Computer Science* (pp. 292-302). Springer.

Nierman, A., & Jagadish, H. (2002). Evaluating Structural Similarity in XML Documents. In *Proceedings of the ACM SIGMOD International Workshop on the Web and Databases (WebDB),* (pp. 61-66).

Papapetrou, O., Siberski, W., & Fuhr, N. (2010). Text Clustering for Peer-to-Peer Networks with Probabilistic Guarantees. In C. Gurrin, Y. He, G. Kazai, U. Kruschwitz, S. Little, T. Roelleke, S. M. Rüger, and K. van Rijsbergen (Eds.), *Proceedings European Conference on Information Retrieval Research (ECIR): Vol. 5993. Lecture Notes in Computer Science* (pp. 293-305). Springer.

Polyzotis, N., & Garofalakis, M. (2002). Structure and Value Synopses for XML Data Graphs. In *Proceedings of the International Conference on Very Large Data Bases (VLDB),* (pp. 466-477). Morgan Kaufmann.

Prodromidis, A. L., Chan, P. K., & Stolfo, S. J. (2000). Metalearning in Distributed Data Mining Systems: Issues and Approaches. In *Proceedings of the Advances in Distributed and Parallel Knowledge Discovery* (pp. 81-87).

Rao, P. R., & Moon, B. (2009). Locating XML Documents in a Peer–to–Peer Network Using Distributed Hash Tables. *IEEE Transanctions On Knowledge and Data Engineering, 21*(12), 1737-1752.

Rodriguez, R., & Druschel, P. (2010). Peer–to–Peer Systems. In *Communications of the ACM, 53*(10), 72-82.

Steinbach, M., Karypis, G., & Kumar, V. (2000). A Comparison of Document Clustering Techniques. In *Proceedings of the KDDWorkshop on Text Mining.*

Strehl, A., Ghosh, J., & Mooney, R. (2000). Impact of Similarity Measures on Web-page Clustering. In *Proceedings of the AAAI Workshop on AI for Web Search* (pp. 58-64). AAAI.

Tagarelli, A., & Greco, S. (2006). Toward Semantic XML Clustering. In J. Ghosh, D. Lambert, D. B. Skillicorn, and J. Srivastava (Eds.), *Proceedings of the SIAM International Conference on Data Mining (SDM),* (pp. 188-199). SIAM.

Tagarelli, A., & Greco, S. (2010). Semantic Clustering of XML Documents. *ACM Transactions on Information Systems, 28*(1), 1-56.

Tran, T., Nayak, R., & Bruza, P. (2008). Combining Structure and Content Similarities for XML Document Clustering. In J. F. Roddick, J. Li, P. Christen, and P. J. Kennedy (Eds.), *Proceedings of the 7th Australasian Data Mining Conference (AusDM): Vol. 87. CRPIT* (pp. 219-226). Australian Computer Society.

Vercoustre, A. M., Fegas, M., Gul, S., & Lechevallier, Y. (2005). A Flexible Structured-based Representation for XML Document Mining. In N. Fuhr, M. Lalmas, S. Malik, and G. Kazai (Eds.), *Advances in XML Information Retrieval and Evaluation, 4th International Workshop of the Initiative for the Evaluation of XML Retrieval (INEX): Vol. 3977. Lecture Notes in Computer Science* (pp. 443-457). Springer.

Zhao, Y., & Karypis, G. (2004). Empirical and Theoretical Comparison of Selected Criterion Functions for Document Clustering. *Machine Learning, 55*(3), 311-331.