Application of Different Clustering Algorithms to Multilevel Clustering of XML Documents

Michał Kozielski[1]

Abstract: The large sets of XML documents which are created make the new possibilities for data mining analysis such as clustering. The existing clustering algorithms are not dedicated for hierarchical structure of XML documents and therefore they do not meet all the requirements which may be stated considering different applications. In this paper the application of clustering algorithm to accelerating queries on XML documents is considered. Three different implementations of Multilevel Clustering of XML Documents according to their structure (*ML*) are presented in the paper. Different types of clustering algorithms which are used in the implementations of *ML* method are analysed and verified in the experiments on two sets of XML documents. The results of the analysis show that the proposed implementations of *ML* method perform better then *HCM* algorithm. Additionally they process only a small subspace of a feature vector opposite to *HCM* algorithm which analysis the whole feature vector space.

Keywords: clustering, clustering XML documents.

1. Introduction

XML (eXtensible Markup Language) has become a popular and commonly used standard for data representation and interchange. A large number of XML documents which are used imposed a need for development of the systems effectively storing and processing both XML documents and information contained in them. The constantly increasing number of XML documents and the existence of powerful database systems enabling storage of the documents make the application of data mining algorithms to XML documents more and more interesting and intensively developed (XML data mining 2006).

One of the methods of data mining which may be applied to XML is clustering of XML documents. When clustering XML documents it is possible to analyse:

- text encapsulated in XML documents,
- structure of XML documents,
- text encapsulated in XML documents in the context of its position in the structure of the documents.

[1] Institute of Informatics, Silesian University of Technology, Akademicka 16, 44-100 Gliwice
e-mail: michal.kozielski@polsl.pl

The paper presented describes a method of multilevel clustering of XML documents according to their structure (*ML*) introduced in Kozielski (2007). A given paper (comparing to Kozielski (2007)) presents a formal definition of *ML* algorithm and a comparison of performance of three implementations of *ML* method using different clustering algorithms and two datasets having different characteristics.

Typical clustering algorithms (Han and Kamber 2004, Hand, Mannila and Smyth 2005, Jain, Murty and Flynn 1999) were not designed for hierarchical documents and do not meet all the requirements which may be stated for such data. *ML* method is dedicated to hierarchical structure of XML documents and it may take advantage of different clustering algorithms. Depending on the clustering approach used in *ML* algorithm it is possible to receive different results. The analysis of the clustering results concerning the application of different types of *ML* method to accelerating query execution on a set of XML documents is presented in the paper.

The paper is organised as follows. Section 2 describes the approach to accelerating XML queries by clustering XML documents according to their structure. Section 3 presents justification of the chosen XML structure encoding method, Multilevel clustering of XML documents algorithm and application of different clustering algorithms to *ML* algorithm. Datasets which were used in the analysis and the results of the experiments which were performed are presented in section 4. The final conclusions are drawn in section 5.

2. Accelerating query execution on XML documents

Elements and attributes of XML documents which are addressed in the queries are defined by means of path expressions. Flexibility of the structure of XML documents stored in a database causes that occurrence of an element or attribute may be optional and not all the documents in a collection match a path specified in a query. Assuming that an execution of a query on a subset of documents is less time consuming then querying the whole collection it is worth verifying the methods which could determine the collection subsets addressing the given queries.
Occurrence of an element or an attribute is a feature of a structure of XML document. It is possible therefore, to apply methods of clustering XML documents according to their structure to determine such document subsets. The result of a clustering algorithm is a set of clusters of documents having similar structure within the cluster and different structure between the clusters. Having a cluster of documents it should be possible to calculate a signature of the cluster representing all the features (elements and attributes) existing in the cluster. It should be also possible to calculate a signature of a query representing all the features which are addressed by a query path. Comparison of the two signatures (of a cluster and a query) should show whether the query addresses any documents in a cluster and whether the XML documents in a cluster should be processed by the query. The method enabling acceleration of a query execution which was presented above may be summed up in the following points:

- Cluster the documents according to their structure.

- For each cluster calculate a signature representing all the features existing in the clustered documents.
- Calculate a signature representing all the features addressed by a query.
- Compare the signatures of the clusters with a signature of a query and determine which clusters are addressed by the query.
- Execute the query on the documents belonging to the chosen clusters.

3. Clustering of XML documents

Clustering of XML documents according to their structure may be defined as determining the groups of XML documents having similar structure within the group and different structure between the groups. In order to perform clustering of XML documents according to their structure it is needed to:

- apply one of the methods calculating similarity or distance between the XML document structures,
- apply one of the clustering algorithms determining the clusters.

3.1. Similarity of XML structure feature vectors

In the work presented an approach calculating structural similarity or distance on the bases of feature vectors encoding the structure features of XML documents was used. In general, the process encoding structural information into the feature vectors consists of two steps:

- analysis of all the documents creating a data set in order to create a dictionary of all the features occurring in the data set,
- encoding of each document according to the created dictionary.

A method called bit encoding was used in the experiments performed. It provides a very clear model of XML structure which simplifies calculations needed in the application of clustering algorithm to accelerating query execution on XML documents. Bit encoding method defines a structure of XML document as a string of bits. Each bit in a feature vector denotes occurrence or lack of occurrence of a chosen XML structure fragment in the analysed document. A pair of bits (Lian et al. 2004) connected in a parent-child relation or a path (Yoon, Raghavan and Chakilam 2001) starting from a root element and leading to a chosen node may be taken as a structure fragment. Distance between two document structures encoded in this way may by calculated according to one of the following formulas:

$$dist(d_i, d_j) = 1 - \frac{|d_i \cap d_j|}{max(d_i, d_j)} \tag{1}$$

where $|d_i, d_j|$ is a number of common features in the documents d_i and d_j, $|d_i|$ denotes a number of features in a document d_i,

$$dist(d_i, d_j) = \frac{|xOR(d_i, d_j)|}{max(d_i, d_j)} \tag{2}$$

where $|xOR(d_i, d_j)|$ is a number of distinct features in the documents d_i and d_j, $|d_i|$ denotes a number of features in a document d_i,

$$dist(d_i, d_j) = \frac{|xOR(d_i, d_j)|}{n} \tag{3}$$

which is a Hamming distance, where $|xOR(d_i, d_j)|$ is a number of distinct features in the documents d_i and d_j, n is a length of a feature vector.

It is also possible to calculate Euclidean distance especially, when comparing a given feature vector and a cluster prototype calculated as an arithmetic mean.

As it was stated in section 2 in order to define clusters of XML documents it is needed to compare the calculated signatures of the clusters and a query path.

Calculating a cluster signature is very simple when a structure of XML document is encoded as a string of bits. It is needed to perform OR operation on the clustered feature vectors in order to calculate a signature which contains information about all the elements and attributes occurring in the documents creating a cluster. Calculation of the query signature should be performed by encoding all the features addressed by the query path according to the dictionary created during the document structures encoding. Verification whether the query path addresses any of the documents creating a cluster is performed by AND operation on the cluster and the query signatures.

A path-based bit encoding is used in a presented work. A path is defined as a sequence of elements starting from a root of XML document up to a given element or attribute. A query is tokenized to all the paths creating a query path. Next, all the paths are encoded according to the dictionary created during the documents encoding process. In this way a query signature which is compatible with a documents signature is received. An assumption was taken in the work presented that the query paths are fully defined and indicate all the elements starting from a root element up to a target node.

3.2. Clustering algorithms

There is a large number of clustering algorithms (Ester et al. 1996, Guha, Rastogi and Shim 1999, Han and Kamber 2004, Hand, Mannila and Smyth 2005, Jain, Murty and Flynn 1999) which can be applied to the task of clustering bit feature vectors and supporting proposed method of accelerating XML queries execution. Algorithms of different types like hierarchical algorithms, e.g. Complete or Single Link (Han and Kamber 2004, Hand, Mannila and Smyth 2005, Jain, Murty and Flynn 1999), partitional algorithms, e.g. Hard C-Means (Han and Kamber 2004, Hand, Mannila and Smyth 2005, Jain, Murty and Flynn 1999), algorithms dedicated to categorical data e.g. ROCK (Guha, Rastogi and Shim 1999) or density

based algorithms e.g. DBSCAN (Ester et al. 1996) can be used in the presented task. However, these algorithms are not dedicated to the data representing a structure of XML documents. They do not take under consideration the document's hierarchical structure of nested elements and they do not make use of the information whether a given feature is an element placed near to the root of a document or if it is a leaf node.

The algorithms mentioned above perform clustering in a full feature vector space. Bit encoding produces very long feature vectors what decreases a clustering quality (Kozielski 2006a, Liu et al. 2004). Another observation is that the queries which may be performed on XML documents do not traverse through the whole tree structure to the leaves very often. Especially, concerning document-centric XML documents (Bourret 2005) it is very probable that the query path will reach only few out of many levels in a document tree. It is worth trying to reduce the number of features which are analysed. It cannot be performed by any of the known methods (Kozielski 2006a, Liu et al. 2004) because they operate on the whole feature space and they do not differentiate features according to the document structure level. They can reduce therefore the accuracy of the model represented by the feature vectors.

It is also a common observation that the most general and therefore important information is enclosed nearby the root element concerning the structure of XML document. The features which are placed on the levels neighbouring a root element should have therefore a greater influence on the clustering results then the leaf nodes. This feature of XML documents is also not taken under consideration by the clustering algorithms mentioned above. There are approaches to clustering XML documents concerning their structure which take under consideration tree-like structure of XML documents and the significance of the features depending on their level in this structure (Flesca et al. 2004, Nayak). These algorithms however, introduce methods dedicated to XML structure on a level of calculating similarity between document structures. They do not operate on bit feature vectors encoding XML document structure which were shown to be very effective in the presented method of accelerating XML queries.

There was therefore a need to introduce a new clustering algorithm which would be dedicated to XML documents and which would address all the requirements which are not met by the clustering algorithms mentioned above. The new clustering algorithm giving promising results was called Multilevel Clustering of XML Documents (ML) (Kozielski 2007).

3.3. Multilevel clustering of XML documents

Multilevel Clustering of XML Documents (ML) (Kozielski 2007) is a method dedicated to XML documents. Multilevel approach starts clustering at a root level and continues the process at the following levels. In this way it differentiates features treating the elements placed in the neighbourhood of a root element as more significant. It is possible to stop the algorithm at a certain level of the document structure tree reducing a number of features which are processed. Defining a set of XML documents as $D = \{d_1, ..., d_N\}$ and a feature vector B encoding each doc-

ument as a string of bits as $B = \left\{ {b_1}^1, ..., {b_{n1}}^1, ..., {b_1}^l, ..., {b_{nl}}^l \right\}$ where $l = 1, ..., L_T$ is a number of a level at which occurs a given feature. A hard clustering result on a level l is a partition of a set D to a set of clusters $C^l = \left\{ {C_1}^l, ..., {C_{K_l}}^l \right\}$, where K_l is a number of clusters determined on a level l, $\bigcup_{i=1}^{K_l} {C_i}^l = D$, and ${C_i}^l \cap {C_j}^l = \Phi$, where $i \neq j$. Each cluster ${C_i}^l$ may be partitioned on a level $l+1$ giving a set of new clusters. A final clustering result is a set of clusters $C = C^L$, where L is a level of XML structure tree which is defined by a user as a stop condition. The other input parameters are a user defined final number of clusters K_L and a distribution of the features among the document structure levels. *ML* algorithm proceeds in simplification according to the following steps:

```
starting from a root level (l = 1)
for ( each consecutive level l of the document structure trees )
    for ( each cluster C_i^l - 1 determined on a previous level )
        determine a new partition {C_1^l, ..., C_m^l}
        if ( stop condition = K_L )
            break
    if ( stop condition = L)
        break
```

Clustering on each level can be performed by means of any clustering algorithm. Different approaches to multilevel clustering of XML documents are presented in the following sections.

3.3.1. Multilevel hard partitional clustering of XML documents

Hard partitional clustering algorithm e.g. *Hard C-Means* (Han and Kamber 2004, Hand, Mannila and Smyth 2005, Jain, Murty and Flynn 1999) may be applied to *ML* method as an algorithm determining a partition on each structure level. *Hard C-Means* takes a number of clusters which must be determined as an input parameter. Clustering process iteratively minimizes the following criterion function:

$$Q = \sum_{i=1}^{c} \sum_{k=1}^{N} u_{ik} ||x_k - v_i||^2 \tag{4}$$

where $V = [v_1, v_2, ..., v_c]$ is a matrix consisting of cluster's prototypes, $U = [u_{ik}]$ is a partition matrix, c is a number of clusters and N is a number of data objects.

$$u_{ik} = \begin{cases} 1, & x_k \in C_i \\ 0, & x_k \notin C_i \end{cases} \tag{5}$$

As a result of clustering process partition matrix U and a matrix of cluster's centers V are returned.

Application of *HCM* algorithm to *ML* method will be further called as MLHCM algorithm. It requires from a user to define a number of clusters which should be determined on each structure level. It is possible however, to predict the number of clusters which should be determined on each level by analysis of variance. In this

case a user needs to set a total number of clusters K_L which should be produced by *ML* algorithm. Analysing a variance of the feature values on each structure level l it is possible to calculate a number of clusters K_l which should be determined on this level so that:

$$\sum_{l=1}^{L} K_l = K_L \tag{6}$$

Applying hard clustering to *ML* method a partition, once created, is transferred to the lower structure levels. Therefore, partitioning on a one structure level determines directly partitions on the next levels.

3.3.2. Multilevel density based clustering of XML documents

Another clustering approach which may be applied to *ML* method is density based clustering. Density Based Spatial Clustering of Application with Noise (*DBSCAN*) (Ester et al. 1996) algorithm defines a cluster by means of notions of neighbourhood and density (Ester et al. 1996). The input parameters of the algorithm are a minimal distance between two points defining if they are the neighbours and a minimal number of density-reachable points creating a cluster (Ester et al. 1996). *DBSCAN* algorithm determines a hard partition of a dataset for dense clusters and a set of data objects treated as noise. In case of the considered application of accelerating queries on XML documents these features are very important. The clusters which are created are consistent and therefore well fitted to the queries. The data objects marked as noise may be assigned to a cluster marked as "others" which will be probably addressed by most of the queries. Another advantage of *DBSCAN* algorithm is the fact that a number of clusters is not required as an input parameter.

When *DBSCAN* is applied to *ML* approach (MLDBSCAN) it finds at the first level as many dens clusters as possible. On the following structure levels *MLDBSCAN* looks for the dens regions among previously created clusters. In this way *MLDBSCAN* may be compared to *SUBCLU* (Kailing, Kröger and Kriegel 2004) clustering algorithm which performs density based clustering in subspaces of the feature vectors. In case of *MLDBSCAN* however, subspaces are defined by a hierarchy of XML document structure.

3.3.3. Multilevel conditional fuzzy clustering of XML documents

Another interesting algorithm which may be applied to *ML* method is *Conditional Fuzzy C-Means* algorithm which was introduced in (Pedrycz 1996) and generalised in (Łęski 2003). It extends *Fuzzy C-Means* (Łęski 2003, Pedrycz 1996) algorithm by introducing conditional variable f_k. Conditional variable f_k specifies what is the impact of a data object x_k on the created partition. Therefore, the set of all possible fuzzy partitions of N data objects into c clusters is defined as:

$$\tilde{U} = \left\{ u_{ik} \in [0,1] | \sum_{i=1}^{c} u_{ik} = f_k \quad \forall k, \quad 0 < \sum_{k=1}^{N} u_{ik} < N \quad \forall i \right\} \tag{7}$$

The criterion function describing a quality of a partition is defined as:

$$Q = \sum_{i=1}^{c} \sum_{k=1}^{N} {u_{ik}}^{p} ||x_k - v_i||^2 \tag{8}$$

where $||x_k - v_i||$ denotes a distance between a data object x_k and a cluster prototype v_i, $p > 1$ is a parameter influencing a fuzziness of the clusters, $U = [u_{ik}]$, where $U \in \tilde{U}$, is a partition matrix defined as:

$$u_{ik} = f_k \cdot \sum_{j=1}^{c} \left(\frac{||x_k - v_i||}{||x_k - v_j||} \right)^{\frac{-2}{p-1}} \tag{9}$$

The prototypes of the clusters are calculated according to the following formula:

$$v_i = \frac{\sum_{k=1}^{N} (u_{ik})^p x_k}{\sum_{k=1}^{N} (u_{ik})^p} \tag{10}$$

The result of an algorithm is a pair of iteratively modified matrices: a fuzzy partition matrix $U = [u_{ik}]$ defining what is the membership value of each data object x_k to each cluster C_i and a prototype matrix $V = [v_i]$.
In order to apply *CFCM* algorithm to *ML* method (*MLCFCM* algorithm) it is needed to modify the definitions presented in section 3.3 which were applicable to approaches presented in sections 3.3.1 and 3.3.2. The clustering results at the level l defined for hard clustering must be modified and defined as a fuzzy partition having a form of a matrix $U^l = [{u_{ik}}^l]$ of size $K_l \times N$, where K_l was defined as a number of clusters determined on a level l, N is a number of all documents which are analysed. The created clusters are not separated and partitioned in the consecutive iterations of the algorithm as it was performed when hard clustering was used. The partition matrix U^{l-1} calculated on a previous level $l-1$ of the document structure trees becomes a bases of condition matrix ${F_{K_l}}^l$ containing the values of condition parameter f_k used in *CFCM* algorithm. Condition parameters impact a new fuzzy partition U^l on a level l. Condition matrix ${F_{K_l}}^l = g(U^{l-1})$ may be calculated by means of different forms of function g. A final partition U^L is binarised in the following way:

$$u_{ik} = \begin{cases} 1, & u_{ik} = max_{j=1..K_L}(u_{jk}) \\ 0, & u_{ik} \neq max_{j=1..K_L}(u_{jk}) \end{cases} \tag{11}$$

what gives a hard partition determining a set of clusters C^L as defined in section 3.3.
The approach presented above ensures that clustering on a level closer to a root element will influence the partition of the features placed further in the XML document tree. At the same time however, a direct transfer of cluster borders, which is performed when hard clustering is used, is avoided.
Another feature of *MLCFCM* algorithm is its ability to detect the documents having a strongly different structure comparing to the prototypes of the clusters. Concerning the application of the proposed algorithm, which is acceleration of

the queries on XML datasets, it is advantageous to the results to receive as compact clusters as possible. A document having a strongly different structure from the others may distort the clustering results when assigned by hard to any cluster. The characteristic feature of the documents of this type is a very little variance of the membership values in a partition matrix U^l. The proposed *MLCFCM* algorithm assigns documents strongly distinct from the cluster prototypes on each level to the "others" cluster determining therefore, more compact clusters.

4. Experiments

4.1. Datasets

Several experiments were performed on two sets of XML documents having different structure which are characterised as follows.

INEX dataset consists of 1500 XML documents being a randomly chosen subset of a set of document-centric (Bourret 2005) XML documents containing articles and other conference information like e.g. call for papers and erratum (XML data mining 2006). Encoding the documents creating *INEX* dataset gave a feature vector containing 5365 bits distributed among 20 levels of XML document structure.

Wiki dataset consists of 989 randomly chosen XML documents coming from a larger extract of Wikipedia to XML format (XML data mining 2006). The XML documents creating this dataset are document-centric XML documents. Encoding the documents creating *Wiki* dataset gave a feature vector containing 6557 bits distributed among 36 levels of XML document structure.

As it was presented in the section 2, it is assumed that a query should be executed faster on a reduced set of documents containing all the documents addressed by the query. Therefore, several selective queries were defined for each dataset. The algorithms were compared according to a dataset reduction degree which was received in the experiments for a given query.

The characteristics of the queries defined on documents from *Wiki* and *INEX* datasets are presented in tables 1 and 2.

Table 1. Characteristics of the queries defined on *INEX* dataset

Query	Query path	Path length	Documents addressed by a query	Documents which may be reduced
q1	/article/fm/abs	3	876	624
q2	/article/fm/kwd	3	441	1059
q3	/article/bdy/sec	3	1472	28
q4	/article/bdy/sec/st	4	1441	59
q5	/article/bm/bib/bibl/bb	5	881	619

Table 2. Characteristics of the queries defined on *Wiki* dataset

Query	Query path	Path length	Documents addressed by a query	Documents which may be reduced
q1	/article/body/section/title	4	620	369
q2	/article/body/definitionlist	3	3	986
q3	/article/body/normallist	3	56	933
q4	/article/body/figure	3	155	834

4.2. Results

The results of datasets reduction for different queries are presented in the tables 3 and 4. Different applications of *ML* method were compared in the experiments (*MLCFCM*, *MLHCM*, *MLDBSCAN*) with each other and with *HCM* algorithm clustering the whole feature space. Number of clusters which should be created and a structure level (depth) of XML documents which should be reached were used as input parameters of *ML* algorithm. *Wiki* dataset was partitioned into 5 clusters whereas *INEX* dataset was partitioned into 10 clusters. The query paths which were defined for the datasets are 3, 4 or 5 nodes long. It was decided therefore, that the depth of the analysis should be set to 3 and 4 structure levels.

Table 3. Number of documents reduced for selective queries on *INEX* dataset

Query	MLCFCM		MLHCM		MLDBSCAN		HCM
	Level 3	Level 4	Level 3	Level 4	Level 3	Level 4	
q1	0	0	21	0	117	117	0
q2	912	520	881	698	272	188	796
q3	0	0	0	0	0	0	0
q4	0	0	0	0	0	0	0
q5	310	262	263	263	117	117	0

Table 4. Number of documents reduced for selective queries on *Wiki* dataset

Query	MLCFCM		MLHCM		MLDBSCAN		HCM
	Level 3	Level 4	Level 3	Level 4	Level 3	Level 4	
q1	0	0	355	319	31	43	0
q2	553	508	347	211	372	206	329
q3	93	0	14	0	372	206	5
q4	13	127	0	0	315	199	272

The results which are presented in tables 3 and 4 show that the implementations of *ML* algorithm performed better then *HCM* algorithm in most cases. The largest numbers of the reduced documents were received for *MLCFCM* algorithm. It may be also noticed that *MLDBSCAN* algorithm is able to determine clusters containing documents which are not addressed by the queries and which may be reduced for most of the query paths which were defined in the analysis. The other algorithms performed worse considering this aspect. The results show that the application of clustering algorithms to accelerating XML query execution gives much better results when the analysed queries are strongly selective and there is a large number of documents which may be reduced for a given query.

It must be also emphasized that *ML* algorithms performed clustering on a much smaller number of features then *HCM* algorithm what is presented in the table 5.

Table 5. Number of features analysed by different clustering algorithms

		INEX dataset	*Wiki* dataset
ML algorithms	Level 3	39	424
	Level 4	230	1930
HCM algorithm	Whole feature vector	5365	6557

5. Conclusions

A method of Multilevel Clustering of XML Documents (ML) and its application to accelerating query execution on XML documents is presented in the paper. Three different implementations taking advantage of different clustering algorithms are considered. The results of the experiments which are presented show that *ML* methods perform better then a typical partitional clustering algorithm represented by *HCM* algorithm. The values received depend on the clustering algorithm implemented in *ML* method. *MLCFCM* algorithm gave the best results concerning accelerating of a chosen query whereas *MLDBSCAN* was able to accelerate the largest number of queries. The experiments performed show also that *ML* approach may be regarded as an effective method of clustering XML documents in subspaces of feature vector defined by structural hierarchy.

References

Bourret R. (2005) XML and Databases. `http://www.rpbourret.com/xml/~XMLAndDatabases.htm`.

Ester M. et al. (1996) A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. *Proc. 2nd Int. Conf. on Knowledge Discovery and Data Mining (KDD'96)* 226-231.

Flesca S. et al. (2004) Fast Detection of XML Structural Similarity. *IEEE Transactions on Knowledge and Data Engineering* **17**, 2, 169–175.

Guha S., Rastogi R. and Shim K. (1999) ROCK: A Robust Clustering Algorithm for Categorical Attributes. *15th International Conference on Data Engineering (ICDE'99).*

Han J. and Kamber M. (2004) *Data Mining: Concepts and Techniques.* Morgan Kaufmann Publishers, Academic Press, San Francisco.

Hand D., Mannila H. and Smyth P. (2005) *Eksploracja danych (Principles of Data Mining).* WNT, Warsaw.

Jain A. K., Murty M. N. and Flynn P. J. (1999) Data Clustering: A review. *ACM Computing Surveys* **31**, 3, 264-323.

Kailing K., Kröger P. and Kriegel H.-P. (2004) Density- Connected Subspace Clustering for High-Dimensional Data. *Proc. SIAM Int. Conf. on Data Mining (SDM'04)* 246-257.

Kozielski M. (2007) Przyspieszanie realizacji zapytań na dokumentach XML z wykorzystaniem grupowania względem ich struktury. *Bazy Danych, Nowe Technologie: Architektura, metody formalne i zaawansowana analiza danych.* WKŁ 305-314.

Kozielski M. (2006a) Improving the Results and Performance of Clustering Bit-encoded XML Documents. *Proc. of ICDM Workshops 2006, IEEE Computer Society Press* 60-64.

Lian W. et al. (2004) An Efficient and Scalable Algorithm for Clustering XML Documents by Structure. *IEEE Transactions on Knowledge and Data Engineering* **16**, 1, 82–96.

Liu J. et al. (2004) XML Clustering by Principal Component Analysis. *Proceedings of the 16th IEEE International Conference on Tools with Artificial Intelligence, (ICTAI 2004).*

Łęski J. (2003) Generalized Weighted Conditional Fuzzy Clustering. *IEEE Transactions on Fuzzy Systems* **11**, 6, 1–7.

Nayak R. Fast and Effective Clustering of XML Data Utilizing their Structural Information. (Under publication in *KAIS: Knowledge and Information Systems - An International Journal*).

Pedrycz W. (1996) Conditional Fuzzy C-Means. *Pattern Recognition Letters* **17**, 625-631.

XML data mining (2006) `http://xmlmining.lip6.fr`.

Yoon J. P., Raghavan V. and Chakilam V. (2001) Bitmap Indexing-based Clustering and Retrieval of XML Documents. *Proceedings of ACM SIGIR Workshop on Mathematical/Formal Methods in Information Retrieval.*