An Encoding and Labeling Scheme Based on Continued Fraction for Dynamic XML

Yi Jiang
Department of Computer Science, Xiamen University, Xiamen, 361005, China
Email: jiangyi@xmu.edu.cn

Xiangjian He
Research Centre for Innovation in IT Services and Applications (iNEXT)
University of Technology, Sydney, Australia
Email: Xiangjian.He@uts.edu.au

Fan Lin
School of Software, Xiamen University, Xiamen, 361005, China
Email: iamafan@xmu.edu.cn

Wenjing Jia
Research Centre for Innovation in IT Services and Applications (iNEXT)
University of Technology, Sydney, Australia
Email: Wenjing.Jia-1@uts.edu.au

***Abstract*— Much research about labeling schemes has been conducted to efficiently determine the ancestor-descendant relationships and the document-order between any two random XML nodes without re-labeling for updates. In this paper, we present an efficient XML encoding and labeling scheme for dynamic XML document, named *Continued Fraction-based Encoding (CFE)*. The proposed CFE scheme labels nodes with continued fractions and has the following three important properties: (1) CFE codes can be inserted between any two consecutive CFE codes with the orders kept and without re-encoding the existing nodes; (2) CFE is orthogonal to specific labeling schemes, thus it can be applied broadly to different labeling schemes or other applications to efficiently process the updates; (3) CFE supports all structural relationships query in XPath. Two test data sets were built for evaluation. The experimental results show that CFE provides fairly reasonable XML query processing performance while completely avoiding re-labeling for updates.**

***Index Terms*— continued fraction; labeling scheme; dynamic XML data**

I. Introduction

XML-based query and processing have drawn more and more attentions in both academia and industry for many years. The main XML query languages are XPath [1]–[4] and XQuery [5]–[7]. The common core technologies of these query languages include the use of regular path expressions to query XML data and the retrieval of the user specified structure model to realize the structural query of XML. In order to efficiently support the structural query of XML, many researchers have proposed a variety of static encoding schemes, such as the range encoding scheme [8]–[15], the prefix encoding scheme [16]–[21] and the prime number encoding scheme [22]. This is to quickly determine whether the relationship between any two XML elements is parent-child or ancestors-descendant relation, and to determine the document-order. If the XML document is static, these encoding schemes are able to effectively deal with the structural query of the XML. When an XML document is to be updated, one more important point for the labeling scheme is the label update cost in inserting or deleting a node into or from the XML tree. All the current labeling schemes have high update costs. Therefore, in order to effectively deal with XML document updating, many other researchers have proposed a variety of dynamic encoding schemes [23]–[37]. Amagasaetal [23] useed floating-point numbers rather than integral numbers to label the XML element. However, the storage of floating-point numbers on a computer is limited by accuracy [23], and it is still unable to avoid the re-labeling of XML elements. OrdPath [24] is a prefix encoding scheme, which uses only odd numbers during the initial labeling of the XML in order to avoid re-labeling documents. In the event of inserting an element, it uses an even number between the two odd numbers to connect the new element to the existing elements. Although OrdPath can effectively deal with XML document updating, the length of the code is O(n) when frequent insertions happen. QED [25] encoding completely avoids re-labeling and is orthogonal to some of the encoding schemes. However, when an

This paper is based on “A Continued Fraction Encoding and Labeling Scheme for Dynamic XML Data,” by Y. Jiang et al., which appeared in the Proceedings of the 2011 International conference on Industry, Information System and Material Engineering (IISME 2011), Guangzhou, China, April 2011. © 2011 IEEE.

This work was supported by National Natural Science Foundation of China (Grants No. 61001143 and No. 50604012).

doi:10.4304/jsw.6.10.2043-2049

XML node is inserted with new nodes continuously, the coding length of the node grows rapidly.

To tackle the above mentioned issues, in this paper we propose a new dynamic encoding scheme, named *Continued Fraction-based Encoding (CFE)*, and computer simulations are run to demonstrate its performance. Experimental results show that the proposed CFE scheme has a good query and update performance, and high feasibility. At the same time, it has the following advantages:

1) It supports all structural queries of the Xpath to determine the ancestors-descendant and parent-child relationships, and the document order between any two XML nodes;
2) It can insert a new CFE node between any two consecutive CFE nodes. When updating an XML document, CFE can completely avoid re-labeling existing nodes; and
3) CFE is orthogonal to the range encoding scheme and the prefix encoding scheme. Hence, they can be combined to further improve the updating performance.

II. Dynamic Encoding based on Continued Fraction

With continued fraction, CFE completely avoids the re-labeling issue when inserting XML elements. Table I shows the encoding results from numbers 1 to 18 with various encoding schemes. For encoding details of QED, please refer to [25] and [30]. The description of CFE coding is as follows.

Definition: Known i and n are positive integers, and $1 \leq i \leq n$, the coding for number i is

$$\mathrm{cfcode(i)} = [0, n + 2 - i].$$

It can be seen from Table I,

$$\mathrm{cfcode}[1] < \mathrm{cfcode}[2] < \cdots < \mathrm{cfcode}[18]$$

which is exactly the same as decimal number order.

A. The Application of CFE

CFE is orthogonal to the prefix encoding and range encoding. It is called as *Range-CFE* when CFE is applied to the range encoding [10]. In the Range-CFE, each node is labeled as $(\mathrm{startCF}, \mathrm{endCF}, \mathrm{level})$, in which $\mathrm{startCF}$ and endCF are simple continued fractions. Figure 1(b) is an example of Range-CFE.

Similarly, CFE can also be applied to the prefix encoding, keeping the document order. It is called *Prefix-CFE* when CFE is applied to the prefix encoding. Figure 2 is an example of Prefix-CFE. As there is only a single root node, there is no need to label it. Root node has four children nodes which are labeled as [0, 5], [0, 4], $[0, , 3]$ and [0, 2] from left to right. Similarly, Nodes [0, 4] and [0, 2] have two children each which are labeled as [0, 4], [0, 3], [0, 4], [0, 2] and [0, 2], [0, 3], [0, 2], [0, 2] respectively.

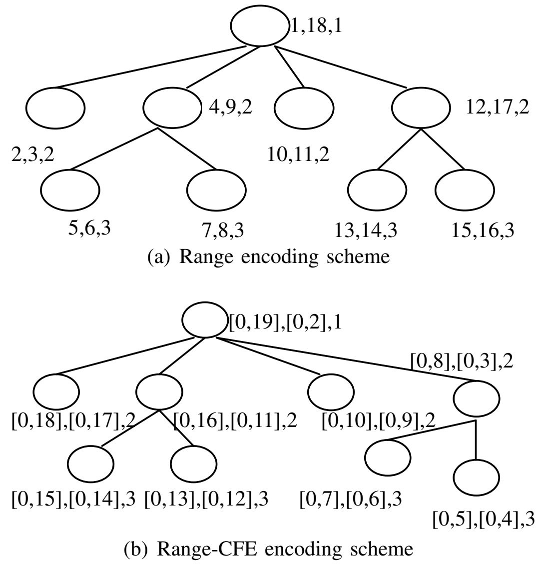

Figure 1. Example of applying CFE to Range Encoding scheme.

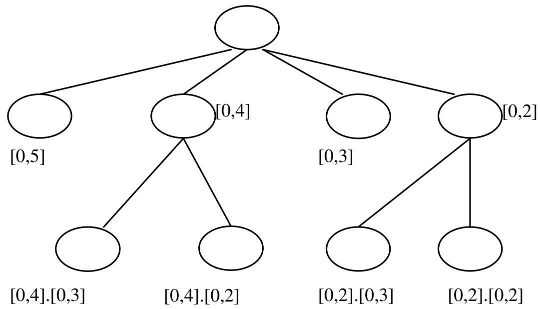

Figure 2. Example of Prefix-CFE encoding scheme.

B. Determining Ancestor-Descendant Relationship and Document Order

Theorem 1: In Range-CFE, the node A is the ancestor of node D, if and only if the label of node A $(\mathrm{startCFA}, \mathrm{endCFA}, \mathrm{levelA})$ and the label of node D $(\mathrm{startCFD}, \mathrm{endCFD}, \mathrm{levelD})$ satisfy the condition:

$$\mathrm{startCFA} < \mathrm{startCFD} < \mathrm{endCFD} < \mathrm{endCFA}.$$

Theorem 2: In Range-CFE, Node A is the father of Node D, if and only if A is the ancestor of D, and at the same time, the label of Node A $(\mathrm{startCFA}, \mathrm{endCFA}, \mathrm{levelA})$ and the label of Node D $(\mathrm{startCFD}, \mathrm{endCFD}, \mathrm{levelD})$ satisfy the condition:

$$\mathrm{levelA} - \mathrm{levelD} = 1.$$

C. Storage of the Encoding

CFE encoding uses Utf-8 format. In the case of Range-CFE, for example, the storage format will be $192, 81$ when a node has been labelled as $[0, 19], [0, 2, 8], 1$. It follows the following three rules:

1) each different item is separated by a space;
2) the starting 0 is not to be stored with the same continued fraction; and

TABLE I.
COMPARISON OF DIFFERENT ENCODING SCHEMES.

Decimal Number	QED	CFE	Decimal Number	QED	CFE
1	112	[0,19]	10	1	[0,10]
2	12	[0,18]	11	10001	[0,9]
3	122	[0,17]	12	1001	[0,8]
4	13	[0,16]	13	101	[0,7]
5	132	[0,15]	14	1011	[0,6]
6	2	[0,14]	15	11	[0,5]
7	212	[0,13]	16	1101	[0,4]
8	22	[0,12]	17	111	[0,3]
9	222	[0,11]	18	1111	[0,2]

3) each different item is separated by a comma with the same continued fraction.

D. Processing XML Documents Update with CFE

It is a very important issue how to minimize the cost of update in processing dynamic XML documents when there are frequent insertings and deletings of nodes. One important feature of CFE code is that it does not require re-labeling of existing nodes when inserting a new CFE node between any two CFE nodes. The following reasons show how to insert a new CFE code between two arbitrary CFE codes.

Deduction: For two given continued fractions A and B with

$$0 < A < B < 1,$$

$$A = [0, \alpha_2, \ldots, \alpha_j, x_{j+1}, \ldots, x_s],$$

$$B = [0, \alpha_2, \ldots, \alpha_j, y_{j+1}, \ldots, y_t], \text{and}$$

$$x_{j+1} \neq 1, y_{j+1} \neq 1, \text{and } x_{j+1} \neq y_{j+1},$$

when any of the following conditions are met, the continued fraction C obtained will meet $A < C < B$:

1) $s = j$ and $t \geq j + 1$, then

$$C = [0, \alpha_2, \ldots, \alpha_j, y_{j+1} + 100];$$

2) $s \geq j + 1$ and $t = j$, then

$$C = [0, \alpha_2, \ldots, \alpha_j, x_{j+1} + 100];$$

3) $s \geq j + 1$, $t \leq j + 1$, and $|x_{j+1} - y_{j+1}| > 1$, then

$$C = [0, \alpha_2, \ldots, \alpha_j, (x_{j+1} + y_{j+1})/2];$$

4) $s = j + 1$, $t = j + 1$, $|x_{j+1} + y_{j+1}| = 1$, and j is an even number, then

$$C = [0, \alpha_2, \ldots, \alpha_j, x_{j+1}, 100];$$

5) $s \geq j + 2$, $t = j + 1$, $|x_{j+1} - y_{j+1}| = 1$, and j is an even number. If $x_{j+2} = 2$, then

$$C = [0, \alpha_2, \ldots, \alpha_j, x_{j+1}, 1, 100];$$

otherwise,

$$C = [0, \alpha_2, \ldots, \alpha_j, x_{j+1}, x_{j+2} - 1];$$

6) $s \geq j + 1$, $t \geq j + 2$, $|x_{j+1} - y_{j+1}| = 1$, and j is an even number, then

$$C = [0, \alpha_2, \ldots, \alpha_j, y_{j+1}, y_{j+2} + 100];$$

7) $s = j + 1$, $t = j + 1$, $|x_{j+1} - y_{j+1}| = 1$, and j is an odd number, then

$$C = [0, \alpha_2, \ldots, \alpha_j, y_{j+1}, 100];$$

8) $s = j + 1$, $t \geq j + 2$, $|x_{j+1} - y_{j+1}| = 1$, and j is an odd number. If $y_{j+2} = 2$, then

$$C = [0, \alpha_2, \ldots, \alpha_j, y_{j+1}, 1, 100];$$

otherwise

$$C = [0, \alpha_2, \ldots, \alpha_j, y_{j+1}, y_{j+2} - 1];$$

9) $s \geq j + 2$, $t \geq j + 1$, $|x_{j+1} - y_{j+1}| = 1$, and j is an odd number, then

$$C = [0, \alpha_2, \ldots, \alpha_j, x_{j+1}, x_{j+2} + 100].$$

Example-1. For two given continued fractions $A = [0, 1]$ and $B = [0, 1, 3]$, while $s = j = 2$, $t = 3 \geq j + 1$, in line with Case 1, $C = [0, 1, 103]$, $A < C < B$.

Example-2. For two given continued fractions $A = [0, 1, 3, 5]$ and $B = [0, 1, 3]$, while $j = 3$, $s = 4 \geq j + 1$, $t = 3 = j$, in line with Case 2, $C = [0, 1, 3, 105]$.

Example-3. For two given continued fractions $A = [0, 1, 3]$ and $B = [0, 1, 9]$, while $j = 2$, $s = 3 \geq j + 1$, $t = 3 \geq j + 1$, $|x3 - y3| > 1$, in line with Case 3, $C = [0, 1, 6]$.

E. Processing XML Documents Update with Range-CFE

Updating on an XML document includes inserting or deleting XML elements, text, attributes, or changing text. Encoding of a new node takes place only when an insertion is involved, while deleting elements and changing text have nothing to do with encoding. The insertion of XML nodes can be divided into the following four types:

1) the inserted node has no brother nodes;
2) the inserted node has only left brother node;
3) the inserted node has only right brother node; and
4) the inserted node has a brother node on both sides.

Following are discussions for these four situations.

1) Node A has a label of (startCFA, endCFA, levelA) without child nodes. Now insert Node D to Node A. D's label is calculated as follows (GetInsertedCode

is the algorithm for obtaining insertion label in deduction):

startCFD = GetInsertedCode (startCFA, endCFA),
endCFD = GetInsertedCode (startCFD, endCFA).

2) Assume the most right node to Node A is B. Insert Node D right to Node B. Then,

startCFD = GetInsertedCode (endCFB, endCFA),
endCFD = GetInsertedCode (startCFD, endCFA).

3) Assume the most left node to Node A is B. Insert Node D left to Node B. Then,

startCFD = GetInsertedCode (startCFA, startCFB),
endCFD = GetInsertedCode (startCFD, startCFB).

4) Assume Nodes A and B are next to each other, Node A is on the left side of Node B. Assuming a new node denoted by D is inserted between Nodes A and B, then,

startCFD = GetInsertedCode (endCFA, startCFB),
endCFD = GetInsertedCode (startCFD, startCFB).

III. Experimental Results

In our experiments, the three dynamic encoding schemes, OrdPath [24], QED [25] and Range-CFE, have been discussed and compared with regard to their storage space, query time and update performance, etc. All algorithms are implemented with C# programming language. All experiments are run on a 2.02GHz Celeron processor with 512MB of physical memory running Windows XP Professional. XML testing data set comes from [38], of which characteristics are shown in Table II.

A. Comparison of Storage Space

Figure 3 shows the total coding size of all nodes in each data set. QED takes up the largest space for each data set. Compared with OrdPath, Range-CFE takes up lager space for data sets D1 and D3, and takes up less space for data sets D2 and D4.

Shakespeare's plays data set (D4) is used to test the query performance of the three encoding schemes. The test query sentence and the number of returned nodes are shown in Table III. Figure 4 indicates that Range-CFE has the best query performance and QED is the second with regard to query time.

B. Update Property

Assuming that the elements are in order in the Shakespeare's plays (D4), Club (D1), Movie (D2), and edmunds (D3). This study has tested the update performance of different encoding schemes under the follwoing four situations:

1) inserting one new element between any two adjacent elements in the Club;
2) inserting one new element between any two adjacent elements in the Movie;
3) inserting one new element between any two adjacent elements in edmunds; and
4) inserting one new element between any two adjacent elements in Shakespeare's plays.

It can be seen from Figure 5 that the Range-CFE has the best update performance and QED is the second.

IV. Conclusions

By applying the continued fraction based XML encoding CFE to range encoding, i.e., the Range-CFE encoding, it provides a good solution to the problems of existing encoding schemes, such as high cost of re-labeling XML elements when updating XML documents, as well as the rapid growth of code length when continuously inserting new nodes at an XML node. This is because the Range-CFE not only supports the two XML nodes having ancestors-descendent or parent-child relationships, and can quickly determine document order, but also does not require re-label existing nodes when inserting or deleting any XML node. The experimental results have demonstrated that the Range-CFE encoding scheme has superior performance to existing encoding schemes on the label updates and query processing performance. Follow-up work will explore how to establish an index based on CFE to better fit the need of XPath query.

Acknowledgment

We thank sincerely for all resources offered to us, including various kinds of data, dissertations, reports and other materials. We also thank the reviewers for their detailed and valuable comments.

References

[1] A. Berglund, S. Boag, and D. Chamberlin, "Xml path language (xpath) 2.0," *W3C working draft*, 2002.
[2] S. Abiteboul, D. Quass, J. McHugh, J. Widom, and J. L. Wiener, "The lorel query language for semistructured data," *International Journal on Digital Libraries*, vol. 1, no. 2, pp. 68–88, 200.
[3] A. Deutsch, M. Fernadez, D. Florescu, A. Levy, and D. Suciu, "A query language for xml," in *Proceedings of the 8th International World Wide Web Conference*, Toronto,Canada, 1999, pp. 77–91.
[4] D. S. J. Robie, J. Lapp, "Xml query language (xql)," in *Proceedings of the W3C Query Languages Workshop (QL'98)*, Boston, Massachusets, USA, 1998, pp. 3–4.
[5] P. Fankhauser, M. Fernandez, and A. Malhotra, "Xquery 1.0 formal semantics," *W3C Working Draft*, 2002.
[6] D. Chamberlin, J. Robie, and D. Florescu, "Quilt: An xml query language for heterogeneous data sources," in *Proceeings of Web DB 2000*, Dallas,texas, 2000, pp. 53–62.
[7] D. Chamberlin, "Xquery: A query language for xml," in *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data*, San Diego, California, 2003, pp. 682–682.
[8] P. F. Dietz, "Maintaining order in a linked list," in *Proceedings of the 14th Annual ACM Symposium on Theory of Computing*. San Francisco: ACM Press, 1982, pp. 122–127.

TABLE II.
TESTING DATA SET.

Data set	Subject	Number of documents	Max fan out of element	Max depth of element	Total number of nodes in data set
D1	Club	12	47	3	2928
D2	Movie	490	38	4	26044
D3	edmunds	1190	162	3	234400
D4	Shakespeare's plays	37	434	5	179689

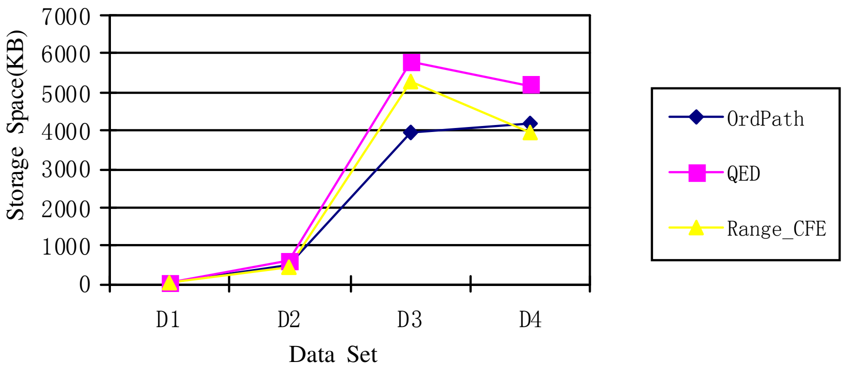

Figure 3. Comparison on storage space.

TABLE III.
TESTING DATA SET.

Query		Number of Returned Nodes
Q1	/play/act[4]	37
Q2	/play/act[5]//preceding::scene	611
Q3	/play/act/scene/speech[2]	730
Q4	/play/*/*	1938
Q5	/play/act//speech[3]/preceding-sibling::*	3093
Q6	/play//act[2]/following::speaker	18406
Q7	/play//scene/speech[6]/following-sibling::speech	26705
Q8	/play/act/scene/speech	30933
Q9	/play/*//line	107833

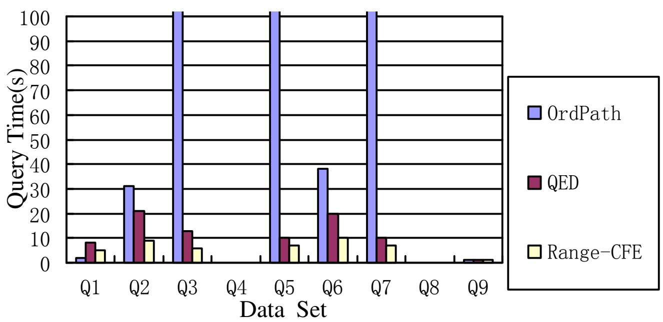

Figure 4. Comparison on query time.

[9] Q. Li and B. Moon, "Indexing and querying xml data for regular path expressions," in *Proceedings of the 27th International Conference on VLDB*. Roma: Morgan Kaufmann Publishers, 2001, pp. 361–370.

[10] C. Zhang, J. Naughton, and D. DeWitt, "On supporting containment queries in relational database management systems," in *Proceedings of the 2001 ACM SIGMOD International COMAD*. Santa Barbara:ACM Press, 2001, pp. 425–436.

[11] C. X. Wan and Y. S. Liu, "Restore: Middleware for xml's relational storage and retrieve," *Wuhan University Journal of Natural Science*, vol. 8, no. 1A, pp. 28–34, 2003.

[12] C. W. Chung, J. K. Min, K. Shim, and K. Seoul, "Apex: An adaptive path index for xml data," in *Proceedings of the 2002 ACM SIGMOD international conference on Management of data*, Madison, Wisconsin, 2002, pp. 121–132.

[13] R. Kaushik, P. Shenoy, P. Bohannon, and E. Gudes, "Exploiting local similarity for indexing paths in graph-structured data," in *Proceedings of the 18th International Conference on Data Engineering*, San Jose, USA, 2002, pp. 129–140.

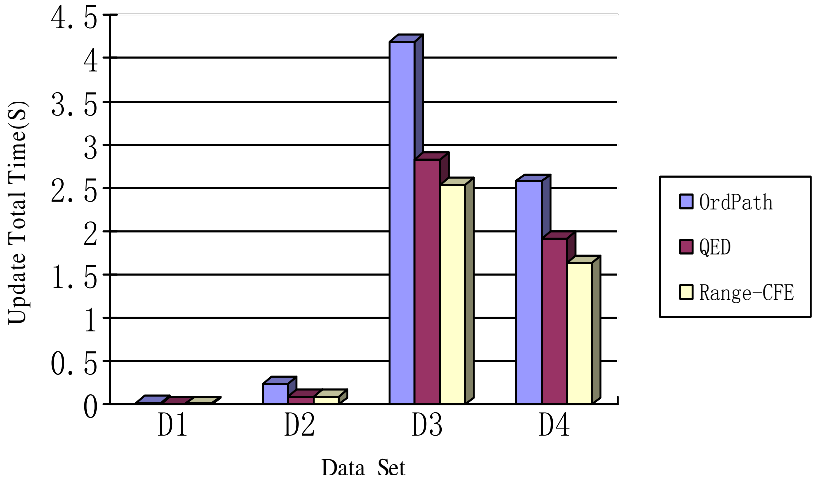

Figure 5. Comparison on renew property.

[14] H. Wang, S. Park, W. Fan, and P. S. Yu, "Vist: a dynamic index method for querying xml data by tree structures," in *Proceedings of the 2003 ACM SIGMOD*, Madison, USA, 2003, pp. 110–121.

[15] P. Rao and B. Moon, "Prix: Indexing and querying xml using prufer sequences," in *Proceedings of the 2004 ICDE*, Boston, USA, 2004, pp. 288–299.

[16] S. Tatarinov, K. S. Viglas, and Beyer, "Storing and querying ordered xml using a relational database system." in *Proceedings of the 2002 ACM SIGMOD int'l COMAD*. Madison: ACM Press, 2002, pp. 204–215.

[17] E. Cohen, H. Kaplan, and T. Milo, "Labeling dynamic xml trees. in spds," in *SPDS*. Madison: ACM Press, 2002, pp. 271–281.

[18] H. Wang and X. Meng, "On the sequencing of tree structures for xml indexing," in *Proceedings of the 2005 ICDE*, Tokyo, Japan, 2005, pp. 372–383.

[19] S. A. l. Khalifa, H. V. Jagadish, and N. Koudas, "Structural joins: A primitive for efficient xml query pattern matching," in *Proceedings of the 2002 ICDE*, San Jose, USA, 2002, pp. 141–152.

[20] H. Jiang, W. Wang, H. Lu, and J. X. Yu, "Holistic twig joins on indexed xml documents," in *Proceedings of the 2003 VLDB*, Berlin, Germany, 2003, pp. 273–284.

[21] Q. Li and B. Moon, "Indexing and querying xml data for regular path expressions," in *Proceedings of the 2001 VLDB*, Rome, Italy, 2001, pp. 361–370.

[22] X. Wu, M. L. Lee, and W. Hsu, "A prime number labeling scheme for dynamic ordered xml trees," in *Proceedings of the 19th International Conference on Data Engineering*. Boston: IEEE Computer Society, 2004, pp. 66–78.

[23] T. Amagasa, M. Yoshikawa, and S. Uemura, "Qrs: A robust numbering scheme for xml documents," in *Proceedings of the 19th International Conference on Data Engineering*. Bangalore: IEEE Computer Society, 2003, pp. 705–707.

[24] P. O'Neil, E. O'Neil, and S. Pal, "Ordpaths: Insert-friendly xml node labels," in *Proceedings of the 2004 ACM SIGMOD International COMAD*. Paris: ACM Press, 2004, pp. 903–908.

[25] C. Li and T. W. Ling, "Qed: a novel quaternary encoding to completely avoid re-labeling in xml updates," in *Proceedings of the 14th ACM International CIKM*. Bremen: ACM Press, 2005, pp. 501–508.

[26] H. Xiao, C. Tang, and T. Zhang, "Btcs: The binary traveling coding scheme for xml document," *Journal of Sichuan University of Natural Science*, vol. 43, no. 3, pp. 532–536, 2006.

[27] L. Xu, Z. Bao, and T. Wang, "A dynamic labeling scheme using vectors," in *Proceedings of 18th International Conference on DEXA*. Regensburg: Springer-Verlag, 2007, pp. 130–140.

[28] R. Thonangi, "A concise labeling scheme for xml data," in *Proceedings of International COMAD*. Delhi, India: Computer Society of India, 2006.

[29] J.-K. Min, J. Lee, and C.-W. Chung, "An efficient encoding and labeling for dynamic xml data," in *Proceedings of 12th International Conference on DASFAA*. Bangkok, Thailand: Springer-Verlag, 2007, pp. 715–726.

[30] C. Li, T. W. Ling, and M. Hu, "Efficient processing of updates in dynamic xml data," in *Proceedings of the 22nd International Conference on Data Engineering*. Atlanta, Georgia: IEEE Computer Society, 2006, pp. 13–23.

[31] Y. Lu, L. Zhang, and W. Wang, "A new xml document coding scheme," *Journal of Computer Research and Development*, vol. 41, no. 3, pp. 500–503, 2004.

[32] H. Jiang, H. Lu, and W. Wang, "Xr-tree:indexing xml data for efficient structural joins," in *Proceedings of the 2003 ICDE*, San Jose, USA, 2003, pp. 78–82.

[33] N. Bruno, N. Koudas, and D. Srivastava, "Holistic twig joins: optimal xml pattern matching," in *Proceedings of the 2002 ACM SIGMOD*, Madison, USA, 2002, pp. 310–321.

[34] T. Chen, J. Lu, and T. Ling, "On boosting holism in xml twig pattern matching using structural indexing techniques," in *Proceedings of the 2005 ACM SIGMOD*, Baltimore, Maryland, 2005, pp. 455–466.

[35] J. Lu, T. W. Ling, C. Y. Chan, and T. Chen, "From region encoding to extended dewey: On efficient processing of xml twig pattern matching," in *Proceedings of the 2005 VLDB*, Trondheim, Norway, 2005, pp. 193–204.

[36] T. Chen, T. W. Ling, and C. Y. Chan, "Prefix path streaming: A new clustering method for optimal holistic xml twig pattern matching," in *Proceedings of the 2004 DEXA*, Zaragoza, Spain, 2004, pp. 801–810.

[37] J. Lu, T. Chen, and T. W. Ling, "Efficient processing of xml twig patterns with parent child edges: a look-ahead approach," in *Proceedings of the 2004 CIKM*, Washington, USA, 2004, pp. 533–542.

[38] "The niagara project experimental data," http://www.cs.wisc.edu/niagara/data.html.

Yi Jiang is currently an Associate Professor at the Department of Computer Science, School of Information Science and Technology, Xiamen University, China. He was a lecturer at the Department of Finance in the School of Economics, Xiamen University, China from 1994 to 1998. His research interest includes database, data mining, knowledge discovery and embedded system.

Xiangjian He received his PhD degree in Computing Sciences in 1999 from University of Technology, Sydney (UTS). He is currently a professor at UTS and a Deputy Director of Research Centre for Innovation in IT Services and Applications (iNEXT) at UTS. His research interest includes computer vision, image processing, e-services, and computer and network security. In recent years, he has had over 200 refereed publications in books, journals and conferences. Professor He is a Senior Member of IEEE.

Fan Lin received his MS degree in Computer Science from the Department of Computer Science, Xiamen University, China in 2003. He is currently an Assistant Professor at School of Software, Xiamen University. Dr. Lin's research interest includes security and cloud computing, evolutionary computation, embedded system, Internet middleware, and machine vision.

Wenjing Jia received her PhD degree in Computing Sciences in 2007 from University of Technology, Sydney (UTS). She is currently a Lecturer in Faculty of Engineering and Information Technology at UTS, and a core member of Research Centre for Innovation in IT Services and Applications (iNEXT) at UTS. Before joining UTS, Dr. Jia was an Associate Lecturer in Fuzhou University from 1999 to 2003. Her research interest includes computer vision, pattern recognition, and machine learning. Dr. Jia is a member of IEEE and IPRS since 2006.