International Journal of Computer Science and Business Informatics

IJCSBI.ORG

A New Online XML Document Clustering Based on XCLS++

Ahmad Khodayar E Qaramaleki
Department of Computer Engineering
Shabestar Islamic Azad University of Science and Technology
Shabestar, Iran

Hassan Naderi
Department of Computer Engineering
Iran University of Science and Technology
Resalat St., Tehran, Iran

ABSTRACT
Many methods have been proposed to XML document clustering. These methods can be divided into three categories: structure-based, content-based and hybrid methods. XCLS++ is one of the most effective and efficient algorithms to XML document clustering which fit into the structural clustering category. Because of its efficiency, XCLS++ can be used XML stream clustering. In this paper, we will show one of the weaknesses of this method and then we will try to solve it by deleting a factor in the XCLS++formula. As we will show, this factor is related to the node weight in a tree which represents a given XML document. According to our experimentations which have been presented in this paper, the effectiveness (in term of accuracy) and efficiency (in term of execution time) of XCLS++ can be improved once this weight factor is eliminated from the original XCLS++ formula.

Keywords
Clustering, XML documents, XCLS++, Structure similarity, Content similarity.

1. INTRODUCTION
One of the methods used to extract information from databases is data mining which is used in the search engines. The appropriate structure to store data in databases contributed to data mining. The easy execution of data mining in database causes increasing of search engine efficiency. So, using the proper structure in a database is a crucial point. One of the good ideas for using appropriate structure in database is clustering.

During past years, various formats like HTML and XHTML are presented for showing documents using the content and the structure. Because XML documents are used to transfer and to search in documents and also the usage of this technology is increased day by day, a good management on these documents is vital.

In general, XML document clustering methods can be divided into three categories: 1- Structural, 2-Content-basedand 3- Combination of structure and content (hybrid). In the structural approach, an important criterion in clustering is the structure of the document. In the other word, the structural similarity between documents is a criterion for placing them in the same class. In the content-based method, the criterion of clustering is the similarity between the texts of both documents, and finally in the hybrid approach, the content and the structure similarity together are criterions to clustering two documents.

XCLS++ is one of the structured based methods to XML document clustering [12]. It's an improved version of XCLS+ algorithm which is an efficient method for clustering XML documents [9]. Our studies showed that XCLS++ can be improved, because it has some problems which make it away from optimal output. In this paper, our focus is on XCLS++ method and new method has been proposed with better performance. In the next section, the related articles are investigated. In the section 3 the XCLS++method and in the Section 4 the problem of XCLS++ method are presented. In the Section 5 our new proposed method is explained and then it will be compared with the other algorithms in the next section. Execution times of the algorithms are compared in Section 7, and the last section is allocated to conclusion.

2. PREVIOUS WORKS

The criterion of clustering is based on the similarity of documents. As mentioned above, there are three ways to find the similarity of documents: 1- structural [2][5][10][11][12]which consider only the structure of the document2- content-based [9] which considers only the content and finally 3- hybrid (content with structure) [1][3][4][8] which consider both the content and the structure. As we know, each XML document can be transferred into a tree and then clustering operations can be done with those trees. Structural methods only consider the structure and do not pay attention to content. The XCLS++ is one of the most efficient online algorithms to XML document clustering that fit into the structural clustering category. Our focus in this paper is concentrated on this type of clustering. Based on our studies, we have seen some problems in the XCLS++formula, which makes it away from optimal value. The presented solution in this paper can solves the problem of XCLS++ and optimizes it. Details of this method and its evaluations will be presented in the next sections.

3. THE FORMULA OF XCLS++ METHOD

As previously mentioned XCLS++ method is an example of structural approach and it works based on the similarity between tag names. So it doesn't consider the content of XML documents. XCLS++ by make some changes on XCLS+ at two stages could solve existent drawback of XCLS+

[11]. But in this paper we will show that those changes are not enough to make optimization. Some trees which will be shown later will prove inefficiency of XCLS++ in some cases. In this paper we will try to eliminate these inefficacies. The operations of XCLS+ and XCLS++ are similar. In this way like the XCLS+ method the incoming XML document is compared to clustered documents. If the value of similarity is greater than are equal to a threshold (t), the XML document will be placed in the relevant cluster. Otherwise, new incoming document is clustered in a new cluster. This process is continued until the last document is entered. The similarity calculation is performed based on a formula. In what follow, the XCLS++ details will be explained and calculation of similarity value between two trees is shown. Then problem of XCLS++ has been presented. After that a solution to this problem will be introduced. The formula of XCLS++, which is an improved formulation of the XCLS+ method, is as follow:

$$\textbf{sim base on XCLS} + + = \frac{\mathbf{0.5} * \sum_{\mathbf{i=0}}^{\mathbf{l-1}} (\mathbf{CN_1^i} + \mathbf{CB_1^i} + \mathbf{CC_1^i}) * \mathbf{r^{l-i-1}} + \mathbf{0.5} * \sum_{\mathbf{i=0}}^{\mathbf{l-1}} (\mathbf{CN_1^j} + \mathbf{CB_1^j} + \mathbf{CC_1^j}) * \mathbf{r^{l-j-1}}}{(\sum_{\mathbf{k=0}}^{\mathbf{l-1}} \mathbf{N^k}) * \mathbf{z} + \mathbf{0.5} * \left(\sum_{\mathbf{i=0}}^{\mathbf{l-1}} (\mathbf{CB_1^i} + \mathbf{CC_1^i}) * \mathbf{r^{l-i-1}} + \sum_{\mathbf{j=0}}^{\mathbf{l-1}} (\mathbf{CB_1^j} + \mathbf{CC_1^j}) * \mathbf{r^{l-j-1}}\right)}$$

Formula 1. The formula of XCLS++ method

In the Formula1the value is between zero and one which is a positive feature of this formula. The variables used in this formula are:

1. Z is cluster size or in other words the number of documents within cluster.
2. CNi: is sum of incoming nodes which is ***similar*** to clustered document in level i.
3. CNj: is sum of incoming nodes which is ***similar*** to clustered document in level j.
4. CBi: is sum of incoming nodes which is ***similar&&samebrother*** to clustered document in level i.
5. CBj: is sum of incoming nodes which is ***similar&&samebrother*** to clustered document in level j.
6. CCi: is sum of incoming nodes which is ***similar&&samebrother&&samechild*** to clustered document in level i.
7. CCj: is sum of incoming nodes which is ***similar&&samebrother&&samechild*** to clustered document in level j.
8. l: is high of tree in the each document.
9. i, j:are related numbers of level.
10. r: is the incremental factor, which is considered number 2.
11. k: is equal to 2.
12. N: is sum of clustered nodes of related levels.

The algorithm of XCLS++ clustering method is as follow:

1- Represent a XML document by a relative tree.
2- Consider this tree with a tree related to a cluster.
3- Start to search same node in two trees from root node. If a node is found then do the calculation of the Formula1.Then go to step2, otherwise go to Step3.
4- If depth of trees moves toward the lower level in the both trees. Search the same node as step1. If there is same node, calculate the Formula1 and repeat step2, otherwise go to step3.
5- If depth of tree is (usually in clustered document), move toward down level in the clustered document and stay in the same level of the new incoming document. Search again the same nodes. If there is the same node, calculate the formula and then repeat step2, otherwise repeat step3.

3.1. An Example of XCLS++

For understanding the algorithm an example is given in this section. Our goal is to find the similarity between tree1 and tree2 based on the XCLS++ method. It should be noted that the tree1 referred to the incoming document and the tree2 referred to the clustered documents. Dotted arrows from left to right indicate the order of execution of algorithm steps. In this example for facility the variable values are calculated and placed on the dotted arrows

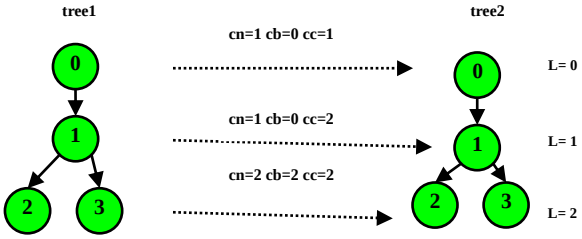

Figure 1. An example for showing work the XCLS+ method

After obtaining above factors, the similarity value base on the XCLS++ method will be equal to:

$$\frac{0.5 * \left((1+0+1)*2^2+(1+0+2)*2^1+(2+2+2)*2^0\right)+0.5*\left((1+0+1)*2^2+(1+0+2)*2^1+(2+2+2)*2^0\right)}{\left((1*2^2+1*2^1+2*2^0)*1\right)+0.5*(1*2^2+2*2^1+4*2^0)+0.5*(1*2^2+2*2^1+4*2^0)}$$

$$= 1$$

4. THE PROBLEM OF XCLS++

As was mentioned, XCLS++ method was presented to improve XCLS+ method. The primary cause of inefficiency XCLS+ is ignoring nodes repetitions in the original formula. This reason cases hierarchy of the nodes is changed. XCLS++ has been proposed to solve this problem. Dispute this improvement, studies show that XCLS++ algorithm has some problems too which can be further improved. In order to illustrate the problem, two examples are cited. The similarity value for the first example (Figure 2) calculated by XCLS++ method, is 1. Also the similarity value for in the

second example (Figure 3) with XCLS++ method is 1.4. Note that factors without parenthesis () on the dotted arrows belong to both trees.

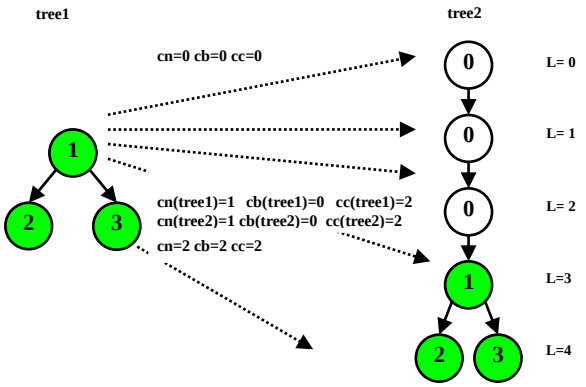

Figure 2. Same nodes of both trees are in quite different levels

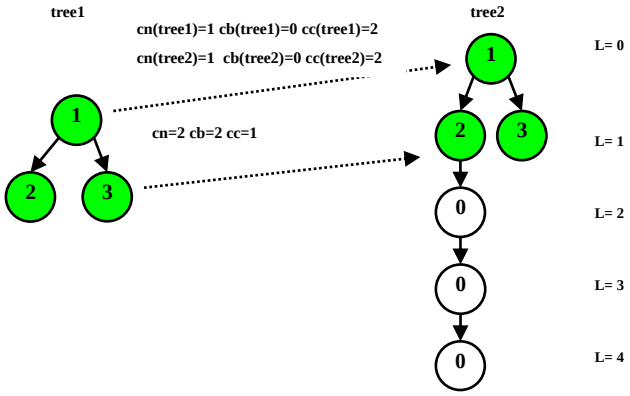

Figure 3. Same nodes of both trees are in the same upper levels

These results are unreal. Because similarity values in Figure 2 and Figure 3 which only differ in placed levels, must be closed together and less than one. But similarity values calculated with XCLS++ are very different. The reason of this difference is related to the weighting in XCLS++ formula. Our new formula is capable to solve this problem. In what follows, the proposed method is discussed in more details.

5. NEW METHOD: THE OTHER CHANGE ON THE XCLS++

The main drawback of XCLS++ method which is derived from XCLS+ and XCLS methods is its weighting factor. The weight causes high level nodes in a tree be hardly compared to low level nodes in another tree. So we must reduce the effect of the levels in order to minimize the difference between two such trees. A simple change can solve the problem of the formula

XCLS++. As previously mentioned, the main reason of this problem is due to the weighting factor. By removing the weight factor, this problem can be solved. So by deleting this factor the expected results can be earned. Finally, the proposed formula without weighting factor will be:

$$\textbf{sim base on XCLS} + + = \frac{\mathbf{0.5} * \sum_{i=0}^{l-1}(\mathbf{CN_1^i} + \mathbf{CB_1^i} + \mathbf{CC_1^i}) + \mathbf{0.5} * \sum_{i=0}^{l-1}(\mathbf{CN_1^j} + \mathbf{CB_1^j} + \mathbf{CC_1^j})}{(\sum_{k=0}^{l-1}\mathbf{N_k}) * \mathbf{z} + \mathbf{0.5} * (\sum_{i=0}^{l-1}(\mathbf{CB_1^i} + \mathbf{CC_1^i}) + \sum_{j=0}^{l-1}(\mathbf{CB_1^j} + \mathbf{CC_1^j}))}$$

Formula 2. The new formula to use in clustering XML documents

In this formula all variables are equivalent to variables of XCLS++. The new N^k parameter is the average of nodes clustered and by incoming trees. N of XCLS++ method causes results dependent to clustered or incoming tree and then unreal results. So for getting better results in the new method, N will be the average of nodes. For proving the effectiveness of this formula, the similarity the previous trees are calculated with new algorithm again. Results for two groups according new algorithm are: 0.85 for Figure 1 and 0.84 for Figure 2. These results have easily obtained with replacing variables in new formula. The calculated results show that new algorithm has more effectiveness and these results are near to reality. It's good to mention that by deleting the weight factor we can increment the speed of algorithm too. So we can say the efficiency of our algorithm is greater than XCLS++ efficiency too. If **k** and **h** be the depths of two trees, the time complexity in the worst condition will be **O[2^(h+k)]** and in the best condition will be **O(2^h)** if **h** is depth of clustered tree.

6. EVALUATING ALGORITHMS

Our justification showed that the effectiveness as well as the efficiency of new method is better that the two other methods. For proving this sentence in this section the new way, XCLS++and XCLS+ algorithms have been implemented and compared. All of them were implemented with **C** language in **DOS** environment on a machine with 2.4 GHZ Intel Celeron CPU and 512 MB of RAM. The evolution criteria were implemented for evaluating XML files in the same conditions too. As we will see, the results of experiments like above examples, confirm optimality and efficiency of the proposed algorithm is higher than the two other methods.

6.1. Dataset to evaluation

For evaluating, files have been considered from two addresses [6] and [7].

6.2. Evaluation criteria

There are three items for calculating accuracy clustering algorithms: 1-entropy 2-purity 3-fscore

6.2.1 Entropy

Entropy is sum documents which located in the cluster i which are of the class r. The entropy formula is:

$$\boldsymbol{Entropy} = \sum_{i=1}^{k} \frac{\boldsymbol{n_i}}{\mathbf{N}} \boldsymbol{E(C_i)}$$

$$as$$

$$\boldsymbol{E(C_i)} = \frac{\mathbf{1}}{\boldsymbol{logk}} \sum_{i=1}^{k} \frac{\mathbf{n_i^r}}{\boldsymbol{n_i}} \boldsymbol{log} \frac{\mathbf{n_i^r}}{\boldsymbol{n_i}}$$

In above formulas C_i, N, k, n_i and n_i^r are respectively ith cluster, total number of incoming documents, number of clusters, number clustered documents in cluster i and number clustered documents in cluster i of class r. If the entropy value be closer to zero the efficiency is better.

6.2.2 Purity

Purity is sum maximum documents which located in the cluster i which are of the class r. The purity formula is:

$$\boldsymbol{Purity} = \sum_{i=1}^{k} \frac{\boldsymbol{n_i}}{\mathbf{N}} \boldsymbol{P(C_i)}$$

as

$$\boldsymbol{P(C_i)} = \frac{\mathbf{1}}{\boldsymbol{n_i}} \mathbf{max}(\mathbf{n_i^r})$$

If the purity value be closer to one the efficiency will be better.

6.2.3 Fscore

Fscore is another item created by combination of above two items and is:

$$\boldsymbol{FScore} = \frac{\sum_{r=1}^{k} \boldsymbol{n_r F(Z_r, C_i)}}{\boldsymbol{N}}$$

as

$$\boldsymbol{F(Z_r, C_i)} = \frac{\boldsymbol{P(Z_r, C_i)} * \boldsymbol{r(Z_r, C_i)}}{\boldsymbol{P(Z_r, C_i)} + \boldsymbol{r(Z_r, C_i)}} = \frac{\mathbf{2} * \mathbf{n_i^r}}{\boldsymbol{n_i} + \boldsymbol{n_r}}$$

,

$$\boldsymbol{r(Z_r, C_i)} = \frac{\mathbf{n_i^r}}{\boldsymbol{n_r}}$$

,

$$\boldsymbol{P(Z_r, C_i)} = \frac{\mathbf{n_i^r}}{\boldsymbol{n_i}}$$

If the fscore value be closer to one the efficiency will be better.

After implementation, results are calculated and compared for analyzing. In order to testing implemented program, incoming XML files consists of 1000 different classes, such as medical files, colleges, shops, cars, insurance,

etc…. Files were evaluated with the algorithms. The results of algorithm are in Table 1 and include:

Table1. Results of algorithms on XML files

	ENTROPY			PURITY			FSCORE		
ALGORITHM ---------- ------ THRESHOLD	**XCLS++ improved**	**XCLS++**	**XCLS+**	**XCLS++ improved**	**XCLS++**	**XCLS+**	**XCLS++ improved**	**XCLS++**	**XCLS+**
0.7	0.22	0.26	0.30	0.95	0.90	0.75	0.88	0.80	0.80
0.8	0.18	0.25	0.26	0.80	0.80	0.78	0.90	0.80	0.79
0.9	0.20	0.30	0.30	0.96	0.83	0.80	0.95	0.90	0.88

As previous examples, the results obtained in this section shows that the new method has higher effectiveness than both XCLS++ and XCLS+ methods. Not only optimality of new method is high but also deleting weight factor causes decreasing execution time. This topic is discussed in the next section.

7. COMPARING EXECUTION TIME

After comparing effectiveness of algorithms, in this section execution time is compared too. As previously cited weighting factor has lost its efficiency due to the fundamental change on the basic formula. With deleting weight factor amount of calculation is deleted and running time is predictably decreases. Algorithms have been simulated in **MATLAB 6.5.1** for reaching expected results. This simulation has been done on two same trees with XCLS++ and new algorithms. Simulated results are:

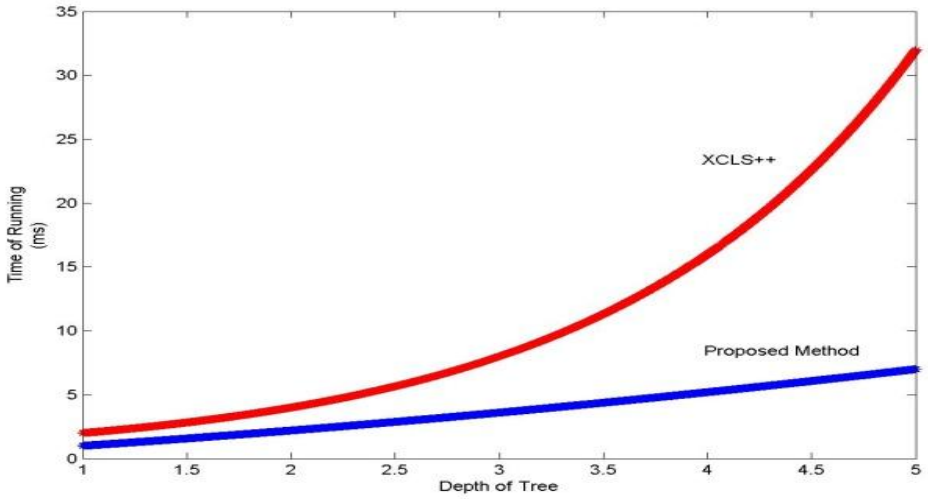

Figure 4. Same nodes of two trees are in up levels

As Figure 4 shows the increasing of the depth of trees causes the augmentation of calculations and running time for new method is lower than XCLS++ method. We must mention that the running time for the XCLS+ is same as XCLS++ because weighting factor is in both of them.

8. CONCLUSION

The purpose of this paper is clustering of XML documents. Criteria for clustering are structural, content-based, and hybrid (the structure with the content). XCLS++ method is a clustering method in which criteria for clustering is done based on the structure. Despite good performance for it in comparison to the previous methods, weighting nodes in some documents causes it to be inefficient. Therefore, new algorithm with deleting it has been proposed. The results of entropy, purity and Fscore calculation show that the proposed method works better than the previous method. In future new weight of levels will be replaced for obtaining a similarity actually and better than proposed method too. Also in future the new method will be evaluated on much more documents and by comparing those we will be able to obtain better results.

9. REFERENCES

[1] Ilwan Choi, Bongki Moon, Hyoung-Joo Kin, A clustering method based on path similarities of XML data, Data & Knowledge Engineering, 2006.

[2] Andrewdn, Jag, Information systems engineering, Evaluating Structural Similarity in XML Document, *WISE'07 Proceedings of the 8thinternational conference on Web Information*, 2007.

[3] Tien Tran, Richi, Peter, Data Mining, Combining Structure and Content Similarities for XML Document Clustering, *Conference 27-28November, Glenelg, South Australia,* 2008.

[4] Woosaeng Kim, Computer Engineering and Applications, XML document similarity measure in terms of the structure and contents, *CEA'08 Proceedings of the 2nd WSEAS International Conference*, 2008.

[5] G. R. Nayak, Fast and effective clustering of XML data using structural information knowledge. Information System, 2008.

[6] The Wisconisn's XML data bank. Accessed from:http://www.cs.wisc.edu/hiagara/data.html Cited2012.

[7] The XML data repository. Accessed from: http://www.cs.washington.edu/research/xmldatasets/. Cited 2012.

[8] Waraporn Viyanon, Sanjay K. Madria, Sourav S. Bhowmick, Management of Data, XML Data Integration Based on Content and Structure Similarity Using Keys, 2008.

[9] Aptarshi Ghosh and Pabitra Mitra, Pattern recognition, ICPR Combining Content and Structure Similarity for XML Document Classification using Composite SVM Kernels, *19th International Conference*, 2008.

[10] Jing Peng Dong Qing Yang Shi Wei Tang et al, similarity in chinese text processing, A New Similarity competing method based on concept, series F: Information science, 51(9): p1212-1230, 2008.

[11] Mohamad Alishahi, Mohmoud Naghibzadeh and Baharak Shakeri Aski, Tag Name Structure-based Clustering of XML Documents, *International Journal of Computer and Electrical Engineering* Vol. 2, No. 1, February, 2010.

[12] Ahmad Khodayar and Hassan Naderi, XCLS++: A new algorithm to improve XCLS+ for clustering XML documents, *International Journal of Information Technology, Control and Automation (IJITCA) Vol*.2, No.4, 2012.