STUDIA INFORMATICA 2009
Volume 30 Number 2A (83)

Michał KOZIELSKI
Politechnika Śląska, Instytut Informatyki

CLUSTERING COLLECTIONS OF XML DOCUMENTS HAVING DIFFERENT STRUCTURE TYPES

Summary. The paper presents comparison of application of several clustering algorithms and XML structure encoding methods to clustering XML documents having different structure types. Quality of the clustering is evaluated regarding the application of the resulting partitions to acceleration of the selective queries execution on XML collections. The results show that application of multilevel clustering algorithm to analysis of XML documents having complex structure gives the partition of better quality.

Keywords: clustering, XML documents clustering

GRUPOWANIE KOLEKCJI DOKUMENTÓW XML O RÓŻNYCH TYPACH STRUKTURY

Streszczenie. Praca przedstawia porównanie zastosowania różnych algorytmów grupowania i kodowania do analizy dokumentów XML o różnym typie struktury. Jakość grupowania jest oceniana względem zastosowania uzyskanego podziału do przyspieszania realizacji zapytań selektywnych na kolekcji dokumentów XML. Otrzymane wyniki pokazują, że zastosowanie metody grupowania wielopoziomowego do analizy dokumentów XML o złożonej strukturze daje podział na grupy o lepszej jakości w porównaniu do tradycyjnych metod grupowania.

Słowa kluczowe: grupowanie, grupowanie dokumentów XML

1. Introduction

XML (eXtensible Markup Language) [4] has become a popular and commonly used standard for data representation and interchange. A large number of XML documents which are used and the existence of powerful database systems enabling storage of such documents

make the application of data mining algorithms to XML documents more and more interesting and intensively developed [6].

One of many data mining algorithms which may be used in order to analyse XML documents is clustering. The methods clustering XML documents considering their structure are presented in this paper.

Structure of XML documents may be compared using e.g. either tree edit distance [17] or level similarity [16] or one of the methods encoding a tree-like structure of XML document into a form of a feature vector [5, 8, 14, 20].

Using a feature vector representation of XML document structure it is possible to use one of the classical clustering algorithms known from literature [9, 10, 11] or the Multilevel clustering algorithm which was proposed in [12].

Analysing XML collections it is possible to distinguish two types of XML documents structure [3]:

- data-centric XML documents,
- document-centric XML documents.

Data-centric XML documents have generally regular structure and equal granulation, whereas document-centric XML documents have irregular structure and unequal granulation. The first type of documents may be created e.g. when exporting data from relational database system to XML format whereas the latter type of documents may be created e.g. when formatting text documents by adding XML tags. Knowing the characteristics of both types of XML structures it may be expected that the quality of clustering results can differ depending on the type of the documents collection.

This paper presents the results of clustering collections of XML documents having different structure type. Several clustering algorithms are verified, with a special focus on multilevel clustering algorithm proposed by the author, in connection with different methods encoding XML documents structure. Clustering results are compared according to the application of the calculated partition to accelerating the execution of different types of quantitative selective queries on XML collection.

The paper is organized as follows. In section 2 different types of methods encoding XML structure into a form of feature vectors are presented. Section 3 presents the clustering algorithms which were used in the experiments. Section 4 presents the experiments which were performed. The final conclusions are drawn in section 5.

2. Encoding XML documents structure

Analysing the methods which encode XML document structure into the form of feature vector it is possible to distinguish two types of approaches:

- flat structure encoding – the methods that encode only occurrences of elements and attributes but do not encode their position in a document tree,
- hierarchical structure encoding – the methods that encode the occurrences of elements and attributes and also their position (level) in a document tree.

Bit encoding [14, 20] may be presented as an example of flat structure encoding. Bit encoding 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 part of XML structure in the analysed document. A pair of bits [14] connected in a parent-child relation or a path [20] starting from a root element and leading to a chosen node may be taken as a part of structure. Considering a collection D of N XML documents it is possible to encode structure of each document as a vector $\boldsymbol{X}_k=[x_{ik}]$, where $i=1,...,n$, $k=1,...,N$, $n=\mathrm{Card}(\Gamma)$, where Γ is a dictionary of all the features existing in the analysed collection of XML documents.

When a number of occurrences of structure parts is important it is possible to modify bit encoding and use natural numbers instead of $\{0,1\}$. This type of method may be called *cardinality encoding*.

Both bit and cardinality encoding are easily interpretable and convenient methods of encoding XML document structure. However, a feature space produced by this type of encoding can be very large, especially in case of document-centric XML documents.

Distance between the feature vectors $\boldsymbol{X}_k$ and $\boldsymbol{X}_m$ received by means of cardinality encoding, which was used in the experiments presented in the further points, may be calculated e.g. by means of Euclidean distance:

$$d(X_k, X_m) = \sqrt{\sum_{i=1}^{n} (x_{ik} - x_{im})^2} \tag{1}$$

where n is a number of features.

Cardinality and bit encoding do not encode a level on which a given feature is placed in a document. It is however possible to acquire the level information during the encoding process and use it later during the clustering process.

Fuzzy encoding [5] may be presented as an example of hierarchical structure encoding. Fuzzy encoding defines a structure of XML document as a fuzzy bag [19]. Each feature is represented by its cardinality and a membership value calculated on the basis of the feature's level in XML document structure according to the following formula:

$$\mu = \frac{\sum_{l=0}^{L-1} l! \xi_{l+1}}{L!} \tag{2}$$

where L is a level in XML document structure where a given node is placed, ξ is a node importance value which can be provided by an expert. Formula (2) enables to calculate different membership values considering a node position in a document structure. The closer to the document's root element the feature is placed the larger is its membership value.

Similarity of the fuzzy bags X_k and X_m received by means of fuzzy encoding may be calculated according to the following formula [2, 5]:

$$S(X_k, X_m) = \frac{M(X_k \cap X_m)}{M(X_k \cup X_m)} \tag{3}$$

where fuzzy measure M on fuzzy bag X_k having values from a space $\mathbf{X}_k$, is defined as [2]:

$$M(X_k) = \sum_{x \in X_k} \mu_{X_k}(x) \tag{4}$$

Fuzzy encoding is a more complex method comparing to cardinality encoding but it encodes a structure level hierarchy into a feature vector. A more detailed explanation of fuzzy encoding method may be found in [5].

3. Clustering XML documents structure

Analysing the methods which may be applied to clustering XML documents structure it is possible to distinguish two types of approaches:

- traditional algorithms which do not take under consideration additional explicit information about features position in hierarchical XML document structure and cluster feature vectors analysing the whole feature space,
- multilevel algorithm which performs clustering on consecutive levels of XML documents feature space.

Traditional methods like hierarchical algorithm complete link [9, 10, 11], partitional algorithm c-means [9, 10, 11] or density based algorithm DBSCAN [7] belong to the first of the mentioned classes of algorithms.

The algorithm called *Multilevel clustering of XML documents* (ML) [12, 13], opposite to traditional algorithms, uses an explicit information about the features position in XML document structure as a parameter. 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 then leaves because clustering on one structure level influences the clustering process on the following

levels. It is possible to stop clustering at a given structure level reducing in that way the analysed feature space.

Multilevel approach may use any traditional clustering algorithm in order to perform clustering on a given structure level. An interesting implementation is the one using *Conditional Fuzzy C-Means* [15, 18] and called *Multilevel Conditional Fuzzy C-Means* (MLCFCM) [12].

Conditional Fuzzy C-Means algorithm assumes that different data objects $\boldsymbol{X}_k$ may have different impact on a resulting partition. This impact is determined by a value of condition f_k connected with each data object $\boldsymbol{X}_k$. In this case all the possible partition matrices which may be calculated for N data objects clustered into c clusters may be defined as:

$$\widetilde{U} = \left\{ u_{ik} \in [0,1] \mid \forall k \ \sum_{i=1}^{c} u_{ik} = f_k, \ \forall i \ \ 0 < \sum_{k=1}^{N} u_{ik} < N \right\} \tag{5}$$

Clustering XML documents by means of MLCFCM algorithm it is possible to use a fuzzy partition matrix $\boldsymbol{U}^l=[u_{ik}]$ calculated on a structure level l as a basis for condition matrix $\boldsymbol{F}^{l+1}=[f_k]$ used as a parameter for clustering on a structure level $l+1$.

In this way it is possible to introduce an influence of a partition received on a previous level (l) to partitioning on the next level ($l+1$).

4. Experiments

The experiments performed aimed to compare a combination of the methods belonging to the classes presented in the previous points. The following two combinations of methods were compared:

- clustering data by means of the algorithm which does not take under consideration a hierarchical structure of XML documents, where the data are feature vectors which are the result of hierarchical structure encoding method,
- clustering data by means of the algorithm which takes under consideration a hierarchical structure of XML documents, where the data are feature vectors which are the result of flat structure encoding method.

In the first case Complete Link and DBSCAN clustering algorithms were used. Clustering was performed analysing the feature vectors calculated by means of fuzzy encoding.

In the latter case the multilevel clustering algorithm (MLCFCM) was used. Clustering was performed analysing the feature vectors calculated by means of cardinality encoding.

4.1. Datasets

The experiments were performed on two collections of XML documents having different structure which are characterized below. The description is completed by the number of features received when cardinality encoding was used. It shows the difference in feature space of data-centric and document-centric XML documents.

Bio dataset [1] consists of 1500 XML documents being a randomly chosen subset of a set of data-centric XML documents where each document contains information about one protein sequence e.g. organism name, taxonomy, references to other databases. Cardinality encoding of the documents creating *Bio* dataset gave a feature vector containing 134 features distributed among 6 levels of XML document structure.

INEX dataset [6] consists of 1500 XML documents being a randomly chosen subset of a set of document-centric XML documents containing articles and other conference information like e.g. call for papers and erratum. Cardinality encoding of the documents creating *INEX* dataset gave a feature vector containing 5365 features distributed among 20 levels of XML document structure.

Using collections of XML documents having different structure type aimed to show that structure type is not meaningless when clustering XML documents.

4.2. Clustering quality measure

The partitions received in clustering process were applied to accelerating XML query execution. Therefore, the quality of the clustering performed was measured in context of this application.

The idea of accelerating XML query execution using clustering results is based on the following concept. The structure of XML documents stored in a database is flexible what means that occurrence of an element or attribute may be optional. Therefore, it is possible that not all the documents in a collection will match a path specified in a query. Such a query may be called *selective query*. Assuming that an execution of a selective query on a subset of documents is less time consuming then analysing the whole collection it is worth verifying the methods which could determine the collection subsets addressing the given selective queries. Clustering algorithms can be applied to this task because they produce a set of clusters of documents having more similar structure within the cluster then between the clusters. Having a cluster of documents it is possible to calculate a signature of the cluster. Such signature represents all the features (elements and attributes) existing in the cluster. It is also possible to calculate a signature of a query which represents all the features which are addressed by this query. Comparison of both signatures (of a cluster and a query) shows if the

query addresses any document in a cluster and if the given cluster of documents should be processed by the query. The method described may be summed up in the following points:

- Cluster the documents considering 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.

Two different types of selective queries may be distinguished:

- qualitative selective queries which consider only the occurrence of an element or an attribute, e.g.:

```
/entry/reference/citation
```

- quantitative selective queries which consider the number of occurrences of an element or an attribute, e.g.:

```
/entry[count(reference/citation)>2]
```

In the work presented quantitative selective queries were analysed. For each collection of documents a set of queries in the form of path expressions was defined. The analysed query paths are presented in tables 1 and 2.

Table 1

Examples of query paths defined for *Bio* collection

No.	Path	Cardinality variance
1	/entry/accession	12.4
2	/entry/reference/citation	19.5
3	/entry/dbReference/property	937.93
4	/entry/feature	345

Table 2

Examples of query paths defined for *INEX* collection

No.	Path	Cardinality variance
1	/article/body/sec	7.6
2	/article/body/sec/ss1	34
3	/article/bin/bib/bibl/bb	250.1
4	/article/fm/au/sum	2.27

Quantity conditions were defined for each query path from tables 1 and 2. The conditions were defined on the basis of cardinality histogram showing what is a number of documents containing the given number of the analysed path.

A typical cardinality histogram of the analysed query paths is presented on fig. 1. There are three values marked on fig. 1: x refers to medoid, x_1 and x_2 are the values placed around x.

The values x, x_1 and x_2 determined for each path presented in tables 1 and 2 are presented in tables 3 and 4 respectively.

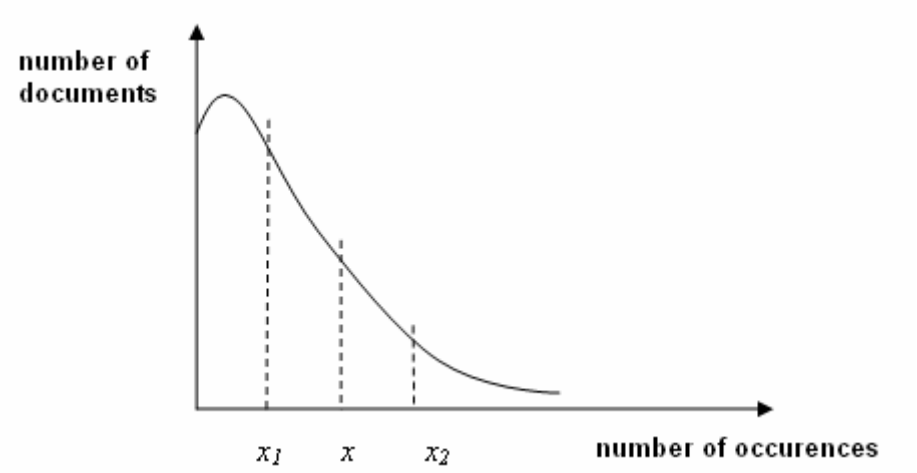

Fig. 1. Typical cardinality histogram showing a number of documents containing the given occurrence number of the analyzed path

Rys. 1. Typowy histogram krotności przedstawiający liczbę dokumentów zawierających daną liczbę wystąpień analizowanych ścieżek

Table 3

Values x, x_1 and x_2 determined for queries on *Bio* collection from tab. 1

No.	x	x_1	x_2
1	4	2	6
2	4	2	6
3	15	10	20
4	5	3	7

Table 4

Values x, x1 and x2 determined for queries on INEX collection from tab. 2

No.	x	x_1	x_2
1	6	4	8
2	8	4	10
3	20	10	30
4	3	2	4

Quantity condition of the queries were determined by the cardinality histogram (fig. 1) and the values x, x_1 and x_2. The following conditions were defined for each query path p defined in tables 1 and 2:

- p_1^1: $\text{count}(p) \leq x$,
- p_1^2: $\text{count}(p) > x$,
- p_2^1: $\text{count}(p) \leq x_1$,
- p_2^2: $\text{count}(p) > x_2$.

In this way each path p defines four queries p_i^j.

Number of documents which do not have to be analysed by such p_i^j query on collection *Bio* or *INEX* is considered as a quality measure of a partition received in clustering process.

4.3. Results

Algorithms applied in the work presented used the following parameters.

All the clustering algorithms that use a number of clusters to be created c as a parameter used a value of c=10. DBSCAN algorithm used parameters of the following values:

- $\varepsilon = 0.15$; m=10 for *Bio* dataset,
- $\varepsilon = 0.325$; m=10 for *INEX* dataset.

Multilevel clustering algorithm performs analysis on the number of structure levels L which is given as a parameter. In the experiment performed two values of the parameter were used: L=3 and L=4. The applied L values stem from the average number of nodes creating the query paths used in the analysis.

The experiment results are presented in tables 5 for *Bio* collection and 6 for *INEX* collection. The results are expressed in the form of numbers of documents which do not have to be analysed during a selective query execution when a given partition was used. The larger number of documents was reduced the better are the results.

Table 5

Number of documents which do not have to be analysed when a p_i^j query is executed on *Bio* collection

Clustering algorithm	Average no. of documents reduced			
	Query p_1^1	Query p_1^2	Query p_2^1	Query p_2^2
Complete Link	67.1	218.8	80.9	347.7
DBSCAN	14.2	325.8	50.9	361
MLCFCM (L=3)	92	2.6	190.7	181.8
MLCFCM (L=4)	73.7	53.1	189	328.7

Table 6

Number of documents which do not have to be analysed when a p_i^j query is executed on *INEX* collection

Clustering algorithm	Average no. of documents reduced			
	Query p_1^1	Query p_1^2	Query p_2^1	Query p_2^2
Complete Link	1	100	1	235.4
DBSCAN	4.2	82	16.2	130.2
MLCFCM (L=3)	52.2	235	85.6	382
MLCFCM (L=4)	0	90.6	1.4	272

The results presented in tables 5 and 6 may be analysed in two ways:

- considering the structure type of the XML documents collection,
- considering the type p_i^j queries resulting from a quantity condition.

Considering the structure type of the XML documents collection the values presented may be aggregated per dataset in order to present it in a more readable form. Average values calculated on the basis of tables 5 and 6 are presented in table 7 and illustrated on fig. 2.

Table 7

Average number of documents which do not have to be analysed when a query path is executed on *Bio* and *INEX* collection

Clustering algorithm	Average no. of documents reduced	
Complete Link	178.61	84.35
DBSCAN	187.97	58.15
MLCFCM (*L*=3)	116.75	188.7
MLCFCM (*L*=4)	161.11	91

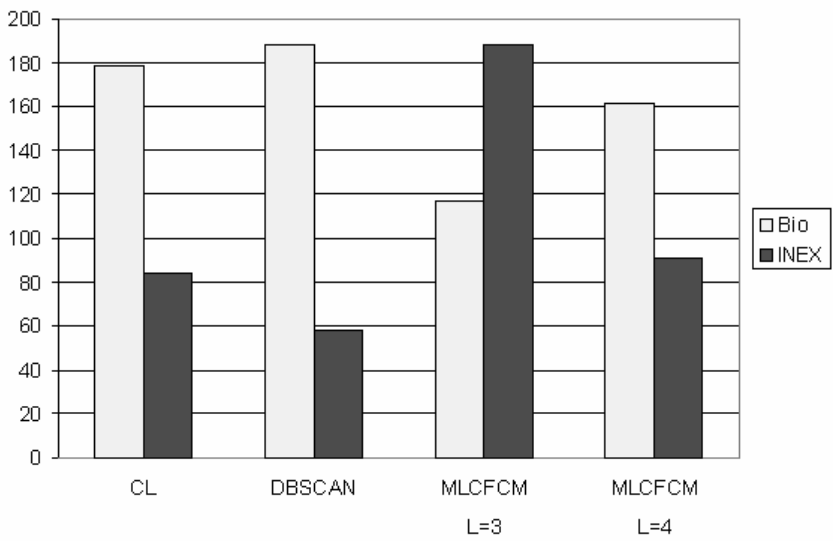

Fig. 2. Average number of documents which do not have to be analysed when a query path is executed on *Bio* and *INEX* collection

Rys. 2. Średnia liczba dokumentów, które nie muszą być analizowane przy realizacji zapytania na kolekcjach *Bio* i *INEX*

The results presented show that traditional clustering algorithms (Complete Link and DBSCAN) performed better on *Bio* collection whereas multilevel clustering algorithm performed better on *INEX* collection. It means that traditional clustering algorithms give better results on the collection of data-centric XML documents having a relatively simple structure and giving a relatively small feature space. In case of the collection of document-centric XML documents having a complex structure and giving a large feature space multilevel clustering algorithm gives better results.

It is also possible to notice that multilevel clustering algorithm is less sensitive for a type of the analysed XML documents structure. This fact may be illustrated by the comparison of average number of documents (aggregated per algorithm type) which do not have to be analysed during a query execution what is presented on fig. 3.

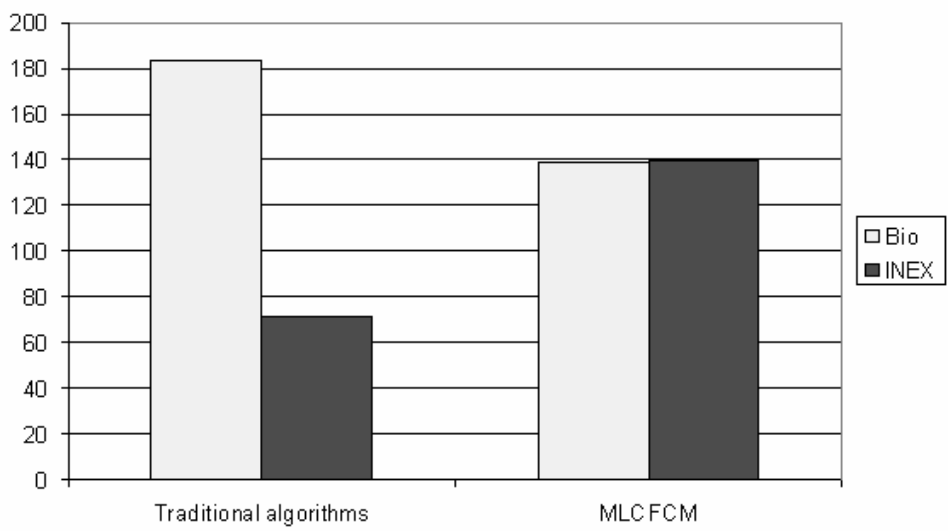

Fig. 3. Average number of documents which may be reduced during a query execution
Rys. 3. Średnia liczba dokumentów, które mogą być pominięte przy realizacji zapytania

Analysing the results in the context of the type of p_i^j queries it is possible to aggregate the values from the tables 5 and 6 per query type what is presented in table 8 and illustrated on fig. 4.

Table 8

Average number of documents which do not have to be analysed per p_i^j query type

Clustering algorithm	Average no. of documents reduced			
	Query p_1^1	Query p_1^2	Query p_2^1	Query p_2^2
Traditional algorithms	21.6	181.6	37.2	268.6
MLCFCM	54.4	95.3	116.7	291.1

The results presented on fig. 4 show that multilevel clustering algorithm performs better considering most of the query types.

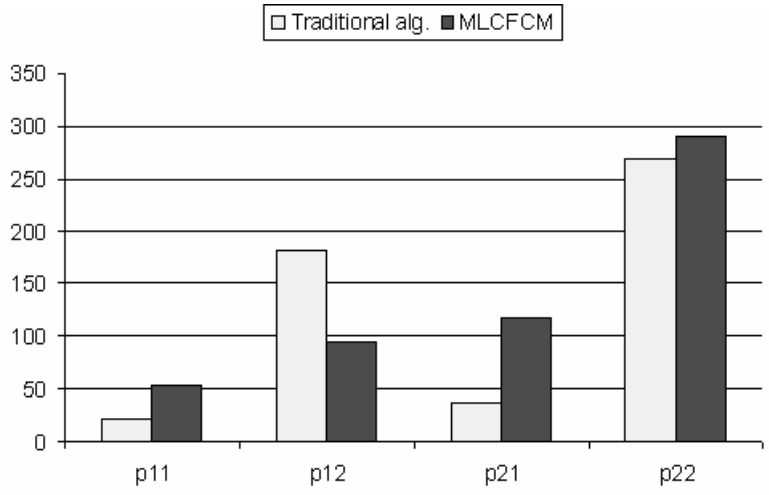

Fig. 4. Average number of documents which do not have to be analysed per p_i^j query type
Rys. 4. Średnia liczba dokumentów, które nie muszą być analizowane przy zapytaniu typu p_i^j

It is also possible to notice that the results reflect selectivity relations between the queries. Lets define selectivity of a query type p_i^j as $s(p_i^j)$. Analysing the cardinality histogram (fig. 1) it is possible to notice that selectivity relation is the following:

$$s(p_1^1) < s(p_2^1) < s(p_1^2) < s(p_2^2) \tag{6}$$

The less documents are addressed by a query the greater is the query selectivity. Thus, it is easier to reduce a greater number of documents for the more selective queries.

It is possible to notice that multilevel clustering algorithm performs significantly better considering the least selective query types p_1^1 and p_2^1.

Considering the query types defined in the experiment and their selectivity it is possible to write the following relation between the number of documents reduced:

$$r(p_1^1) < r(p_2^1) < r(p_1^2) < r(p_2^2) \tag{7}$$

where $r(p_i^j)$ is an average number of documents which do not have to be analysed when a query of p_i^j type is executed.

The relation (7) is reflected by the average results presented on fig. 4.

5. Conclusions

In this paper the analysis of the results of clustering XML documents having different structure types was presented. Combinations of several clustering algorithms and XML structure encoding methods were analysed. The partitions received were applied to acceleration of the selective queries execution on XML collections and compared with respect to this application.

The results show that the quality of clustering depends on the complexity of XML documents structure. Analysis of data-centric XML documents by means of traditional clustering algorithms gives relatively good results whereas document-centric XML documents should be analysed by the methods which are dedicated to XML complex structure like multilevel clustering algorithm.

It is also shown that multilevel clustering algorithm gives significantly better results for the least selective query types considering the analysed application of the clustering results.

BIBLIOGRAPHY

1. Bairoch A., Apweiler R., Wu C. H., Barker W. C., Boeckmann B., Ferro S., Gasteiger E., Huang H., Lopez R., Magrane M., Martin M. J., Natale D. A., O'Donovan C., Redaschi N., Yeh L. S.: The Universal Protein Resource (UniProt), Nucleic Acids Res. 33: D154-D159, http://www.uniprot.org/database/download.shtml (2005).
2. Bouchon-Meunier B., Rifqi M., Bothorel S.: Towards general measures of comparison of objects, Fuzzy Sets and Systems, 1996, Vol. 84, p. 143÷153.
3. Bourret R.: XML and Databases, http://www.rpbourret.com/xml/XMLAndDatabases-.htm, (2005).
4. Bray T., Paoli J., Sperberg-McQueen C. M., Maler E., Yergeau F. (ed.): Extensible Markup Language (XML) 1.0 (Fourth Edition), W3C Recommendation 16 August 2006, edited in place 29 September 2006, http://www.w3.org/TR/2006/REC-xml-20060816/ (20.12.2007).
5. Ceravolo P., Nocerino M. C., Viviani M.: Knowledge Extraction from Semi-structured Data Based on Fuzzy Techniques, Knowledge-Based Intelligent Information and Engineering Systems, Lecture Notes in Computer Science, 2004, Vol. 3215, p. 328÷334.
6. Denoyer L., Galliari P.: Dataset used in the experiment, http://xmlmining.lip6.fr (2006).
7. Ester M., Kriegel H. P., Sander J., Xu X.: 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), 1996, p. 226÷231.
8. Flesca S., Manco G., Masciari E., Pontieri L., Pugliese A.: Fast Detection of XML Structural Similarity, IEEE Transactions on Knowledge and Data Engineering, 2004, Vol. 17, No. 2, p. 160÷175.
9. Han J., Kamber M.: Data Mining: Concepts and Techniques, Morgan Kaufmann Publishers, Academic Press, San Francisco 2001.
10. Hand D., Mannila H., Smyth P.: Principles of Data Mining, WNT, Warszawa, 2005.
11. Jain A. K., Murty M. N., Flynn P. J.: Data Clustering: A review, ACM Computing Surveys, 1999, Vol. 31, No. 3, p. 264÷323.
12. Kozielski M.: Multilevel Conditional Fuzzy C-Means Clustering of XML Documents, Lecture Notes in Artificial Intelligence, Springer-Verlag, 2007, Vol. 4702, p. 532÷539.
13. Kozielski M.: Application of Different Clustering Algorithms to Multilevel Clustering of XML Documents, TPD 2007 Conference Proceedings, Wydawnictwo Politechniki Poznanskiej, 2007, p. 59÷70.

14. Lian W., Cheung D. W., Mamoulis N., Yiu A. M.: An Efficient and Scalable Algorithm for Clustering XML Documents by Structure, IEEE Transactions on Knowledge and Data Engineering, 2004, Vol. 16, No. 1, p. 82÷96.
15. Łęski J.: Generalized Weighted Conditional Fuzzy Clustering, IEEE Transactions on Fuzzy Systems, 2003, Vol. 11, No. 6, p. 1÷7.
16. Nayak R.: Fast and effective clustering of XML data using structural information, Knowl. Inf. Syst., 2008 , Vol. 14, No. 2, p. 197÷215.
17. Nierman A., Jagadish H. V.: Evaluating Structural Similarity in XML Documents, Fifth International Workshop on the Web and Databases (WebDB 2002), 2002.
18. Pedrycz W.: Conditional Fuzzy C-Means, Pattern Recognition Letters, Vol. 17, 1996, p. 625÷631.
19. Rocacher D.: On fuzzy bags and their application to flexible querying, Fuzzy Sets and Systems, 2003, Vol. 140, No. 1, p. 93÷110.
20. Yoon J. P., Raghavan V., Chakilam V.: Bitmap Indexing-based Clustering and Retrieval of XML Documents, Proceedings of ACM SIGIR Workshop on Mathematical/Formal Methods in Information Retrieval, 2001.

Recenzent: Dr hab. inż. Zbigniew Huzar, prof. Pol. Wrocławskiej

Wpłynęło do Redakcji 17 lutego 2009 r.

Omówienie

Popularność standardu XML umożliwiła zastosowanie algorytmów eksploracji danych do analizy dużych kolekcji dokumentów XML. Jedną z metod, która może być wykorzystana do analizy dokumentów XML, jest grupowanie dokumentów względem ich struktury. Zadanie to może zostać zrealizowane przez zastosowanie jednego ze znanych algorytmów grupowania (np. hierarchiczny algorytm typu Complete Link). Dokumenty XML są jednak danymi o drzewiastej, często bardzo złożonej, strukturze. Z tego względu warto zastosować do analizy takich danych metody grupowania dedykowane dla dokumentów XML (np. grupowanie wielopoziomowe). Niniejszy artykuł przedstawia porównanie wybranych algorytmów grupowania i kodowania struktury dokumentów XML zastosowanych do analizy dokumentów posiadających odmienne typy struktury. Do oceny jakości grupowania wykorzystywane jest zastosowanie otrzymanego podziału na grupy do przyspieszania realizacji zapytań selektywnych na kolekcji dokumentów XML. Otrzymane wyniki pokazują,

że zastosowanie algorytmu wielopoziomowego grupowania do dokumentów XML o złożonej strukturze daje podział lepszej jakości w porównaniu do tradycyjnych algorytmów grupowania. Grupowanie wielopoziomowe pozwala również na osiągnięcie lepszych wyników dla większości typów analizowanych zapytań selektywnych.

Address

Michał KOZIELSKI: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16,
44-100 Gliwice, Polska, michal.kozielski@polsl.pl.