A Hybrid Method for XML Clustering by Structure and Content

Yong Piao and Xiu-kun Wang
School of Electronic and Information Engineering, Dalian University of Technology, Dalian, China
Email: {piaoy, jsjwxk}@dlut.edu.cn

***Abstract*—An effective XML cluster method called neighbor center clustering algorithm (NCC) is presented in this paper, whose similarity is obtained through both structural and content information contained in XML files. Structural similarity is firstly measured by frequency-path model and its similarity calculation algorithm with position and frequency weight by longest common subsequence is introduced. In order to improve the performance and precision, the frequency-path model is further extended by considering the structure and content information simultaneously. Experiments show that the NCC embed with hybrid similarity calculation method can obtain high purity and F-measure value and is effective and applicable for clustering XML with both homogenous and heterogeneous structures.**

***Index Terms*—neighbor center clustering, position and frequency weight, longest common subsequence, hybrid similarity calculation**

I. INTRODUCTION

XML (eXtensible Markup Language), as a common data representation and exchange format on the Internet, contains a rich entailment of information. Also, data mining on XML has become an important part in text mining research, in which large-scale text clustering is one of the effective solutions for massive texts. An efficient and fast XML clustering mechanism, which can provide better data for decision support, will greatly shorten the information retrieval time, improve the efficiency of data query and help find out potential value of information. The most important feature of XML document, which is different from other textural ones, is its structural character. For this reason, we believe that the structure of XML should also play important role in XML clustering.

Considering the structural character of XML documents, many traditional text clustering methods are not suitable for XML. Currently, partitioning and hierarchical methods are most widely used in XML clustering [1-3], but the effect of these two traditional methods in dealing with irregular non-spherical document clustering is not so satisfactory, and besides, they are not well in distinguishing noise or isolated points effectively. In terms of computational complexity, the searching time of traditional methods for cluster centers increase rapidly, this is an obstacle to get better performance in XML clustering. In addition, traditional partitioning methods represented by K-Means and K-Medoids have to be specified the clusters number K in advance. Due to these reasons, a neighbor center clustering algorithm with similarity (NCC) is proposed in this situation. It is not only simple, but can find non-spherical structure documents and distinguish noise or isolated point effectively as well.

Similarity among documents is the key issue in the field of document clustering. So far, the methods proposed for this purpose can be roughly classified into three types, namely by the graph matching, by the edit distance or by the tree path model. Reference [4] describes an XML document with a directed graph and calculates similarity between XML documents by graph matching in order to cluster XML with similar structure. But the result is not satisfactory due to its low accuracy. Reference [5] improves [4], in which its method leads to some limitations to clustering results without considering the order relationships between edges in the discussion of equal direct edges. Reference [6] and [7] introduce a concept of edit distance. Reference [6] calculates the similarity with graph matching algorithm by describing XML as a directed graph, while [7] uses tree editing distance to calculate the similarity, so do [1,8,9]. However, it is not suitable for document processing due to its high computational complexity. As the graph cannot express XML structure well, Reference [10] proposes a tree path model representation, which is simpler than the tree editing distance with a lower time complexity, but it uses a complete path matching method widely while handling the matching procedures, so do references [11,12] in frequent path mining and matching, including its improvement [13]. The complete path matching method is useful in XML clustering in tree path models because XML structure information ignored by the complete path matching has little impact on clustering XML which have the same DTD, but it is not true if their DTDs are different, e.g. they have various structures.

In this paper our similarity measurement among XML documents with different structures is firstly presented,

Corresponding author: Yong PIAO

doi:10.4304/jsw.6.12.2361-2368

which is a similarity calculation algorithm with position and frequency weight by longest common subsequence (PFWLCS). On the basis of expressing XML document structures by correspondent paths using DOM tree, we extend the original tree path model to the frequency-path model by which we not only preserve label information of correspondent nodes, which decrease the original tree path model scale consumedly on the condition of not losing meaningful information and reduce the burden for later calculation, but also save the frequency structure of the original XML to improve the accuracy of the similarity. It makes the calculated similarity closer to the actual value by using the longest common subsequence method with position and frequency information. Furthermore, we continue to improve the frequency-path model by also considering node textural content, which makes the result more accurate and applicable, e.g. the hybrid method.

In Section II, Frequency-Path model and basic idea of hybrid similarity calculation methods are briefly introduced, followed by main steps of cluster algorithm NCC in detailed description in Section III. In Section IV experiments and its results are given showing better performance of our methods. Finally we summarize the whole work and provide future applications and research directions.

II. Similarity Calculation

A. Structural Similarity

1) Frequency-Path Model

Definition 1: $FPath=(f, v_1, v_2, \ldots, v_m, c_1, c_2, \ldots, c_m)$, where f denotes the number of occurrences of path in the current document; $(v_1,v_2,\ldots,v_m)$ is a node sequence from the root of XML DOM tree to one of its leaves; the c_i in $(c_1,c_2,\ldots,c_m)$ denotes the number of occurrences of v_i, whose ancestor nodes $v_1,v_2,\ldots,v_{i-1}$ are the same; m denotes the length of FPath (the node sequence).

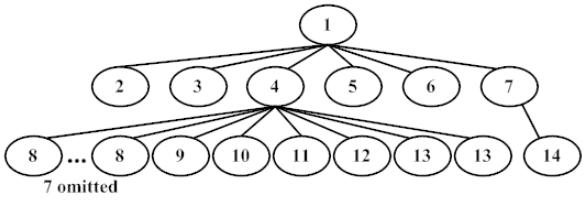
Definition 2: $XMLDoc=(FPath_1, FPath_2, \ldots, FPath_n)$, $FPath_i$ and $FPath_j$ are not the same FPath. We call two FPaths same only if the nodes at corresponding location of the two FPaths are completely identical. n denotes the number of various FPath.

Before calculating the similarity of XML, we extract the structure information of XML into FPaths where only the label of the node (structure) is considered. Other information such as data type and constraints are ignored. In Fig. 1 is an XML tree model and in Fig. 2 is a path model with its statistical information.

2) Frequency-Path Model Generation

Fig. 3 is the pseudo code of the algorithm to create the FPath model from XML document. The input of this algorithm is an XML document and output is an FPath model of it.

We also consider the semantic meanings that a node name can have during the structure matching. It is necessary since we are aiming at XML documents from different DTDs, which may not use the same word to express the similar meaning. For expressing the similar meaning only one word was taken from the synonymous word sets provided by WordNet. Besides, we assume that the node in higher hierarchy contributes more to the similarity than in lower hierarchy during the FPath matching. We will cover that in detail later.



1: automobile
2: manufacturer
3: model
4: feature
5: engine
6: year
7: engine
8: pro
9: mpg
10: safety
11: drive
12: seat
13: con
14: type

Figure 1. XML DOM Model

For reducing the complexity of getting semantic FPath, we use a parameter ζ to denote the depth of the node hierarchy being considered. For example, if $\zeta=1$, we just consider the meaning of the first node (root) of FPath.

```
1 automobile manufacturer 20 1
1 automobile model 20 1
1 automobile year 20 1
1 automobile engine type 20 1 1
1 automobile transmission 20 1
1 automobile feature safety 20 15 1
1 automobile feature drive 20 15 1
1 automobile feature seat 20 15 1
1 automobile feature mpg 20 15 1
9 automobile feature pro 20 15 9
2 automobile feature con 20 15 2
```

Figure 2. A frequency-path model

3) Algorithm PFWLCS

In this part the algorithm PFWLCS (Position and Frequency Weight by Longest Common Subsequence) used to calculate structural similarity of XML documents is described.

Definition 3: Subsequence: we call $<a_{i1}, a_{i2}, \ldots, a_{ik}>$ a subsequence of $<a_1, a_2, \ldots, a_n>$, only if $1<=i1<i2<\ldots<ik<=n$.

Definition 4: Common subsequence: we call $<c_1, c_2, \ldots, c_k>$ one common subsequence of $<a_1, a_2, \ldots, a_n>$ and $<b_1, b_2, \ldots, b_m>$, only when $<c_1, c_2, \ldots, c_k>$ is a subsequence of $<a_1, a_2, \ldots, a_n>$ and also a subsequence of $<b_1, b_2, \ldots, b_m>$, k denotes the length of the common subsequence $<c_1, c_2, \ldots, c_k>$.

In our method, we use Longest Common Subsequence (LCS) in matching two given paths. In Table I different situations are illustrated by xPath [10-12], PCXSS [13] and LCS[14-15] respectively, showing more information can be kept by using LCS than other methods.

```
Name: getMypathModel
Input:  an XML document to: xmlDocumentName
        node hierarchy: ζ
Output: a text document stored F_Path model: mypathModel
Pseudo code: void getMypathModel
                    (xmlDocumentName, ζ, mypathModel.txt)
{doc=getDocument(xmlDocumentName);
 //Parse XML document
 root=getRoot(doc);//get doc root
 getMypathModel(root, mypathModel);
 //get the paths and store in mypathModel.txt

 //the semantic process
 Initialize wordList //for the semantic process
 while(ζ>0){
      for (each Path in mypathModel.txt){
           node=getNodeFromPath(ζ);//get the ζ node
           if(node exists in Line j of wordList)
                Use the first word of Line j in wordlist to
                 replace node;
      else
           Insert node and its synonymous words into
           a new line in wordlist;
      }
      ζ--;
 }}
```

Figure 3. FPath model generation

We introduce a position-frequency weight vector *[W(1), W(2), ..., W(n)]*, where *(1, 2, ..., n)* is the index of nodes in a FPath, representing the hierarchy level. We assume that the node in higher hierarchy contributes more to the similarity than in lower hierarchy during FPath matching. Then the weight function *W(i)* must be digressive with *i* increases. In addition, *W(i)* also has the character below.

$$W(i)>0, \sum W(i)=1 \ (i=1, \ldots, n)$$

TABLE I.
PATH MATCHING USING DIFFERENT METHODS

No.	$Path_1$	$Path_2$	LCS	PCXSS	xPath
1	(a,b)	(a,b,y)	(a,b)	(a,b)	NULL
2	(a,b)	(y,a,b)	(a,b)	(a,b)	NULL
3	(x,a,b)	(y,a,b)	(a,b)	(a,b)	NULL
4	(a,b,x)	(a,b,y)	(a,b)	NULL	NULL
5	(a,b,x)	(a,k,b,y)	(a,b)	NULL	NULL
6	(a,h,b,x)	(a,k,b,y)	(a,b)	NULL	NULL
7	(a,b,x,h)	(a,b,y,k)	(a,b)	NULL	NULL

To further explain the necessity for position-frequency weight vector, the followings are discussed.

Suppose there are several paths when calculating the XML document similarity: P_1*(1, a, b, c, 1, 1, 1),* P_2*(1, a, b, x, 1, 1, 1),* P_3*(1, a, x, b, 1, 1, 1),* P_4*(2, a, b, x, 2, 2, 1)*.

For the similarity comparison of actual data, the nodes in higher hierarchy have greater effect in the XML document tree, namely the more front position the node is at, the more contributions to the similarity during FPath matching. Therefore, the similarity between P_1 and P_2 is significantly higher than it between P_1 and P_3. It shows the weight function *W(i)* is closely related to the position factor, namely position weight *T(i)*.

We also notice that the similarity degree between P_1 and P_4 is significantly greater than it between P_1 and P_2. That is because in the case of FPaths with the same node position, higher frequency of the same node indicates its role is more dominant than other nodes. Hence, the weight function *W(i)* is also closely related to node's frequency factor, namely frequency weight *F(i)*.

From the above, the position-frequency weight function *W(i)* is composed of *T(i)* and *F(i)*, where the position weight function $T(i) = 1/2^i$. As for the frequency weight function *F(i)*, we first define frequency equation of node V_i as $c_i log(c_i/f_s+1)$ (c_i is from the definition 1; f_s is $\sum f$ for all FPaths in the document). Then the normalized function of frequency weight can be expressed as (1).

$$F(i) = c_i \log(c_i / f_s + 1) \Big/ \sum_{i=1}^{n} c_i \log(c_i / f_s + 1) \quad (1)$$

and the position-frequency weight function *W(i)* is,

$$W(i) = (T(i) + F(i))/2 \quad (2)$$

We prove that (2) satisfies the features of *W(i)*, e.g. $W(i)>0$ and $\sum W(i)=1$.

$$\sum_{i=1}^{n} W(i) = \frac{1}{2} \sum_{i=1}^{n} (T(i)+F(i))$$

$$= \frac{1}{2} \left(\sum_{i=1}^{n} T(i) + \sum_{i=1}^{n} F(i) \right)$$

$$= \frac{1}{2} \left(\sum_{i=1}^{n} \frac{1}{2^i} + \sum_{i=1}^{n} \frac{Ci \log(Ci/fs+1)}{\sum_{i=1}^{n} Ci \log(Ci/fs+1)} \right)$$

$$= \frac{1}{2} \left(\frac{\frac{1}{2}(1-\frac{1}{2^n})}{1-\frac{1}{2}} + 1 \right) = 1 - \frac{1}{2^{n+1}}$$

$$\therefore n \to \infty, \sum_{i=1}^{n} W(i) = \lim_{n \to \infty} (1 - \frac{1}{2^{n+1}}) = 1$$

Definition 5: FPath similarity: Suppose two FPaths, $FPath_1 = (f_{p1}, x_1, x_2, \ldots, x_n, c_{x1}, c_{x2}, \ldots, c_{xn})$, $FPath_2 = (f_{p2}, y_1, y_2, \ldots, y_m, c_{y1}, c_{y2}, \ldots, c_{ym})$, the longest common subsequence (LCS) is $LCSPath=(z_1, z_2, \ldots, z_k)$, the hierarchy of the nodes in LCSPath in $FPath_1$ is $Hierarchy_1=(l_1, l_2, \ldots, l_k)$ orderly, the hierarchy of the nodes in LCSPath in $FPath_2$ is $Hierarchy_2= (h_1, h_2, \ldots, h_k)$ orderly. Then the similarity of $FPath_1$ and $FPath_2$ is described as (3) below.

$$\text{similarity} = \left(\sum_{i=1}^{k} W(l_i) \Big/ \sum_{j=1}^{n} W(j) + \sum_{i=1}^{k} W(h_i) \Big/ \sum_{j=1}^{m} W(j) \right) \Big/ 2 \quad (3)$$

In some practical situations, the occurrence number of the same path is also an important component of XML structure information, but (3) does not contain the path frequency information and just uses label information, thus leading to the situation that the calculated result does not represent the true sense of XML similarity. So we integrate the path frequency into (3) in order to make the result closer to actual value.

From definition 5, f_{p1} and f_{p2} are the occurrence numbers of $FPath_1$ and $FPath_2$, while f_{p1}/f_{s1} and f_{p2}/f_{s2}

are the path frequencies of $FPath_1$ and $FPath_2$ (f_{s1} is the sum of all path frequency in $FPath_1$, and f_{s2} is the sum of all path frequency in $FPath_2$). On the basis of the TF-IDF statistics theory widely used in text mining, we assume that, the larger the recurrent number of a path in XML is, the higher path frequency it is. That is to say, the more important the path is, the more structure information the path contains. Therefore, the similarity of path should be appropriate to be improved, in which the path frequency is large and the degree of increasing similarity should be within the scope of $[1-\sum_i W(l_i)]/\sum_j W(j),\ i=1..k,\ j=1..n$, so (3) is modified as follows.

$$\text{similarity}=\left(\frac{\sum_{i=1}^{k}W(l_i)}{\sum_{j=1}^{n}W(j)}\left(1+\left(1-\frac{\sum_{i=1}^{k}W(l_i)}{\sum_{j=1}^{n}W(j)}\right)\frac{f_{p1}}{f_{s1}}\right)+\frac{\sum_{i=1}^{k}W(h_i)}{\sum_{j=1}^{m}W(j)}\left(1+\left(1-\frac{\sum_{i=1}^{k}W(h_i)}{\sum_{j=1}^{m}W(j)}\right)\frac{f_{p2}}{f_{s2}}\right)\right)\Big/2 \quad (4)$$

Definition 6: XML document similarity: Suppose two XMLDocs, $XMLDoc_1= (FPath_1, FPath_2, ..., FPath_n)$, $XMLDoc_2= (FPath_1', FPath_2', ..., FPath_m')$, $m>=n$, Each FPath in $XMLDoc_1$ finds LCS with every FPath in $XMLDoc_2$, and calculates the similarity according to (4). We denote the biggest similarity as s_i, the similarity between $XMLDoc_1$ and $XMLDoc_2$ is formed with (5) below.

$$\text{Similarity}=\left(\sum_{i=1}^{n}S_i\Big/n+\sum_{i=1}^{n}S_i\Big/m\right)\Big/2 \quad (5)$$

```
Name: PFWLCS
Input: two F_Path model: mypathModel_1, mypathModel_2
Output: similarity of two Documents
Pseudo code: double getPFWLCSSimilarity
                        (mypathModel_1, mypathModel_2)
{ //n1, n2: number of paths in mypathModel_1, mypathModel_2
 Initialize n1, n2;
 //fs1, fs2: sum of path frequencies in the two models
 Initialize fs1, fs2;
 if(n1>n2)
      return getSimilarity(mypathModel_2, mypathModel_1);
      else{
           for(each path p1_i in mypathModel_1){
                for(each path p2_j in mypathModel_2){
                //a1, a2: position-frequency weights of p1_i,p2_j
                Initialize a1, a2;
                //get the longest common path of p1_i,p2_j
                lcs=getLCS(p1_i,p2_j);
                //w1,w2: position-frequency weights of the lcs
                Initialize w1,w2;
                im1=w1/a1;
                sim2=w2/a2;
                //f1,f2: path frequency of p1_i,p2_j
                Initialize f1,f2;
                sim=(sim*(1+(1-sim1)f1/fs1)
                          +sim2*(1+(1-sim2)f2/fs2))/2;
                }
                similarity += sim;
           }
 }
 //get the similarity of documents
 similarity=(similarity/n1+similarity/n2)/2;
 return similarity;
}
```

Figure 4. PFWLCS algorithm

Fig. 4 is the pseudo code of the algorithm to calculate the similarity of XML document based on the FPath model. The input of this algorithm is two FPath models, and the output is the similarity of the two XML documents, from which the two inputted FPath models come.

B. Hybrid XML similarity calculation

The PFWLCS algorithm based on frequency and path is introduced previously. Although it can obtain better result when processing XMLs from different DTDs, structured information is mainly concerned, e.g. structural similarity without considering node content. We know that the content feature of XML structure has also effects on XML document classification and clustering and has contributions to the result of data mining, which should not be ignored.

The content information is the textual part between tags. A method called SCSC (Similarity Calculation with Structure and Content) is presented in this subsection. The Frequency-path model is firstly improved in SCSC by adding the element content vector under the same path, making the presentation of XML documents richer. Moreover, a level ratio is introduced in XML similarity calculation, considering both the structural and element content similarity and making the result more sensible.

1) Improved frequency-path model

Definition 7: Improved tree path model: $IFPath=(f, v_1,v_2,...,v_m, /E/,c_1,c_2,...,c_m)$, where f denotes the number of occurrences of path in the current document; $(v_1,v_2,...,v_m)$ is a node sequence from the root of XML DOM tree to one of its leaves; E is a content vector $(e_1,e_2,...,e_t)$ under structural path $(v_1,v_2,...,v_m)$; the c_i in $(c_1,c_2,...,c_m)$ denotes the number of occurrences of v_i, whose ancestor nodes $v_1,v_2,...,v_{i-1}$ are the same; m denotes the length of IFPath (the node sequence).

```
1 automobile manufacturer / Cadillac / 20 1
1 automobile model / Cadillac CTS / 20 1
1 automobile year /2004 / 20 1
1 automobile engine type / 255 hp / 20 1 1
1 automobile transmission / Speed Manua / 20 1
1 automobile feature safety
                / Driver-Passenger airbags / 20 15 1
1 automobile feature drive / RWD / 20 15 1
1 automobile feature seat / 5 / 20 15 1
1 automobile feature mpg / City / 20 15 1
9 automobile feature pro
                / ABS Air Base CD Player Leather
                  Seats Side Theft Tracking
                  Traction Control Highway / 20 15 9
2 automobile feature con
                / Transmission Changer / 20 15 2
```

Figure 5. An improved FPath model

Example in Fig. 5 is improved based on the model introduced before. A content vector $(e_1,e_2,...,e_t)$ is supplemented, by which the XML content information is

further kept, making hybrid computation of XML similarity of both structure and content possible.

2) Content similarity calculation

Suppose we have two paths P_1 and P_2 belonging to documents D_1 and D_2 respectively, E_1 is content vector of P_1 and E_2 is content vector of P_2.

Before processing, we use TF-IDF theory to complete feature word selection and extraction against E_1 and E_2, resulting in feature word vector E_1' and E_2'. Then we use cosine similarity equation to get content similarity.

$$\text{Similarity(content)} = \frac{\omega}{\sqrt{n1+n2}} \qquad (6)$$

Where n_1 is the dimension of E_1' and n_2 is the dimension of E_2', ω is the number of common feature words E_1' and E_2' have.

3) Overall Similarity Calculation

Due to the reason that a new part is added to the frequency-path model, the original similarity calculation method is about to be modified accordingly. Similarity of a complete document is defined:

$$Sim = \alpha * Sim_{content} + (1-\alpha) * Sim_{structure} \qquad (7)$$

The final similarity is determined by both structure and content similarity, where α is called the level ratio parameter, presenting the structure layer of the element content. Structural similarity is still calculated using PFWLCS algorithm.

The higher level an element is at, the more closely it is to root node and has more contribution. That is, the level ratio factor α is greater and so has close relation with the position factor $T(i)$:

$$\alpha = (T(i)+T(j))/2.$$

$T(i)$ and $T(j)$ are two position weights of the last nodes in the compared paths separately.

III. XML Documents Clustering

A. Steps of Algorithm NCC

1) A point is chosen as the initial cluster center O_1 from a data set.

2) Set the center threshold parameter $(\xi_1, \xi_1 \geq 0)$. Then the similarity of O_1 between each remaining points from the data set is calculated and compared. If the similarity is greater than ξ_1, the point will be put into the cluster C_1, where O_1 is in.

3) Set the neighbor threshold parameter $(\xi_2, \xi_2 \geq \xi_1)$. At this time, the similarity of the points except O_1 in C_1 with the remaining points from the data set is calculated and compared again. If the similarity is greater than ξ_2, the point will be put into the cluster C_1.

4) Set the isolated threshold parameter $(\varphi, \varphi=1, 2 \ldots n)$. If the number of points in C_1 is less than φ, the cluster C_1 will be discarded.

5) A point is chosen as another cluster center O_2 from the remaining points in the data set. Repeat *2) 3) 4)* steps until there are no points in the data set left.

```
Name: Similarity calculation using SCSC
Input: Two improved frequency-path models:
                    mypathModel_1, mypathModel_2
Output: Similarity score between the two documents
Process: double getSimilarity
                    (mypathModel_1, mypathModel_2)
{ Initialize n1, n2;
 //n1, n2: number of different paths in the two models
 if(n1>n2)
      return getSimilarity(mypathModel_2, mypathModel_1);
 else{
      //Extract element content feature words using TF-IDF
      mypathModel_1'=do_TFIDF(mypathModel_1);
      mypathModel_2'=do_TFIDF(mypathModel_2);
      for( each path p1_i in mypathModel_1'){
           for(each path p2_j in mypathModel_2'){
                //Get structural similarity using PFWLCS
                SimS=getPFWLCSSimilarity(p1_i, p2_j);
                //Get number of common feature words
                w=getCommonElet(p1_i, p2_j);
                //t1, t2: number of feature words
                Initialize t1, t2;
                //Calculate content similarity
                SimC=w/sqrt(t1+t2);
                //Ti, Tj: position weights of last nodes
                //in p1_i, p2_j
                Initialize Ti, Tj;
                //calculate level ratio factor
                a=getPostion();
                //calculate similarity between two paths
                sim=a*SimC (1-a)*SimS
           }
           similarity +=sim;
      }}
 similarity=(similarity/n1+similarity/n2)/2;
 return similarity;}
```

Figure 6. SCSC algorithm

B. Analyses of NCC

The basic idea of NCC algorithm is trying to identify the actual cluster from the data set in each iterative step greedily. Below explains the main idea of NCC algorithm.

Considering the XML document set $D=\{d_1, d_2, d_3, d_4, d_5, d_6\}$ (d_i denotes the i^{th} XML document), where d_1, d_2, d_3, d_4, d_5 are actually in the same cluster, while d_6 is in another one. Set the parameters $(\xi_1=0, \xi_2=0.1, \varphi=1)$.

NCC arbitrarily selects d_1 as the initial cluster center and then finds the similarity between d_1 and d_2 greater than ξ_1, so d_2 is put into the cluster C_1, where d_1 is. So does d_3. At the moment, in order to avoid the situation that the chosen cluster center may be irrelevant resulting in imperfect clusters, it calculates and compares the similarity of the points except d_1 in C_1, in this case d_2 and d_3, with the remaining points. The purpose here is to spread the function of cluster center out over the neighborhood points in one cluster, which tries the number of matched points into one cluster, e.g. C_1, as many as possible. In Fig. 7, d_4 and d_5 are also put into the cluster because of d_2 and d_3.

Compared with some traditional algorithms, the NCC algorithm reduces the repeated calculation complexity on choosing cluster center in each iterative step, and improves overall efficiency. The clustering result from Fig.

7 is non-spherical, and besides, isolated points or isolated clusters are also identified, e.g. the point d_6 in the Figure.

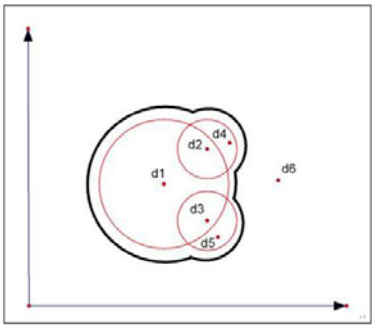


Figure 7. An Example of NCC clusters

NCC algorithm is suitable for XML clustering considering the XML structural character, while some vectorization methods ignore this kind of information during their calculation. It not only retains XML document's structural information, but also evaluates XML's textual meaning when with the method SCSC embedded. Therefore, the NCC algorithm based on SCSC has advantages in XML document clustering.

C. Evaluation of NCC

At present the two parameter indexes used widely to evaluate the overall performance of a clustering algorithm are the Purity and F-measure. The experimental data are the known document set or have usually been categorized before evaluation [16]. The purity of the cluster r is defined as follows:

$$P(Sr)=\frac{1}{n_r}*Max(n_r^i)$$

The purity of the overall clustering is defined as:

$$Purity=\sum_{r=1}^{k}\frac{n_r}{n}P(Sr)$$

Where n_r^i is the number of document belonging to type i, which is assigned to cluster r; n_r is the number of documents in cluster r; n is the number of the whole document set.

High purity can be easily achieved when the number of clusters becomes larger. In particular, the purity will be 1 if each document gets its own cluster. Thus, F-measure is introduced as a harmonic mean to combine both precision and recall factors.

Recall: $Recall(i,r) = n(i,r) / n_i$

Precision: $Precision(i,r) = n(i,r) / n_r$

where $n(i,r)$ is the number of document belonging to type i in the cluster r; n_r is the number of document in cluster r; n_i is the number of documents of type i. So the F-measure between cluster r and type i is defined as follows.

$$f(i,r)=\frac{2*recall(i,r)*precision(i,r)}{recall(i,r)+precision(i,r)}$$

The F-measure of the overall clustering is defined as:

$$F=\sum_{i}\frac{n_r}{n}Max\{f(i,r)\}$$

IV. Experiment and Discussion

The goal of our experiments is to examine the effectiveness of the SCSC by calculating the similarity of XML documents using both structure and content information. Further, SCSC is embedded into NCC in clustering XML and we obtained satisfactory results.

Two data sets are used in the following experiments. The first data set used in our experiment is a real life data set introduced in [17], which is generated in the XML/XSLT version of web pages from 20 different sites belonging to 4 categories, labeled as "automobile" , "movie" , "reference" and "software". There are a total of 120 documents: 24 in "automobile", 24 in "movie", 48 in "reference" and 24 in "software". There is no cross-labeling and the depth of the DOM tree of these XML documents is 5. The second data set is from Sigmod XML collection, 84 files and 4 DTDs are included.

In the experiments, NCC method is used to cluster the two data sets and evaluate results according the recall ratio, accuracy and F-measure introduced before.

First we use improved frequency-path model to extract information from all documents, then complete feature selection and extraction. We randomly choose one file from two data sets as cluster center and calculate similarities of other files against the cluster center respectively, this process is repeated 10 times and the result is as following:

Table II.
Experiment result

		DataSet1 (SCSC)	DataSet1 (PFWLCS)	DataSet2 (SCSC)
Average Values	Recall Ratio	81.13%	83.02%	88.56%
	Accuracy	94.08%	94.29%	95.81%
	F-measure	87.13%	88.30%	92.17%
	Purity	94.35%	93.89%	85.96%
	Clusters	3.72	3.80	3.87

From the Table II, results of dataset1 and dataset 2 are all satisfactory. Specifically, results of the first two columns, e.g. dataset1 (SCSC) considers both the structure and content information, while dataset1 (PFWLCS) is calculated by considering only structure information. Each item of dataset1 (SCSC) is a little lower than dataset1 (PFWLCS), that is because in dataset1 XML files are from many different DTDs, even in the same cluster. Therefore structural information plays more important role than content information in this situation. However, considering both the structure and content information is more nature and reasonable in practice and can reflect more real feature of data.

A good similarity calculation will make the similarity closer within one cluster and larger outside the cluster. In the following experiments, we pick up one file arbitrary from 4 clusters of the two datasets separately. We compare the similarity of this file to other files in its cluster and repeat this process 10 times. The following results are obtained.

TABLE III.
SIMILARITY COMPARISON IN DATASET1

$SimAvg(S_i,S_j)$	S1	S2	S3	S4
S1	0.4853	0.0054	0.0026	0.0018
S2	0.0054	0.4526	0.0021	0.0034
S3	0.0026	0.0021	0.4722	0.0012
S4	0.0018	0.0034	0.0012	0.4336

S_i is the i^{th} cluster in Dataset1, $SimAvg(S_i, S_j)$ is the average similarity between documents in cluster S_i and S_j.

TABLE IV.
SIMILARITY COMPARISON IN DATASET2

$SimAvg(Ci,Cj)$	C1	C2	C3	C4
C1	0.9646	0.1326	0.1464	0.1318
C2	0.1326	0.9524	0.1298	0.1431
C3	0.1464	0.1298	0.9346	0.1373
C4	0.1318	0.1431	0.1373	0.9546

C_i is i^{th} cluster in Dataset2, $SimAvg(C_i,C_j)$ is the average similarity of documents in clusters C_i and C_j.

From the two tables above, Similarities in the same cluster are all far greater than in other clusters using SCSC. We also notice that scores of dataset2 is a litter greater than in dataset1, this is because most XML files in dataset1 are from different DTDs and structure information become more significant, while in dataset2 XML files in one cluster are most from same DTD and content information become important. Anyway, SCSC can handle both of these conditions and results showing that the hybrid method is more applicable and effective.

V. CONCLUSION

We have demonstrated an XML document clustering method NCC, when embed with SCSC higher effectiveness will be achieved by considering both structure and content similarity. On one hand, the structural similarity is calculated using method PFWLCS with position frequency weight, based on frequency path model, in which more valuable information in XML clustering is kept using the longest common subsequence method and the position frequency weight vector. On the other hand, XML's content similarity is obtained through TF-IDF method. Experiments showed SCSC method could greatly help to improve clustering precision and performance, decrease complexity when embedded in NCC, and is suitable to both homogenous and heterogeneous XML documents.

REFERENCES

[1] T. Dalamagas, T. Cheng, K. J. Winkel, and T. Sellis, "A Methodology for Clustering XML Documents by Structure," *Information Systems*, vol. 31, pp. 187-228, 2006.

[2] G. Costa, G. Manco, R. Ortale, and A. Tagarelli, "A Tree-Based Approach to Clustering XML Documents by Structure," *Knowledge Discovery in Databases: PKDD 2004*, pp. 137-148, 2004.

[3] Pan Youneng, "Research on XML Document Cluster," *Journal of the China Society for Scientific and Technical Information*, 2006, 25(2).

[4] Wang Lian, David Wai-Lok Cheung, Nikos Mamoulis, and Siu-Ming Yiu, "An Efficient and Scalable Algorithm for Clustering XML Documents by Structure," *IEEE Transactions on Knowledge and Data Engineering*, 2004, 16(1), pp. 82-96.

[5] Liu Jiang and Wang Jun, "Reseach on Web XML Document Clustering," *Public Science*, 1002-6908(2007)0620038-03.

[6] Sudarshan S. Chawathe, Anand Rajaraman, Hector Garcia-Molina, and Jennifer Widom, "Change Detection in Hierarchically Structured Information," *In Proceedings of the 1996 ACM SIGMOD international conference on Management of data*, 1996, pp. 493-504.

[7] Zhang K and Shasha D, "On the Editing Distance between Unordered Labeled Trees," *Information Processing Letters*, 1992, 42(3), pp. 133-139.

[8] A. Wojnar, I. Mlynkova and J. Dokulil, "Structural and Semantic Aspects of Similarity of Document Type Definitions and XML Schemas," *Information Sciences*, vol. 180, pp. 1817-1836, 2010.

[9] M. Torjmen, K. Pinel-Sauvagnat, and M. Boughanem, "Towards a Structure-based Multimedia Retrieval Model," *in 1st International ACM Conference on Multimedia Information Retrieval, MIR2008*, August 30, 2008 - August 31, 2008, Vancouver, BC, Canada, 2008, pp. 350-357.

[10] Sachindra Joshi, Neeraj Agrawal, Raghu Krishnapuram, and Sumit Negi, "A Bag of Paths Model for Measuring Structural Similarity in Web Documents," *SIGKDD'03*, 2003, pp. 24-27.

[11] YANG Hou-Qun, HE Zhong-Sh, and LEI Jing-Sheng, "Research of Clustering XML Documents Based on Partition," *Computer Science*, 2008, 35(3).

[12] Ho-pong Leung, Fu-lai Chung, Stephen C.F. Chan, and Robert Luk, "XML Document Clustering Using Common XPath," *In Proceedings of the 2005 International Workshop on Challenges in Web Information Retrieval and Integration, WIRI'05*, 2005, pp. 91-96.

[13] T. Tran and R. Nayak, "Evaluating the Performance of XML Document Clustering by Structure Only," *in 5th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2006*, December 17, 2006 - December 20, 2006, Dagstuhl Castle, Germany, 2007, pp. 473-484.

[14] PIAO Yong, TIAN Wei, and WANG XiuKun, "An Effective Path-based Algorithm to Calculate XML Similarity," *Control and Decision*, 2010, 25(4), pp. 497-501.

[15] Y. Piao and X. K. Wang, "A Hybrid Method for XML Clustering," *in 3rd International Symposium on Parallel Architectures, Algorithms and Programming, PAAP 2010*, December 18-20, 2010, Dalian, China, 2010, pp.286-290.

[16] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze, "An Introduction to Information Retrieval," Cambridge University Press, Cambridge, England, 2009, pp. 356-360.

[17] A. Kurt and T. Engin, "Classification of XSLT-generated Web Documents with Support Vector Machines," *In Knowledge Discovery from XML Documents*, 2006, pp.33-42.

[18] E. Bertino, G. Guerrini and M. Mesiti, "Measuring the Structural Similarity among XML Documents and DTDs,"

Journal of Intelligent Information Systems, vol. 30, pp. 55-92, 2008.
[19] E. Bertino, G. Guerrini and M. Mesiti, "A Matching Algorithm for Measuring the Structural Similarity between an XML Document and a DTD and its Applications," *Information Systems*, vol. 29, pp. 23-46, 2004.
[20] C. Wang, X. Yuan, H. Zhang, B. Sun, and H. Zhang, "Structural Query and Ranking for XML Information Retrieval," *Journal of Computational Information Systems*, vol. 5, pp. 1429-1435, 2009.
[21] H. Zhang, X. Yuan, N. Yang, and Z. Liu, "Similarity Computation for XML Documents by XML Element Sequence Patterns," Progress in WWW Research and Development, pp. 227-232, 2008.
[22] C. Wang, X. Yuan, H. Ning, and X. Lian, "Similarity Evaluation of XML Documents Based on Weighted Element Tree Model," *Advanced Data Mining and Applications*, pp. 680-687, 2009.
[23] A. Nierman and H. V. Jagadish, "Evaluating Structural Similarity in XML Documents," *in Proceedings of the Fifth International Workshop on the Web and Databases WebDB*, 2002.

Yong Piao, born in Liaoning province, China, 1975, received his B.A's and M.A's degrees in computer science from Dalian University and Technology, Dalian, Liaoning, China, in 1998 and 2001 respectively.

From 2002 to 2003, he made an advanced study in EI department, University of Siegen, Germany. Since 2004 he has been a lecturer, 2010 an associate professor, in Dalian University of Technology. His current research interests mainly cover database and decision support systems.

Xiu-kun Wang, born in 1945. She has been a professor in Dalian University of Technology. Her main research interests are data mining, database and decision support systems.