AN OVERVIEW OF FRAGMENTATION DESIGN FOR DISTRIBUTED XML DATABASES

Kok-Leong Koong, Su-Cheng Haw and Lay-Ki Soon

Faculty of Computing and Informatics,
Multimedia University, 63100 Cyberjaya, Malaysia

ABSTRACT

XML is a standard of data exchange between web applications such as in e-commerce, e-learning and other web portals. The data volume has grown substantially in the web and in order to effectively retrieve or store these data, it is recommended to be physically or virtually fragmented and distributed into different nodes. Basically, fragmentation design contains of two parts: fragmentation operation and fragmentation method. There are three different kinds of fragmentation operation: Horizontal, Vertical and Hybrid, determines how the XML should be fragmented. The aim of this paper is to give an overview on the fragmentation design consideration.

KEYWORDS

XML Database, Distributed Design, Fragmentation Distributed XML

1. INTRODUCTION

XML is a semi-structured, self describing and human-readable document. A native XML document is stored in a plain text format and thus it can be easily processed by any applications and systems. XML and HTML are both subset of Standard Generalized Markup Language (SGML) [1]. And, HTML is commonly used in web environment. It makes XML a good option for data exchange in web environment. Thus, XML has started to become a standard of data exchange between applications and systems. It has been extensively used in web environment and data exchange between web applications. However, as the nature of XML, it is also commonly used in standalone applications to store metadata or application data.

The emerging of smart phone and tablet market has generated big volume of data and it grown exponentially in every minute. This gigantic volume of data also has been named as Big Data. The cohesiveness between these data is low as data might or might not be related to each other. Thus, XML is a good choice to be used to handle these data. However, large volume data will be only effective to be stored and retrieved in distributed model as it can be making used of the parallelism processing.

There are three main advantages on distributed large database. First of all, a distributed system may require multiple normal specification computer system rather than a very high specification computer system. Thus, it will lower the cost but sustain the high performance on the distributed database. Secondly, it also increased scalability. There is always a boundary for a database to

David C. Wyld et al. (Eds) : CCSIT, SIPP, AISC, NLP - 2015
pp. 41–52, 2015. © CS & IT-CSCP 2015 DOI : 10.5121/csit.2015.50404

expand within a single computer system. When it is design to be distributed, the database can expand beyond a single computer system. Thirdly, it will increase the availability. Normally, distributed design database will be replicated. This will make the database more resistance to the failure of a single computer system [2]. Thirdly, it will increase the performance of the database system as it used parallelism processing to store and retrieve data from the database system [3]. Distributed design of database normally includes three basic steps: fragmentation, allocation and replication [4]. Nevertheless, the focus on this paper is on fragmentation. Fragmentation is a process of divide database into smaller fragments. Fragmentation contains two steps: determine a fragmentation model to be used and select a method or an algorithm to use for the fragmentation. In the first step, it determines what structure or model of fragmentation to be used. It can be horizontal, vertical or mixed. In the second step, it determines how the data should be fragmented into fragments. It also sometimes refers to fragmentation method or technique.

The rest of the paper is organized as follows. Section 2 outlines the factors driven to distribution database. Section 3 gives an overview on fragmentation models, followed by Section 4, which discusses on the fragmentation methods. Section 5 presents our discussion. Finally, Section 6 concludes the paper.

2. FACTORS DRIVEN TO DISTRIBUTION

Main driving forces for having distributed database include:

- Lower costs: having distributed architectures made of hundreds of PC computers proves to be much cheaper and even more powerful the one mainframe systems serving hundred terminals
- Increased scalability: adding a new network node is the easiest way to respond to extensibility needs of the company,
- Increased availability: by replicating data over several network nodes data becomes closer to the end user and more resistant to system failures,

3. FRAGMENTATION MODEL

3.1. Fragmentation Model for Traditional Databases

There are three basic types of fragmentation models in traditional databases such as relational database and object oriented database. There are horizontal, vertical and mixed [5].

In the relational database, horizontal fragmentation referring a fragmentation database at the record, row or tuple level [3, 6]. To illustrate the scenario, assume a simple relational database contains the following fields in each record: *name, gender, address, phone, income* and *tax_id*. There are 56,000 records are stored in a single table for recording 56,000 person data (Table 1). A simple horizontal fragmentation might result into the first node storing the first 28,000 records and the second node storing the last part of 28,000 records. The structure of the fragmentation will be look similar to Figure 1.

Table 1. Sample data of Person Table

name	gender	address	phone	income	tax_id
xxx	x	xxxx	xxxx	xxxx	xxxx
yyy	y	yyyy	yyyy	yyyy	yyyy
……					
Ooo	o	oooo	oooo	oooo	oooo

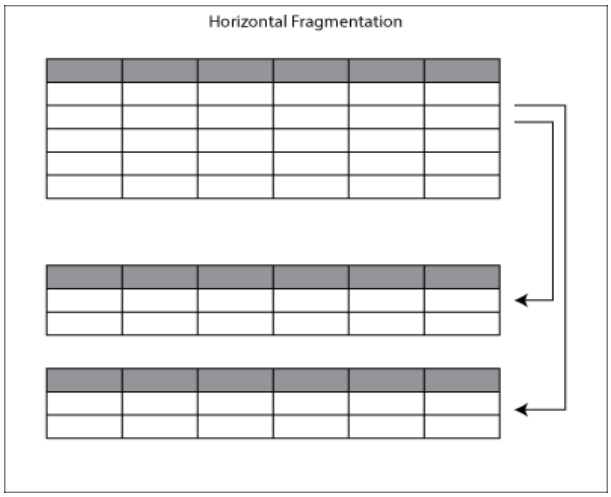

Figure 1: Horizontal Fragmentation for relational database

On the other hand, vertical fragmentation referring a fragmentation database by grouping fields or attributes of records. Using the previous relational database show on Table 1, vertical fragmentation will split this database by grouping fields such that it might group *name, gender, address* and *phone* fields and store in first node, while *income* and *tax_id* into the second node. The fragmentation structure will be look similar to Figure 2.

The mixed or hybrid is a combination of both horizontal and vertical fragmentation. It can be split horizontally then vertically or vice versa. Using the same relational example, a mixed can first split horizontally by grouping records that belong to particular level of income. Then, split further on the current records by splitting *name, gender, address* and *phone* on other node and the rest of the data *income* and *tax_id* on other node.

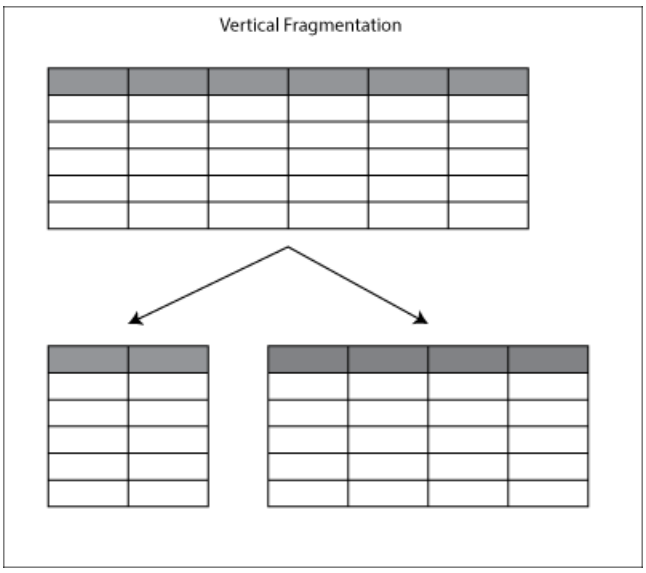

Figure 2: Vertical Fragmentation for relational database.

Object oriented database is totally different from relational database. The data is stored in object form and can be illustrated in a hierarchical or tree format. Fragmentation in object oriented has increased complexity of its hierarchical structure, methods or properties within an object [7]. In term of structure, XML is quite similar to object oriented database. Fragmentation in object oriented share the same fragmentation model like relational database aside the complication involved in object oriented database. It can be fragmented in horizontal, vertical or mixed. Figure 3 and Figure 4 shown the concept how object oriented database can be fragmented into horizontal and vertical model respectively.

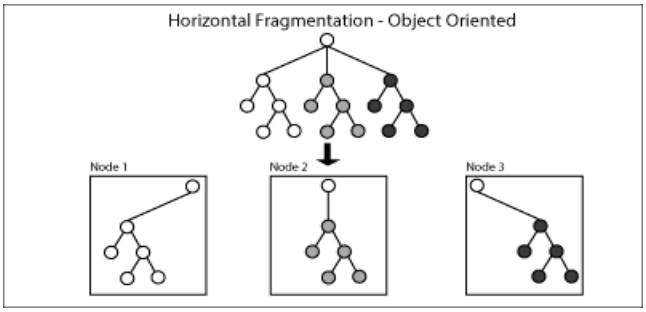

Figure 3: Horizontal Fragmentation for object oriented database

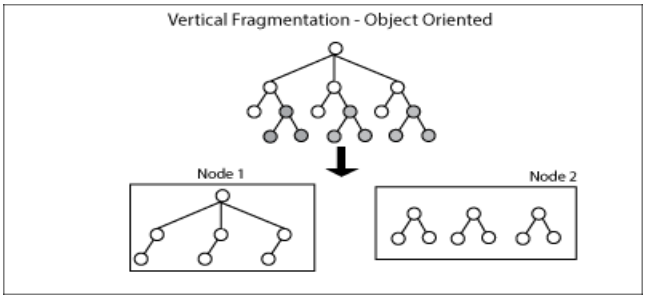

Figure 4: Vertical Fragmentation for object oriented database

3.2. Fragmentation Model for XML Databases

In general, there are only three types of fragmentation models: horizontal, vertical and mixed in XML distributed design [6]. As relational database and object oriented database has started to develop distributed design earlier than XML, the fundamental concept in XML fragmentation also referencing to these two databases. Initially, XML is introduced to run in a single machine. However, as the size of the data grow substantially and it needs to be distributed form in order to achieve better performance on retrieving and storing the data.

Generally, the fragmentation models can be broadly classified into Horizontal, Vertical and Hybrid. The following subsections briefly explain each model.

3.2.1. Horizontal Fragmentation

In XML, horizontal fragmentation can be achieved by selection. Selection is based on the pre defined conditions on splitting the fragments. A horizontal fragment f_i is determined by the selector operator σ of predicates p over collection of elements E in a homogeneous XML document. It can be written so that $f_i = E(\sigma_{pi})$. Assume we have a XML document constructed according to the relational database stated in the previous section. If the simple selection predicate of p1 such that /person/employee/income to be income level less than or equal 5000 and p2 to be income level more than or 5000, thus fragments will be written as $f_1 = E(\sigma_{p1})$ and as $f_2 = E(\sigma_{p2})$.
From Figure 5, employee elements with the of name Wong Wei Wei and Lee Jia Fong will be then split and stored as a new XML document in node 1 as first fragment and the rest of the elements of employee will stored in node 2 as a new XML document.

After the operation, node 1 and node 2 may have DTD like <!DOCTYPE person (employee*)> and <!ELEMENT employee (name, gender, contacts, income, tax_id)>.

Horizontal fragmentation is recommended when the query criteria is based on particular attribute that used as selection predicate to fragmenting the XML database. In this scenario, horizontal fragmentation may reduce the transportation cost and processing time as the data is determined in a specific distributed note. Moreover, horizontal fragmentation can easily transport data between sites to improve system performance [8].

Using the same XML database in this document as an example, we use /person/employee/income as the attribute for selection predicate to fragment the database horizontally. Assume this XML database has been fragmented into 5 nodes with income level as the selection predicate of the following categories: 0-999, 1000-1999, 2000-2999, 3000-3999, 4000 and above. If a query searching for a person detail information with income level of 3000-3999, these data can be obtain by querying the fourth distributed nodes thus the query will be able to locate the data in minimum time and retrieve the data with lease processing time.

3.2.2. Vertical Fragmentation

Vertical fragmentation can be achieved by projection. It will split the data structure into smaller parts as particular selected child elements will be split and stored as fragment in other node. A vertical fragment f_i is determined by the projection operator π by path selection ρ over collection of element E in a homogeneous XML document. It can be written so that $f_i = E(\pi_{\rho i})$. If the path selection ρi is /employee/contact, all the children elements under this tree path will be split and stored in other node. In this case, fragment $f1 = E(\pi_{\rho 1})$ represents all contact elements in the XML document will be split and stored in node 2. And, the remaining elements will be stored in node 1.

After the operation, node 1 may have an DTD like <!DOCTYPE person (employee*)> and <!ELEMENT employee (name, gender, income, tax_id)>. And, node 2 may have an DTD like <!DOCTYPE contacts (contact*)> and <!ELEMENT contact(address,phone)>. In order to create reference link between these two nodes, at least one reference attribute is required for the element that will able to refer back to elements that resided in other node or site [9].

Vertical fragmentation is a kind of affinity-based fragmentation. As opposed to horizontal fragmentation, this type of fragmentation does not encourage transportation of data from node to node which will trade off flexibility to affinity [8].

Assume a particular employee data is needed with a provided phone contact as search criteria. First the contact elements in the node 2 that match search criteria will be searched. If this entry found, the reference attribute will be used to access the employee data in node 1.

3.2.3. Hybrid Fragmentation

Hybrid fragmentation or sometimes also referring to mixed fragmentation uses both horizontal and vertical fragmentation by taken advantages of both models. It operates in the way where a horizontal fragmentation will be implemented to split the document into horizontal fragment and then further fragmented from these fragment by implementing vertical fragmentation.

A hybrid fragment f_i is determined by the horizontal and vertical fragmentation implemented. It is depend on how you would like to implement the hybrid into the XML document. It can be split horizontally then vertically or vice versa.

Assume you would like to do it horizontally then vertically. First fragment the document horizontally and called this fragment f_a. Thus, $f_a = E(\sigma_{pi})$ and from these f_a fragments we further fragmented them vertically such that the hybrid fragment $f_i = f_a(\pi_{\rho i})$.

Assume we use income level as the selection condition in horizontal fragmentation, and vertically fragment further with the path /employee/contact as previous example. There will be 4 hybrid fragments generated for 4 nodes.

After the operation, node 1 and node 2 may have DTD like <!DOCTYPE person (employee*)> and <!ELEMENT employee (name, gender, income, tax_id)>. Node 3 and node 4 may have DTD like <!DOCTYPE contacts (contact*)> and <!ELEMENT contact(address,phone)>. It will look exactly like in vertical fragmentation as it final operation is using vertical fragmentation. However, each node contains only two records instead of four records using vertical fragmentation.

Hybrid fragmentation is the combination of horizontal and vertical fragmentation which getting advantages of both fragmentations. In the above scenario, the search can be limited only to particular income level. At the same time, data can be also obtained from vertical fragments by contact element and then with the reference link to the particular employee.

```
<?xml version="1.0" encoding="utf-8"?>
<person>
    <employee>
        <name>Wong Wei Wei</name>
        <gender>Female</gender>
        <contact>
            <address>Sungai Besi</address>
            <phone>03-91234567</phone>
        </contact>
        <income>3500</income>
        <tax_id>120000</tax_id>
    </employee>
    <employee>
        <name>Lee Jia Fong</name>
        <gender>Male</gender>
        <contact>
            <address>Batu Pahat</address>
            <phone>07-71234567</phone>
        </contact>
        <income>2500</income>
        <tax_id>120001</tax_id>
    </employee>
    <employee>
        <name>Tan Jung</name>
        <gender>Male</gender>
        <contact>
            <address>Genting</address>
            <phone>05-34564567</phone>
        </contact>
        <income>6000</income>
        <tax_id>120002</tax_id>
    </employee>
    <employee>
        <name>Fan Wei Tong</name>
        <gender>Female</gender>
        <contact>
            <address>Kuala Lumpur</address>
            <phone>03-71234567</phone>
        </contact>
        <income>9000</income>
        <tax_id>120003</tax_id>
    </employee>
</person>
```

Figure 5: XML sample

4. FRAGMENTATION METHODS

Fragmentation model only define the fragmentation structure in distributed design. Fragmentation method is required to determine how the data should be fragmented (horizontally,

vertically or hybrid). Fragmentation by arbitrary cutting document in to fragments horizontally, vertically, or hybrid may not necessary effectively improve the query performance. Thus, some fragmentation methods have been introduced. These proposed methods have their own advantages and disadvantages against difference scenario. These methods can be group into four categories: structure and size, query and cost, predicate and holes and fillers (for streamed data).

4.1. Structure and Size

Fragmentation of XML document can be fragmented based on structure and size of the document. The structural information about the document can be obtained from the document schemata (DTD or XML Schema). The structure information also can be obtained by transverse the XML document. There is an advantage of this fragmentation method which balanced the load of site or nodes processing power. And this will lead to more effectively uses of resource and improve query performances.

Skewed query processing problem is a well known problem in distributed design. It merely indicated an unbalance on loading on particular distributed node against other nodes. And, this method of fragmentation can resolve particular skewed query processing problem as the fragment is properly distributed according to the structure and size of the document.

To fragment document using structure and size method, first of all, the document is required to be parsed. In other words, map the XML document into a tree structure. This parser is either tree-based or event-based. A tree-based parser may consume memory resources as it transverse the whole document and save all the relationship and node of these nodes in the memory. DOM is a tree-based parser. On the other hand, event-based consumed less memory. It does not construct a large tree in memory as it only scan particular element, attribute, content sequence in an XML document [10].

In structure and size method, event-based parser is used to construct vertex/node list, structural information. After obtained this information, the document then fragmented accordingly.

A typical example using this method on horizontal fragmentation can be achieved by determining a threshold size of the each fragment. Then, transverse throughout the XML document by determine the size of a single level child node horizontally. If the size of the child node including its descendants is smaller than the threshold size then continue on the next sibling child and so on until reaching the threshold size. These child nodes then will be created as a fragment and store in a distributed node or site as illustrated in figure 6. This scenario is vulnerable to skewed query process problem if particular fragment loading is much higher than other fragments.

Angela et al. proposed a simple top-down heuristics fragmentation method called SimpleX [11]. In order to fragment document using this method, three criteria are required to determine before hand: tree-width constraint, tree-depth constraint and tree-size constraint. These criteria will restrict the size of fragment. Fragment is determined when transverse down from the root element to the leaf elements (top-down). Fragment will be decided upon sub tree size that fulfils the tree-size, tree-width and tree-depth constraint. Then, structure histogram is constructed to evaluate how efficient is the fragmentation generated.

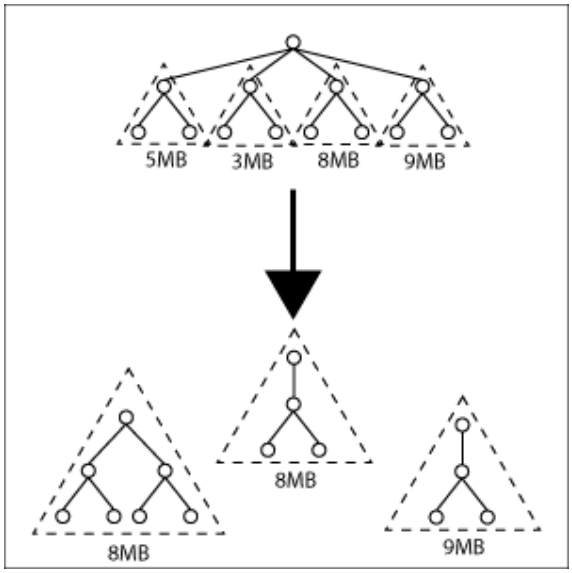

Figure 6: Fragmented by size

4.2. Query and Cost

Fragmentation of XML document can also be fragmented based on XML queries. The most common criteria to determine the fragment using this method are query frequency and the cost of query.

Leykun et al. proposed vertical fragmentation model based on queries. In their approach, two components are required to be set up for the fragmentation: Most frequently used queries with their frequencies, Element Usage Matrix (EUM) and Element Affinity Matrix (EAM) [12].

In this proposed method, it will analyze the total data access in the distributed system to determine the most frequently used queries and its frequencies. A matrix then will be constructed based on the elements access and queries. After EUM, another matrix called EAM is constructed. This matrix illustrated the relationship between elements against the queries requested. Finally, Grouping Heuristic Module is used to group elements and Splitting Heuristic Module will determine the fragment point for the fragmentation.

Ma et al., however, proposed method using heuristic to effectively fragment the XML document in horizontal fragmentation model. This method contains four steps. First of all, a horizontal fragmentation is constructed based on simple selection predicate. A query tree or query plan is build on this distribution design. From the query tree, the total execution query costs have to be determined. The query cost is the summation of storage costs and transportation costs. Storage cost is a measure of time in retrieving data from secondary storage. And, transportation cost is a measure in time for transverse time on XML document at different sites. Finally, the minimum total query cost will determine how the document should be fragmented [13, 14] Sven et al. proposed simplified cost model that work similar to previous method. The query processing cost model is based on the size estimation on the query results and query processing costs to determine the fragmentation of XML document [8].

4.3. Predicates

Predicates are commonly used in horizontal fragmentation model. There are two types of predicates: simple selection predicates and normal selection predicates.

The simple selection predicate takes the form of *path* θ *v*. θ is the comparison operator which belong to the subset of {<. >, =, ≠, ≤, ≥,..}. *path* is the path expression in XML and *v* is the value [15].

Predicate in relational database is differed from XML. In relational database, predicate indicate value of the fields. However, predicate in XML is indicated by path expression. In the previous example, predicate in relational database can be stated as income >= 5000. In XML, it then express in the form of /person/employee/income >= 5000.

4.4. Holes and Fillers

Holes and Fillers is a fragmentation method uses in Ad-hoc fragmentation. Ad-hoc fragmentation is fragmentation model for stream data. It does not required document schema for document fragmentation. Fragment in this model is fragmented and mark with special identifier for reconstruction later [4].

XFrag is the framework used in holes and fillers fragmentation method. In this method, the original document is break into smaller part (fillers) and one or more holes resided in filler with special tag and contains ID to corresponding filler.

5. DISCUSSION

Structure and size fragmentation method will fragment document according to the defined structure and size of XML document. The advantage of this method will distribute the content evenly across the distributed platform. However, it does not mean effectiveness in query processing response time. Skewed query processing problem is a common problem in this fragmentation method if the query processing concentrating only on particular site or distributed nodes.

The advantage of query and cost method is the most efficient method but the fragmentation cost is higher than other two methods.

Simple selection predicate is the most fundamental fragmentation method. It works fine in fragmented large XML document into smaller pieces to reduce search time and processing power on large XML document. However, it does not work efficiently compare to the query based methods.

6. CONCLUSIONS

There are pros and cons on different fragmentation models and fragmentation methods. However, heuristic is a method that can improve the query performance by study the usage of the distributed XML database. A horizontal fragmentation based on simple selection predicate method can be improved by study the query cost. According to the study to create a better fragmentation that will greatly optimize the query performance [14]. Another example of optimizing performance with its top-down heuristic is the SimpleX on structure and size fragmentation method [11].

With XML becoming the dominant standard for data exchange between various systems and databases on the Web, distributed XML is becoming crucial. In this paper, we have reviewed the types of fragmentation operations and fragmentation methods. As the result, we have also suggested the grouping for the fragmentation method.

ACKNOWLEDGEMENTS

This work is supported by funding of Fundamental Research Grant Scheme, from the Ministry of Higher Learning Education (MOHE).

REFERENCES

[1] S.C. Haw, C.S. Lee (2011). Data storage practices and query processing in XML database: A survey. Knowledge Based System. 24(8). 2011, pp. 1317-1340.
[2] M Smiljanic, H Blanken, M Keulen, W Jonker (2002). Distributed XML Database Systems. P1-43.
[3] Iacob, N. (2011). Fragmentation and Data Allocation in the Distributed Environments. Annals of the University of Craiova-Mathematics and Computer Science Series, 38(3), 76-83.
[4] Y.F. Huang, J.H. Chen (2001). Fragment Allocation in Distributed Database. Journal of Information Science and Enginnering 17, pp.491-506.
[5] V. Braganholo, M. Mattoso. (2014). A Survey on XML Fragmentation. ACM SIGMOD Record. 43(3).pp.24-35.
[6] P. R. Bhuyar, A.D.Gawande, A.B.Deshmukh (2012). Horizontal Fragmentation Technique in Distributed Database. International Journal of Scientific and Research Publication 2(5).pp.1-6.
[7] E. Malinowski, S. Chakravarthy (1997). Fragmentation techniques for distributing object-oriented databases.
[8] S. Hartmann, H. Ma, K.D. Schewe (2007). Cost-based vertical fragmentation for xml. In Advances in Web and Network Technologies, and Information Management (pp. 12-24). Springer Berlin Heidelberg.
[9] H. Ma, K. D. Schewe (2010). Fragmentation of XML Documents. In Journal of Information and Data Management, Vol. 1, No. 1, February 2010, pp. 21-33.
[10] Sall, K. B. (2002). XML Syntax and Parsing Concepts. Retrieved from Pearson: http://www.informit.com/articles/article.aspx?p=27006&seqNum=7
[11] A. Bonifati, A. Cuzzocrea, B. Zonno. (2006). Fragmenting XML Documents via Structural Constraints. Local Proceedings of ADBIS 2006,pp.17-29.
[12] L. Birahnu, S. Atnafu, F. Getahun. (2010). Native XML Document Fragmentation Model. 2010 Sixth International Conference on Signal-Image Technology and Internet Based Systems, pp.233 - 240.
[13] H. Ma, K.D. Schewe, S. Hartmann, M. Kirchberg. (2003). Distribution Design for XML documents. Journal of Information and Data Management, 2(1).pp. 21-33.
[14] H. Ma, K. D. Schewe (2005). Heuristic Horizontal XML Fragmentation. In CAiSE Short Paper Proceedings.
[15] H. Ma (2007). Distribution Design for Complex Value Databases, Ph(D) Thesis.

AUTHORS

Kok-Leong Koong received his Bachelor in Computer Science and Master in Business Administration in University of Central Oklahoma, U.S.A. in 1995. He is currently lecturer of Department of Information Sciences and Computing Studies in New Era University College. His major area researches are XML Databases, E-commerce, web application and computer network.

Associate Professor Dr. Su-Cheng Haw's research interests are in XML Databases and instance storage, Query processing and optimization, Data Modeling and Design, Data Management, Data Semantic, Constraints & Dependencies, Data Warehouse, E-Commerce and Web services.

Dr. Lay-Ki Soon received her Ph.D in Engineering (Web Engineering) from Soongsil University Korea in 2009. She is currently a Senior Lecturer in Faculty of Computing and Informatics, Multimedia University. Her research interests relate to Web science, which include Web crawling, Web data mining and social network analysis. She is involved in numerous research projects funded by Malaysian government and also Japan International Cooperation Agency (JICA).