Applied Mathematics, 2015, 6, 1076-1085
Published Online June 2015 in SciRes. http://www.scirp.org/journal/am
http://dx.doi.org/10.4236/am.2015.66098

A Novel Method for Transforming XML Documents to Time Series and Clustering Them Based on Delaunay Triangulation

Narges Shafieian

Department of Computer Engineering, Zanjan Branch, Islamic Azad University, Zanjan, Iran
Email: nshafieian@gmail.com

Received 23 April 2015; accepted 7 June 2015; published 10 June 2015

Abstract

Nowadays exchanging data in XML format become more popular and have widespread application because of simple maintenance and transferring nature of XML documents. So, accelerating search within such a document ensures search engine's efficiency. In this paper, we propose a technique for detecting the similarity in the structure of XML documents; in the following, we would cluster this document with Delaunay Triangulation method. The technique is based on the idea of representing the structure of an XML document as a time series in which each occurrence of a tag corresponds to a given impulse. So we could use Discrete Fourier Transform as a simple method to analyze these signals in frequency domain and make similarity matrices through a kind of distance measurement, in order to group them into clusters. We exploited Delaunay Triangulation as a clustering method to cluster the *d*-dimension points of XML documents. The results show a significant efficiency and accuracy in front of common methods.

Keywords

XML Mining, Document Clustering, XML Clustering, Schema Matching, Similarity Measures, Delaunay Triangulation, Cluster

1. Introduction

The main idea of this method is based on structure of XML documents; it means that, tags and position of elements in XML tree's hierarchy are considerable. So content of documents is not important, in other words, it may exist two documents with completely similar structure, based on our method but, completely different content.

Input of our method implementation is a set of documents and output is clustering these documents into various

clusters, and if clustering is done correctly, documents within a cluster have the same structure and documents belonging to different clusters have fewer structural similarity. The main contribution of our approach is these steps:

1) Mapping each documents to a time series;
2) Getting DFTs and transforming each time series from time domain to frequency domain;
3) Mapping the signals related to each documents to a point in *d*-dimensional space;
4) Triangulation of points related to documents;
5) Clustering documents based on their triangulation.

For analyzing accuracy of clustering, we use external metric. In analyzing based on external metrics, we would evaluate other clustering strategies in front of our proposed clustering method. Having more matches between clustering metric and other clustering, clustering may have more precision too. We use two external metrics named F-Measure and Purity as evaluator of our method. More information about this method is mentioned in [1].

The corpus of documents for evaluating this method is a standard corpus, which a part of that is applied. This corpus has clustering metric itself which we use it as a comparison versus our external metrics. This corpus could download from references mentioned in [2] and is defined in [1].

The rest of the paper is organized as follows: In Section 2, we present some information about common methods for detecting similarities and clustering documents. Section 3 expresses implementation requirements and developing environment. Section 4 illustrates how the structure of an XML document can be encoded into a time series, mapped to *d*-dimension space, triangulated and finally clustered; then it presents some methods for accomplishing such tasks. Section 5 describes several experiments we perform, on encyclopaedia data set, to validate our approach. We sketch some issues which could be faced in future work on this topic, and conclude the paper in Section 6.

2. Related Work Summary

Several methods for detecting the similarity of XML documents have been recently proposed, that is based on the concept of edit distance and use graph-matching algorithms to calculate a (minimum cost) edit script capable of transforming a document into another. Most of these techniques are computationally expensive, *i.e*. at least O (N3), where N is the number of element of the two documents. However, all of them are concerned with the detection of changes occurring in XML documents rather than comparing them on the basis of their structural similarity. Some approaches have a technique for measuring the similarity of a document versus a DTD is introduced. This technique exploits a graph-matching algorithm, which associates elements in the document with element in the DTD. This approach does not seem to be directly applicable to cluster documents without any knowledge about their DTDs, and is not able to point out dissimilarities among documents referring to the same DTD [3].

Indeed, we propose to represent the structure of an XML document as a time series, where each tag occurrence corresponds to an impulse. By analysing the frequencies of the Fourier Transform of such series, we can state the degree of (structural) similarity between documents. As a matter of fact, the exploitation of the Fourier transform to check similarities among time series is not completely new [3] and has been proven successful. The main contribution of our approach is the systematic development of an encoding scheme for XML documents, in a way that makes the use of the Fourier Transform extremely profitable.

Hence, after detection of documents similarity we could group documents into different clusters, which intensively accelerate search engine motors. Particularly, XML document clustering Algorithms divided into two groups:

- Pair wise methods.
- Incremental methods.

Pair wise based algorithms are more common which first create a similarity matrix for each pair of documents. This matrix is initialized by a criterion for measuring similarity between two documents. Finally, after completing the matrix we can use a general clustering algorithm such as K-means to locate a document in its proper cluster. In this paper we applied a new clustering method named Delaunay triangulation, which is used for clustering video frames. In contrast to many of the other clustering techniques, the Delaunay clustering algorithm is fully automatic with no user specified parameters [4].

3. Implements Requirements and Performing

Our proposed method developed by java and in order to run, need JRE or JDK (6 versions). Development environment is Eclipse (and using SAX, Flanagan library to implement some part of the program).

In one phase of project we need to triangulate some points in *d*-dimensional space, with Delaunay method. In consideration of large size of *d* (more than 3), this function extremely reduce efficiency. In order to increase efficiency, we use C++ language and CGAL library [5] for implementing this phase. This program is written and compile independently and it just triangulate points.

4. Clustering Phases

4.1. Mapping Each Documents to a Time Series

In this phase XML documents parsed one by one and a time series produced, it means that a set of numbers which are in time ordered, produced. For producing time series a special coding function applied which is effectively reflect documents structure, means tags and hierarchal structure of XML tree. The coding function has two parts. One of them is local and gives a unique identity to each tag. In order to make this identity for each tag we use a random order. With every visit of a new tag, the tag was added to a data base and an unused identity assigned to it. From now to, with visit of that tag, this local code was looked up in data base and being used.

In addition, local coding based on tags, we use a general coding function which, take attention to position of tags in hierarchy structure. This general coding, assign a special weight to each tags based on its depth and hierarchy structure. Detailed information about labelling documents is finding in [3]. Mapping documents to time series are done as below definition.

Definition 1. Let *D* be a set of XML documents, *d* a document in *D* with $sk(d)=[t_0,\cdots,t_n]$ and γ a tag encoding function for *D*. Moreover, let maxdepth (*D*) represent the maximum depth of any document in *D*, *B* a fixed value and $nest_d(t_n)$ the set of tag instances associated with the ancestors of the element with tag instance *t*. A multilevel encoding of *d* (mlemc (*d*)) is a sequence $[S_0,S_1,\cdots,S_n]$, where:

$$S_i=\gamma(t_i)\times B^{maxdepth(D)-l_{t_i}}+\sum_{t_j\in nest_{d(t_i)}}\gamma(t_j)\times B^{maxdepth(D)-l_{t_j}} \quad (1)$$

We usually set *B* as the number of distinct symbols encoded by (e.g., $B=|tnames(D)|+1$ in the case of invariant γ_d). In this way, we avoid "mixing" the contributions of different nesting levels and can reconstruct the path from the root to any tag by only considering the corresponding value in the encoded sequence. In fact, the summation on the right-hand side of the above formula can be interpreted as the integer whose *B*-base representation is the sequence of the tag codes in $\{\gamma(t_j)|t_j\in nest_d(t_i)\}$, ordered by increasing nesting levels of the corresponding tags. Notice that such a property is stronger than WSL, and is not mandatory for guaranteeing injectivity in the encoding function.

For example in the documents below, we could realize that (a) and (b) documents are more similar to document (c). In order to approve it via their transferred format, we should notice the next phase.

4.2. Getting Discrete Fourier Transform (DFT) and Transferring Each Time Series from Time Domain to Frequency Domain

Time series produced in previous phase are in time domain means that these series reveal tags with its special structure, during time domain. The Length of these signals is different and comparison of them is difficult. In order to capture similarities and structural differences, signals transfer from time domain to frequency domain. So we could compare two signals magnitudes in specific frequency. This comparison reflected in structural differences between documents.

Consider **Figure 1**, representing the documents of **Figure 2**. Observe that all the signals have different shapes. Notwithstanding, the difference among the signals can be summarized as follows:

- Each book element is associated with a unique subsequence within the signals associated with book 1 and book 2. Nevertheless, the sub sequences number's occurrences are different.
- Book 3 has two different sub sequences associated with the book elements. Moreover, the first subsequence is different from the ones in book 1 and book 2.

A comparison in the time domain (accomplished using the time-warping distance) will result in a higher

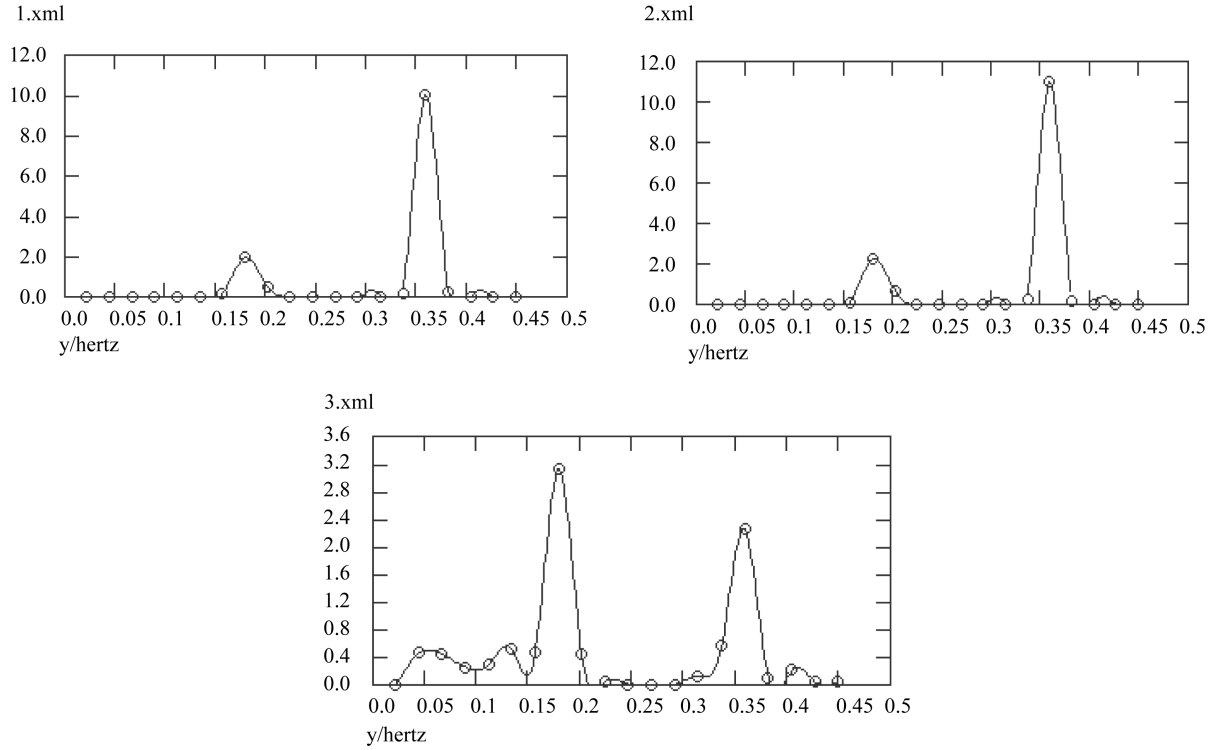

Figure 1. Nonzero frequency components of the book 1, book 2, and book 3 documents in **Figure 2**, Discrete fourier transform.

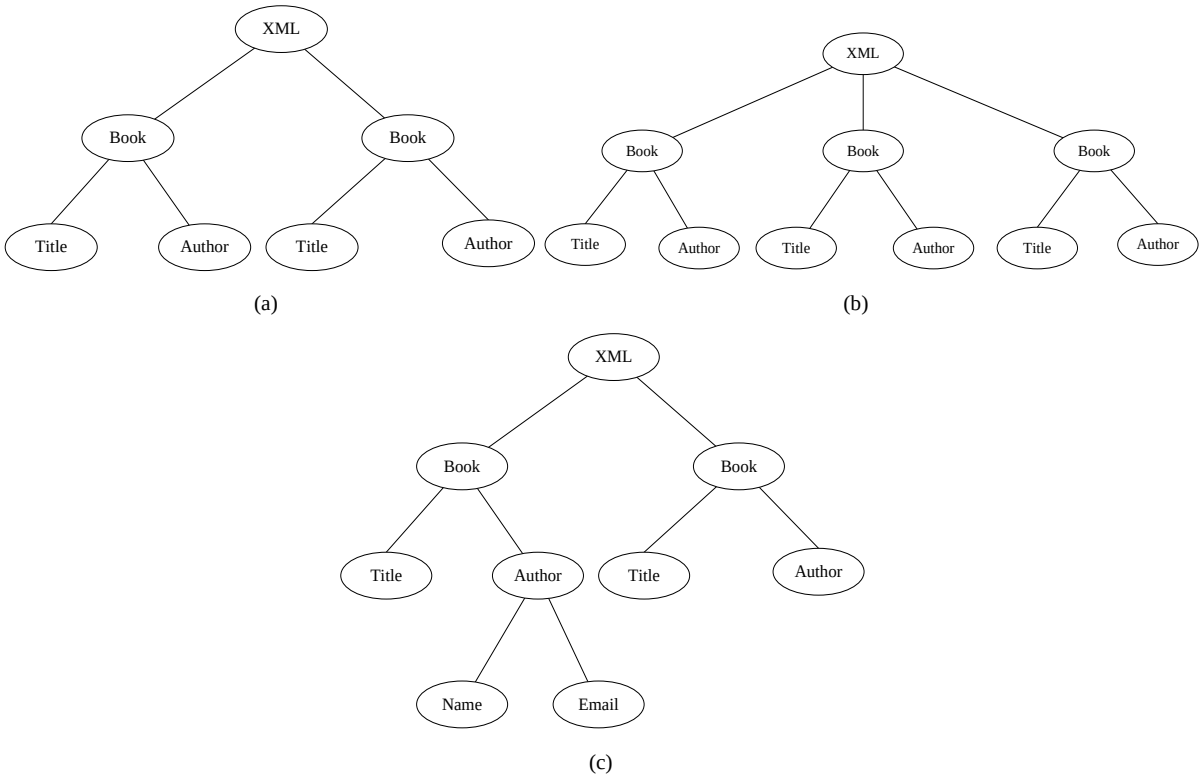

Figure 2. (a) Book 1 and (b) book 2 have the same elements, but with different cardinality. By contrast, (c) book 3 induces a different structure for the author element.

similarity between book 1 and book 3 than between book 1 and book 2. Nevertheless, each different subsequence triggers a different contribution in the frequency domain, thus allowing for detecting the above described dissimilarities. To better understand how the differences between two documents reflect on the frequency spectra of their associated encodings, we can always consider these differences separately and exploit the linearity property of the Fourier transform.

Definition 2. Let d_1, d_2 be two XML documents, and enc a document encoding function, such that $h_1 = enc(d_1)$ and $h_2 = enc(d_2)$. Let DFT be the Discrete Fourier Transform of the (normalized) signals. The exploitation of the Fourier transform to check similarities among time series is not new, for more information see, [3]. The main contribution of our approach is the systematic development of effective encoding schemes for XML documents, in a way that makes the use of the Fourier Transform extremely profitable. The choice of comparing the frequency spectra follows both effectiveness and efficiency issues. Indeed, the exploitation of DFT leads to abstract from structural details which, in most application contexts, should not affect the similarity estimation (such as, e.g., different numbers of occurrences of a shared element or small shifts in the actual positions where it appears). This eventually makes the comparison less sensitive to minor mismatches. Moreover, a frequency-based approach allows us to estimate the similarity through simple measures (e.g., vector distances) which are computationally less expensive than techniques based on the direct comparison of the original document structures.

We define the Discrete Fourier Transform distance of the documents as the approximation of the difference of the magnitudes of the DFT of the two encoded documents:

$$dist(d_1, d_2) = \left(\sum_{k=1}^{M/2} \left(\left| \left[D\tilde{F}T(h_1) \right](k) \right| - \left| \left[D\tilde{F}T(h_2) \right](k) \right| \right)^2 \right)^{\frac{1}{2}} \quad (2)$$

where $D\tilde{F}T$ is an interpolation of DFT to the frequencies appearing in both d_1 and d_2, and M is the total number of points appearing in the interpolation, *i.e.*, $M = N_{d_i} + N_{d_j} - 1$. **Figure 3** display the DFT value of each pair of a, b and c document, showing sequentially 1.xml, 2.xml and 3.xml, which calculated value obviously show similarity between documents, lower value means more similarity.

Hence, produced frequency signals, completely present documents structural differences, which is turned out in similarity matrix, too. Means that bigger value of the cell, related to documents, reveal more distance and also lower similarity.

In order to transferring time series from time domain to frequency domain, Discrete Fourier Transform (DFT) has exploited. We use JAVA library to perform this transferring. This library using FFT algorithm which have suitable time cost too. For more information refer to [3].

4.3. Mapping Signal Corresponding to Each Document to a Point in *d*-Dimensional Space

Finally we could sampling the signal consequence from previous phase and make a discrete signal. If sampling was done for the same frequency and signals magnitude compared in these positions, documents similarity can estimated. More sampling conclude to more accurate comparison and, in other hand, time cost of other remain calculation was increased. Thus, choosing a degree for sampling, have intensive influence to clustering efficiency and accuracy, so is a trade-off.

After this phase, each document mapped to signal point in *d*-dimension space, which *d* is the size of applied sampling. Now there are some points in *d*-dimension, which should be clustered. In other applications, we could

```
(1.xml, 1.xml) --> 0.0
(1.xml, 2.xml) --> 0.17660526638049864
(1.xml, 3.xml) --> 1.4970224941456434
(2.xml, 1.xml) --> 0.17660526638049864
(2.xml, 2.xml) --> 0.0
(2.xml, 3.xml) --> 1.6347015777026328
(3.xml, 1.xml) --> 1.4970224941456434
(3.xml, 2.xml) --> 1.6347015777026328
(3.xml, 3.xml) --> 0.0
```

Figure 3. Similarity matrix corresponds to (a), (b), (c) documents.

map component of videos, images, sounds and etc. to points and use a clustering method for them too. Afterwards, we could compare this clustering method with others. In other word, we could give these points to them and compare the results. Each point can be as a feature vector.

4.4. Triangulate Points Corresponding Documents

From this phase, points correspond to documents is in *d*-dimension, triangulation was exploited because of following reasons: inside of produced triangles was no point or points. It means that each point is at the corner of one or more triangles but not in the any triangle. This good feature is desirable for high efficiency clustering which, is defined in the following part.

Notice that triangle are not only in only two-dimension and be any shape like pyramid and etc. in higher dimensions. In *d*-dimension space each triangle made up $d + 1$ corner. Consequence of this phase is a graph which vertex are points and its edges are sides of triangles produced from this triangulation method. The graph is saved as a file till, clustering algorithm used it. Triangulation in incremental way was implemented by CGAL library, which is complicated and when the dimension be higher, it's done very slowly. Other triangulation methods exist for Delaunay but their problem was dimensions, too. Triangulation was done as below definitions.

The formal definitions for the mean edge length and local standard deviation for each data point follows from Definitions 3 and 4.

Definition 3. The mean length of edges incident to each point p_i is denoted by Local_Mean_Length (p_i) and is defined as

$$\text{Local\_Mean\_Length}\left(p_i\right)=\frac{1}{p\left(p_i\right)}\sum_{j=1}^{d\left(p_i\right)}\left|e_j\right| \tag{3}$$

where $d(p_i)$ denotes to the number of Delaunay edges incident to p_i and $|e_j|$ denotes to the length of Delaunay edges incident to p_i.

Definition 4. The local standard deviation of the length of the edges incident to p_i is denoted by Local_Dev(p_i) and is defined as

$$\text{Local\_Dev}\left(p_i\right)=\sqrt{\frac{1}{d\left(p_i\right)}\sum_{j=1}^{d\left(p_i\right)}\left(\text{Local\_Mean\_Length}\left(p_i\right)-\left|e_j\right|\right)^2} \tag{4}$$

To incorporate both global and local effects, we take the average of local standard deviation of the edges at all points in the Delaunay diagram as a global length standard deviation as defined in Definition 5.

Definition 5 the mean of the local standard deviation of all edges is denoted by Global_Dev(*P*) and is defined as

$$\text{Global\_Dev}\left(P\right)=\frac{1}{N}\sum_{i=1}^{N}\text{Local\_Dev}\left(p_i\right) \tag{5}$$

where *N* is the number of total points and *P* is the set of the points.

All edges that are longer than the local mean length plus global standard deviation are classified as inter-edges (Definition 7) and form the separating edge between clusters. The formal definition for short and separating edges in terms of mean edge length of a point and mean standard deviation are captured in Definitions 4 and 5 below.

Definition 6. A short edge (*intra-cluster edge*) is denoted by Short_Edge(p_i) and is defined as

$$\text{Short\_Edge}\left(p_i\right)=\left\{e_j\,\middle|\,\left|e_j\right|<\text{Local\_Mean\_Length}\left(p_i\right)-\text{Global\_Dev}\left(P\right)\right\} \tag{6}$$

Definition 7. A Separating edge (*inter-cluster edge*) is denoted by Separating_Edge (p_i) and is defined as

$$\text{Seprating\_Edge}\left(p_i\right)=\left\{e_j\,\middle|\,\left|e_j\right|>\text{Local\_Mean\_Length}\left(p_i\right)-\text{Global\_Dev}\left(P\right)\right\} \tag{7}$$

Delaunay Triangulation can be done effectively in $O(n\log n)$ time and the identification of inter and intra edges can be done in $O(n)$ time where *n* is the number of documents processed. For detailed information refer to [4].

4.5. Clustering Documents Based on Their Triangulation

The triangles obtained from previous phase are applied for clustering. The triangulation method has a feature that, having no point inside of any triangles, was guaranteed. With this key feature, analysing every sides of any tri-

angles to recognizing nearest and farthest points, could be possible.

Hence, we could disconnect links between points of graph by deleting fairly big edges. Finally, Graph was divided into some individual sections. Each of them made a new cluster. Recognizing fairly big edges, was done in the way was illustrated in [4].

We use a parameter for control deleting bigger edges. This parameter that named clustering factor, demonstrating what rate of edges deviations expressed the edges should be deleting. This way is fairly complicated for, length mean of every edges end up to each vertex, their standards deviations and average of all standard deviation, should be computed.

Finally some file produced which, each of them contain some documents, were inside of a cluster. With higher value of clustering parameter, number of produced cluster became bigger. This parameter should choose in the way that suitable number of clusters been produced. We could do clustering more times for each set of XML documents to find its suitable clustering factor value.

5. Clustering Evaluation's Parameters and Notifications

As mentioned before, for evaluating accuracy of proposed method, we use two parameters named Purity and F-Measure. These two parameters computation method was illustrated in [1]. For computing the parameter which was in external type, a clustering metric was needed. Corpus of the used data set has this metric, this corpus explained in [6] as this:

Closer value of these parameters to 100 percent means clustering accuracy rate was higher, and in verse. Clustering a set of documents with proposed method expressed that, these parameter isn't high enough. Mean that, this kind of clustering method suitable for clustering the data set.

Three parameters have influenced to clustering efficiency and accuracy, involved:

- Number of sampled points from documents frequency: more number of points is sampling from documents frequencies (produced diagram of transferring time series from time domain to frequency domain), made documents comparisons better. This parameter expressed the points' space dimensions. For intensive inefficiency triangulation in higher dimensions, increasing number of sampling made clustering run slowly; which also, exclude to lower performance in triangulation and clustering.
- Clustering parameter: higher value of this parameter, made number of produced clusters higher. This parameter defined expressed which edges are fairly big. We could find out the suitable value of this parameter by doing some trial and error. However, the value of this parameter is not influenced to the method efficiency but is influenced to consequence's accuracy. The parameter should respectively that, number of produced clusters be about the number of clusters in clustering metric.
- β parameter: This parameter is applied in clustering evaluation, (for F-Measure calculation). Setting this parameter with 5 is a good choice, in reality, we could assign other values to it, which is finally didn't impact on clustering efficiency. Notice that, in evaluating other methods based on this F-Measure, the parameter must be fixed.
- Number and length of documents, tag's length and kinds: these parameters influenced to clustering efficiency, especially, in parsing documents level means in tree structural presentation, so it expressed overall efficiency. User couldn't choose them but, could choose the set of documents.

6. Experimental Results

We have two kinds of data sets in evaluation, one used in [2], is English Single Label Categorization Collection-XML Document, which have 10 categories of documents, each category include 10 documents. And synthesize data set which is produce by piecing the set of tags with different depth and length but single subject together in a single category; we need to use of 5 categories include 10 documents in each category.

The below figures presents the results of running the system, on both real in **Figure 4** and synthesizes data sets in **Figure 5** for different value of clustering factor and also single dimension. (x-axis reveals different clustering factor and y-axis reveals calculated value of number of clusters, Purity and F-Measure by precent.)

In **Figure 6**, we compare common methods of clustering XML documents with proposed system on execution time (x-axis reveals number of xml documents and y-axis reveals calculated execution time by msec), and we find out, our method due to using transferring to time series approach, getting the best information about the

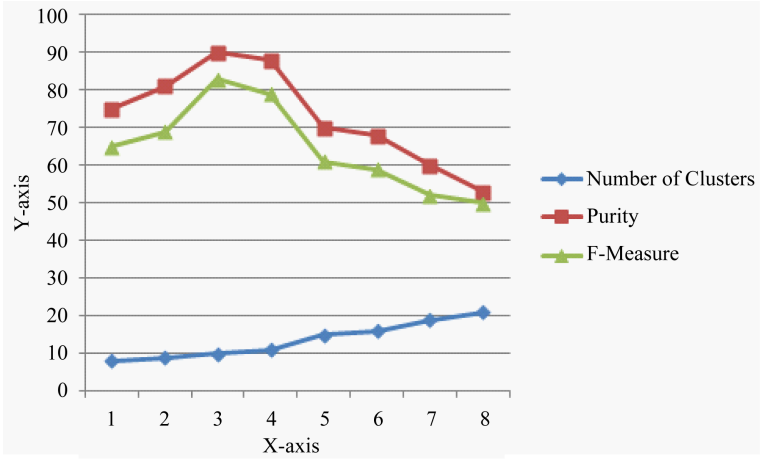

Figure 4. Diagram of running the system on real dataset in different value for clustering factor.

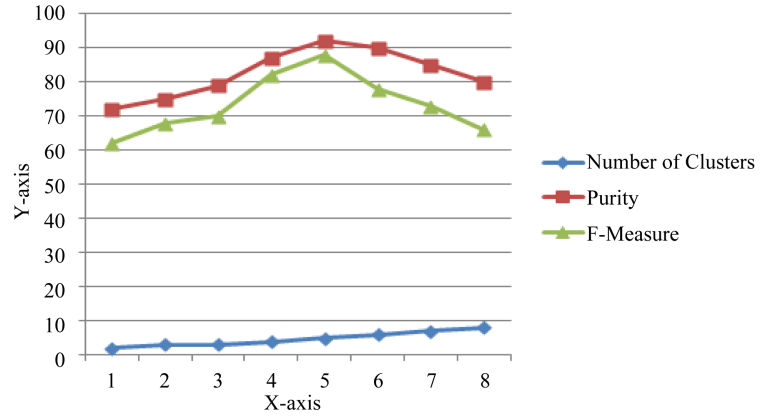

Figure 5. Diagram of running the system on synthesize dataset in different value for clustering factor.

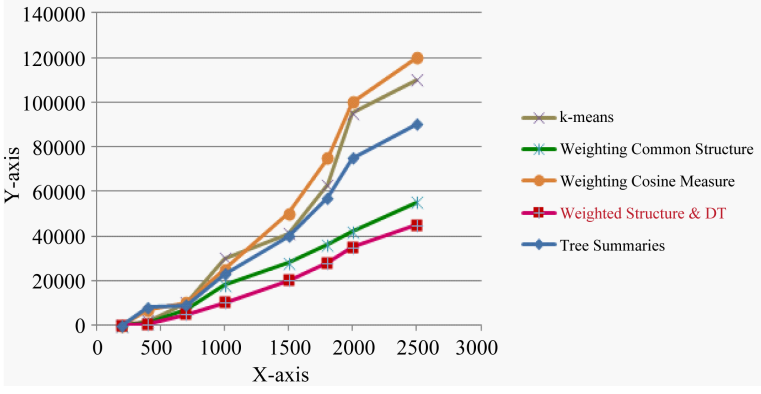

Figure 6. Execution time comparison on varying number of XML documents in msec.

documents very quickly, so it can have less time to clustering documents too. In verse of other clustering methods, likes catching common structure in document's tree, which needed to search the entire document.

Below figures, present result of running common and proposed system on different number of documents. (x-axis reveals number of XML documents and y-axis reveals calculated Purity by percent in **Figure 7** and also

x-axis reveals number of XML documents and y-axis reveals calculated F-Measure by percent in **Figure 8**), Diagrams reveal proposed method do the best in a constant value of dimension and clustering factor parameter, as we expected.

7. Conclusions and Future Works

Evaluation results are desirable enough. Clustering metrics are based on subject and content of documents. But our proposed method clusters them based on structure of documents. So comparing this clustering method with clustering metric is suitable enough but not completely admirable. In fact, it should be a clustering metric based on documents structure to exploit in evaluating, or instead of using external metric, internal metric used. In order to evaluate clustering, we could analyse the distance of the cluster's intra points with each other and with others in other clusters. This criterion named internal metrics.

Efficiency of the proposed method depended on triangulation's efficiency which, in high dimension had lower performance or even impossible. On the other hand, space dimensions greatly affect the comparison accuracy. As a result, we recommend a simpler method, but for a smaller number of points used in high dimensions that can be used as a suitable alternative. This means that the use of methods other than Delaunay, sampling parameter not only will enhance, but also will gain good results.

Acknowledgements

This work is a revised, extended version of the paper that appeared in [7] [8]. The authors are very grateful to

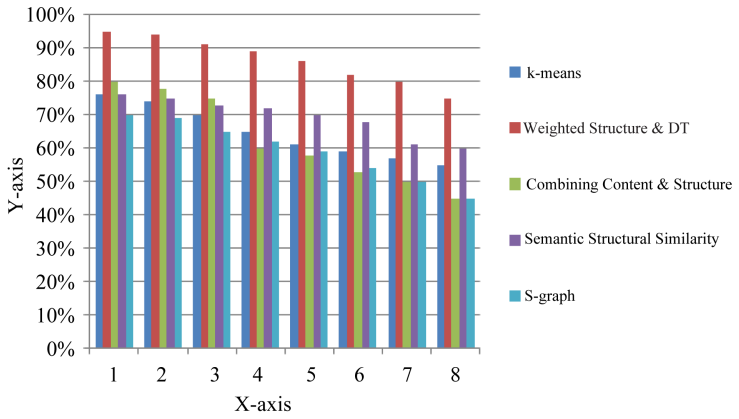

Figure 7. Purity of varying methods on different number of XML documents.

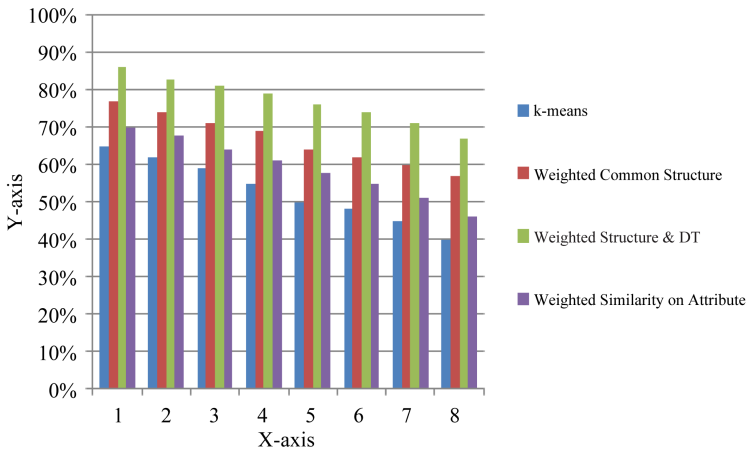

Figure 8. F-Measure of varying methods on different number of XML documents.

Michael Thomas Flanagan for providing Java Scientific Library. This work was exploited other standard java library like SAX, moreover, applied CGAL and also boost library for coding C++ part of the program.

References

[1] Hwang, J. and Ryu, K. (2010) A Weighted Common Structure Based Clustering Technique for XML.

[2] Algergawy, A., Nayak, R. and Saake, G. (2010) Element Similarity Measures in XML Schema Matching.

[3] Flesca, S., Manco, G., Masciari, E., Pontieri, L. and Pugliese, A. (2005) Fast Detection of XML Structural Similarities. *IEEE Transactions on Knowledge and Data Engineering*, **7**, 160-175. http://dx.doi.org/10.1109/TKDE.2005.27

[4] Mundur, P., Rao, Y. and Yesha, Y. (2006) Key Frame Based Video Summarization Using Delaunay Clustering. *International Journal on Digital Libraries*, **6**, 219-232. http://dx.doi.org/10.1007/s00799-005-0129-9

[5] Fabri1, A., Giezeman, G., Kettner, L., Schirra, S. and Schonherr, S. (1999) On the Design of CGAL a Computational Geometry Algorithms Library.

[6] Denoyer, L. and Gallinari, P. (2006) The Wikipedia XML Corpus.

[7] Yuan, J.-S., Li, X.-Y. and Ma, L.-N. (2008) An Improved XML Document Clustering Using Path Feature. 5*th International Conference on Fuzzy Systems and Knowledge Discovery*. http://dx.doi.org/10.1109/fskd.2008.66

[8] Naresh, N. and Ashok, B. (2010) Clustering Homogeneous XML Documents Using Weighted Similarities on XML Attributes. *IEEE* 2*nd International Advance Computing Conference*, Patiala, 19-20 February 2010, 369-372.