IJREAT International Journal of Research in Engineering & Advanced Technology, Volume 2, Issue 2, Apr-May, 2014
ISSN: 2320 – 8791 (Impact Factor: 1.479)
www.ijreat.org

A Conventional Query Processing using Wireless XML Broadcasting

Subhashini.G[1], Kavitha.M[2]

[1]M.E II Year, Department of Computer Science and Engineering, Sriram Engineering College, Perumalpattu – 602 024

[2]Assistant Professor, Department of Computer Science and Engineering, Sriram Engineering College, Perumalpattu – 602 024

Abstract

Wireless broadcasting is an effective information dissemination approach in the wireless mobile environment. This project provides an energy and latency efficient wireless XML Stream Broadcasting scheme supporting twig pattern queries. A XML Automation tool generates customized XML files without relying on third persons for XML files. Parser parses the XML document to generate XML Stream. A traditional and a novel unit structure called G-node is defined for streaming XML data in the wireless environment. It exploits the benefits of the structure indexing and attributes summarization that can integrate relevant XML elements into a group. It provides a way for selective access of their attribute values and text content. A lightweight and effective encoding scheme, called Family Tree Encoding is defined, to support evaluation of predicates and twig pattern queries over the stream. The Query processing technique process the Queries issued by the mobile client and it selectively tunes the data of interest from the Broadcast Stream. The goals of conventional query processing on streamed XML data are to minimize computation costs and filtering time. This scheme reduces the tuning time as it provides an effective way for selective access of XML elements.

Keywords *-Wireless Broadcasting, XML Stream,XML Automation, Structure Indexing, Attribute Summarization.*

I. INTRODUCTION

With the rapid development of wireless network technologies, wireless mobile computing has become popular. Users communicate in the wireless mobile environment using their mobile devices such as smart phones and laptops while they are moving. Wireless broadcasting is an effective information dissemination approach in the wireless mobile environment because of the following benefits: 1) the server can support a massive number of mobile clients without additional costs (i.e., scalability), 2) the broadcast channel is shared by many clients (i.e., the effective utilization of bandwidth), and 3) the mobile clients can receive data without sending request messages that consumes much energy We need to consider energy conservation of mobile clients when disseminating data in the wireless mobile environment, because they use mobile devices with limited battery-power (i.e., energy-efficiency). The overall query processing time must also be minimized to provide fast response to the users (i.e., latency-efficiency).

A XML Automation tool is used for customized XML creation enables the server to Broadcast the customized data's as and when needed without relying on the third party for XML files. The Implementation support to dynamic customized XML is a major advantage of the wireless streaming in mobile environment. The XPath is used as a query language. The results of an XPath query are selected by a location path. A location path consists of location steps. Processing each location step selects a set of nodes in the document tree that satisfy axis, node test and predicates described.

A GNode the novel attribute of our system can be added dynamically to the broadcast channel without interrupting the streaming of XML data. This feature enables to dynamically add events in the existing channel. Dynamic addition of G-Node ensures the credibility of the Broadcast system efficiently proposed by our approach. Dynamic modification of Attribute value enables to change any data on the broadcast

stream whenever needed and is achieved by the Attribute summarization mechanisms and the Structured Indexing of XML data handled in our system.

To measure the energy-efficiency and latency-efficiency in wireless broad-casting, the tuning time and access time are used, respectively. The tuning time is the sum of the elapsed times spent by a mobile client to download the required data.

Fig.1 explains the Wireless XML Broadcasting Architecture. In this Wireless XML Broadcasting Architecture, the broadcast server retrieves XML data to be disseminated from the XML repository which is generated by a tool called XML Automation Tool. Then, it parses and generates a wireless XML stream. The XML stream is continuously disseminated via a broadcast channel. In the client-side, if a query is issued by the mobile client, the mobile client tunes in to the broadcast channel and selectively downloads the XML stream for query processing.

For providing energy-efficient query processing over XML data in wireless and mobile environments, several approaches exploiting the benefits of wireless broadcasting have been proposed to reduce structural overheads of the original XML document and attach indices containing time information to the XML data stream. We refer to these approaches as wireless XML streaming compared to conventional streaming and processing of XML data in the wired environment. These works enable mobile clients to selectively download the data of their interests by using indices. However, they cannot process XML twig pattern queries efficiently since they do not contain branching information or parent-child relationships.

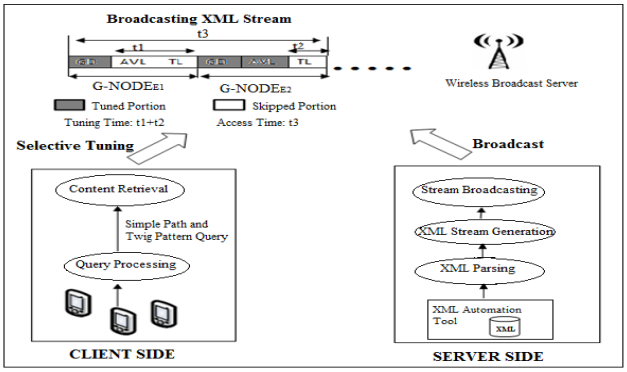

Fig. 1 Wireless XML Broadcasting

In this paper, we propose a novel, energy and latency-efficient wireless XML streaming scheme supporting twig pattern queries in the wireless mobile environment. The overview and main contributions of our work are summarized as follows:

- We define a streaming unit of a wireless XML stream, called G-node. The G-node structure eliminates structural overheads of XML documents, and enables mobile clients to skip downloading of irrelevant data during query processing.
- We propose a new scheme call XML Automation. The XML automation helps in customized creation of XML files ie, it helps in XML file generation without relying on the third party to create XML files.
- We propose a light-weight encoding scheme, called Family Tree (FT) Encoding, to represent parent-child relation-ships among XML elements in the G-nodes.
- We provide algorithms for generating wireless XML stream consisting of G-nodes and query processing over the wireless XML stream.

II. BACKGROUND

A. *XML Data Model and XPath Expression*

```
<mondial>
        <country id="f0_162" name="Belgium" capital="f0_1477" population="10170241">
                <name>Belgium</name>
                <province id="f0_17457" name="Antwerp">1610695</province>
                <province id="f0_17462" name="Hainaut">
                        <located at>West of Belgiym</located at>
                        <city id="f0_2335" country="f0_162" province="f0_17462">Charleroi</city>
                                <population>255634</population>
                        <city id="f0_2345" country="f0_162" province="f0_17462">Mons</city>
                                <population>188639</population>
                        <city id="f0_2345" country="f0_162" province="f0_17462">Liege</city>
                </province>
        </country>
        <country id="f0_174" name="Bulgaria" capital="f0_1487" population="8612757">BU</country>
        <country id="f0_208" name="Finland" capital="f0_1507" population="5105230">
                <province id="f0_33615" name="Aland">
                        <city id="f0_35399" country="f0_208" province="f0_33615">Mariehamn</city>
                                <population>315630</population>
                </province>
                <province id="f0_33620" name="Haeme">662000</province>
        </country>
        <country id="f0_184" name="CzechRepublic" capital="f0_1493" population="10321120">
                <province id="f0_17473" name="Jihomoravsky">
                        <located at>East Cost of Czech</located at>
                        <city id="f0_2394" country="f0_184" province="f0_17473">Zlin</city>
                </province>
                <province id="f0_17475" name="Severocesky"></province>
        </country>
</mondial>
```

Fig.2 Sample XML File

An XML document can be represented as a rooted, ordered, and labeled tree. Elements, attributes, and texts are represented by nodes, and the parent-child relationships are represented by edges in the XML tree.

Fig. 2 shows a simple XML document that will be used as a running example in the paper.

In this paper, we use XPath as a query language. The results of an XPath query are selected by a location path. A location path consists of location steps. Processing each location step selects a set of nodes in the document tree that satisfy axis, node test and predicates. For example, a query that finds cities located in Belgium can be represented by the following XPath expression:

Q1://country[@name=‘‘Belgium’’]/province/city

The above query consists of four location steps:// country[@name="Belgium"], /province, and /city. The first location step is to find a "country" node among the descendant elements of the document root, where its "name" attribute equals "Belgium." The second location step is to find "province" child nodes of the "country" nodes. The final location step is to find the "city" nodes.

B. Twig Pattern Query

A twig pattern query consists of two or more path expressions, thus, it involves element selections satisfying complex patterns in tree-structured XML data. The twig pattern query is a core operation in XML query processing and popularly used as it can represent complex search conditions. Fig. 3 shows an example twig pattern query Q2 with their tree representations. Q2 is to find cities located in the provinces of a country that has a child element "name" whose text content is "Belgium".

D. Structure Indexing

Many techniques using a structure index have been proposed for efficient XML query processing. The structure indexing directly captures the structural information of XML documents and is used for XML query processing. Conventional wireless XML streaming methods using a structure index exhibit good performance for simple path query processing benefitting from the size reduction.

Example 1. Fig. 3 shows results of processing the query Q2 1) over its structure index.

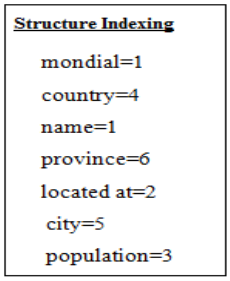

Fig. 3. Query processing results over a structure index.

E. Dynamic XML Data creation

A XML Automation tool is used for customized XML creation enables the server to Broadcast the customized data's as and when needed without relying on the third party for XML files. The Implementation support to dynamic customized XML is a major advantage of the wireless streaming in mobile environment.

The XPath is used as a query language. The results of an XPath query are selected by a location path. A location path consists of location steps. Processing each location step selects a set of nodes in the document tree that satisfy axis, node test and predicates described.

III WIRELESS XML STREAM GENERATION

In this section, we propose a new stream organization for XML data.

A. G-Node

We generate a wireless XML stream by integrating information of elements of the same path. That is, the XML data stream consists of the sequence of integrated (group) nodes, called G-node.

Definition 1 (G-Node). The G-node denoted by G_p =(GD_p,AV L_p, $TL_{p)}$ is a data structure containing information of all the elements e_p whose location path is p, where GD_p is a group descriptor of G_p, AV L_p is a list containing all attribute values of e_p, and TL_p is a list containing all text contents of e_p.

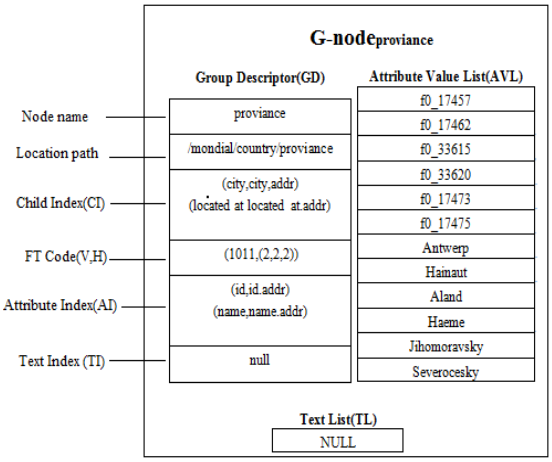

Fig. 4 Structure of G-node

Fig. 4 illustrates the structure of a G-node that integrates elements of the path "/mondial/country/province" in the example XML document in Fig. 2.

The group descriptor is a collection of indices for selective access of a wireless XML stream. Child Index (CI) is a set of addresses that point to the starting positions of child G-nodes in the wireless XML stream. Attribute Index (AI) contains the pairs of attribute name and address to the starting position of the values of the attribute that are stored contiguously in Attribute Value List. Text Index (TI) is an address pointing to the starting position of Text List. In our scheme, an address means a point in time when the relevant data is broadcast on the air. The components of the group descriptor are used to process XML queries in the mobile client efficiently. Indices relating to time information such as CI, AI, and TI are used to selectively download the next G-nodes, attribute values, and text. Attribute Value List (AVL) and Text List (TL) store attribute values and text contents of the elements represented by the G-node, respectively. Attribute values and text contents are stored in document order of elements.

B. Attribute Summarization

In our scheme, we exploit the benefits of the Attribute Summarization technique to reduce the size of a wireless XML stream. In XML, an element may have multiple attributes, each of which consists of a name and value pair. In addition, there is a structural characteristic that elements with the same tag name and location path often contain the attributes of the same name.

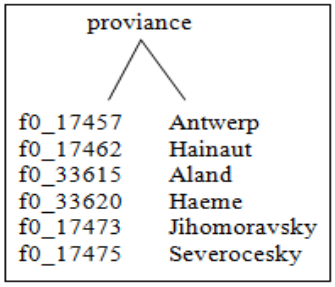

Fig.5 Attribute Summarization

Fig. 5 shows the concept of attribute summarization. Six "province" nodes in the XML document presented in Fig. 2 commonly contain two types of attributes whose names are "id" and "name." In our scheme, redundant attribute names (i.e., five "id" and "name" strings) are eliminated, thus, the size of XML stream can be significantly reduced.

C. Wireless XML Stream Generation

In this section, we explain how to generate wireless XML stream. A server retrieves an XML document to be disseminated from the XML repository and it generates wireless XML steam by using SAX (Simple API for XML), which is an event-driven API. SAX invokes content handlers during the parsing of an XML document. After streaming of XML data, streamed XML data are disseminated via a broadcast channel.

IV FAMILY TREE ENCOADING

A. FT code

We propose a novel encoding scheme, called Family Tree Encoding, to support queries involving predicates and twig pattern matching. In the proposed scheme, two kinds of FT codes, i.e., vertical code denoted by FT code (V) and horizontal code denoted by FT code(H), are used to represent parent-child relationships among XML elements in two G-nodes.

Parent - Child	FT- code(H)	FT-code(v)
country-province	1011	222
province-located at	010010	11
country-name	1000	1
province-city	011010	311

province-population	011000	21
mondial-country	1	4

Fig.7 FT codes for the given XML file

In the above definition of FT code (V, H), 1-value bits in FT code(V) identify the elements of an ordered subset EP' of EP which have at least one child element in EC. Each positive integer ni in FT code (H) denotes the number of child elements in EC of the ith element in EP '.

Fig. 7 shows an example of FT codes for the example XML document.Note that FT code (V) of G-nodeprovince is defined by 1011 since the elements integrated in G-nodeprovince are mapped to only the first, third, and fourth elements in G-nodecountry. FT code (H) of G-nodeprovince is (2, 2, 2), where each value denotes the number of child elements in G-nodeprovince mapped to the same parent element in G-nodecountry in document order.

B. Selection Functions

In evaluating a given twig pattern query with predicates, we should select a subset of elements of a particular type satisfying the given predicates. Then, for the selected elements, we should find their parent elements or child elements of a particular type. For example, to process the query Q2 "//country[name/text()="Belgium"]/province/ city," in Section 2.2, we should find a subset of "name" elements satisfying the given predicate condition, select their parent "country" elements, and then identify "province" elements which are children of those "country" elements. Fig. 8 shows the Result of selection bit string.

Function 1. SelectChildren(C;SBp)
Input: a G-node C, a selection bit string SBp for the parent G-node of C
Output: a selection bit string SBc for C
Let the FT Code of C = (V;H).
Vm = Shrink&Mask (V; SBp);
SBc = Unpack(Vm;H);
Return SBc;

Method 1.Shrink and Mask(BitString V, BitString W)

Let V = v1,v2….vn and W = w1,w2…..wnShrink&Mask(V, W) first shrinks the bit string V by removing the bits with a value of 0 from V . It also shrinks the bit string W, eliminating the bits in the same positions as those removed in V . Then, it calculates Bitwise AND of the two bit strings shrunken from V and W.
For example,

Shrink&Mask(011010 and 110011)= 111 & 101 = 101:

Method 2.Unpack(BitString V, Ordered-List H)

Let V = v1,v2….vn and H = (n1,n2……nk). Unpack(V, H) returns a bit string Vr which is obtained by concatenating bit strings
Si (1 <=i<= k) in order, where Si is a bit string of the length ni in which all bits are equal to vi.

For example, Unpack (101,(2, 2, 2))= 110011.

Function 2. SelectParents(C;SBc)
Input: a G-node C, a selection bit string SBc for C
Output: the selection bit string SBp for the parent G-node P of C
Let the FT code of C = (V;H).
Vm = Pack(SBc, H);
SBp = Expand&Mask(V;Vm);
Return SBp;

Method 3. Pack(BitString V, OrderedList H)

Let V = $v_1v_2 \ldots v_n$ and H = (n_1; n_2; . . . ; n_k). Pack (V, H) returns a bit string $V_r = r_1r_2 \ldots r_k$, where r_i = Bitwise-OR {v_{p+1}; v_{p+2}; . . . ; v_{p+ni}}, where

For example, Pack (10010; (3, 1, 1))= 110.

Method 4.Expand&Mask(BitString V, BitString W)

Let V = v1,v2….vn and W = w1,w2…..wn, where m<=n. Expand&Mask(V,W) produces a bit string Vr=r1,r2….rn, where Expand&Mask(V;W) first expands the bit string W to the length of V by placing the bits in W to the same positions of the bits with a value of 1 in V in order and inserting the bits with a value of 0 in the remaining positions. Then it masks the bit string V by the expanded bit string.

For example,

Expand&Mask (011010, 110)= 011010 & 011000 = 011000.

Finally, we define the function to select elements in a Gnode contained in the query tree of a given twig pattern query, which satisfy all the branching paths and

predicate conditions in the sub-tree rooted at the G-node. Note that the answer element means an element satisfying given predicate conditions.

Function 3. GetSelectionBitStringOf (N)
Input: a G-node N in the query tree T
Output: a selection bit string SBn for N
Let SBn be a bit string of length l, where l equals the number of elements in N and all bits are 1;
IF (N has a predicate on its attributes or text) {
Evaluate the predicate to obtain a selection bit string SBp identifying the answer elements to the predicate in N;
SBn = SBn Bitwise-AND SBp;
}
FOR-EACH (child G-node C of N)
IF (all bits in SBn equal to 0) Return SBn;
SBc =GetSelectionBitStringOf(C); //recursive call
SBp = SelectParents(C; SBc);
SBn=SBn Bitwise-AND SBp;
}
Return SBn;

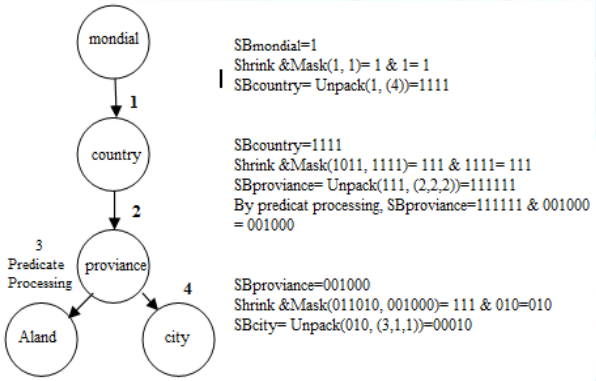

Fig. 8 Result of selection bit string

The above recursive function performs a traversal of the subtree rooted at G-node N in the postorder, depth-first manner. First, it evaluates a selection predicate on the attributes or text, if any, over the elements in the G-node to obtain a selection bit string indicating the answer of the predicate. For each child node C of an internal node N, the algorithm computes its selection bit string SBc recursively and then calculates the selection bit string SBp for N which is derived from SBc and FT code of C using the SelectParents() function. Then, the result selection bit string for N can be produced by performing bitwise AND operations over all the selection bit strings SBp obtained from the child nodes of N.

V QUERY PROCESSING OVER WIRELESS XML STREAM

In this section, we describe how a mobile client can retrieve the data of its interests. Assuming that there is no descendantaxis in the user query, query processing algorithms for a simple path query and a twig pattern query are presented below respectively.

B. Twig Pattern Query Processing

Twig pattern query processing consists of three phases: Tree traversal phase, Subpaths traversal phase, and Main path traversal phase. The main path denotes a path from the root node to a leaf node which represents the target element of the query, while the subpaths denote branch paths excluding the main path in the query tree. In the Tree traversal phase, the mobile client first constructs a query tree. Then, traversing the query tree in a depth-first manner, it selectively downloads group descriptors of the relevant G-nodes into the nodes in the query tree. Attribute values and texts involved in the given predicates are also downloaded into the relevant nodes. In the Subpaths traversal phase, the mobile client performs a post-order depth-first traversal starting from the highest branching node in the query tree. Thus, the selection bit string for the branching node is calculated from all the subpaths in a bottom-up manner. Finally, the Main path traversal phase propagates the selection bit string on the branching node along the main path using the SelectChildren() function. Finally, the mobile client retrieves the set of answer elements in the leaf node of the main path which satisfies the given twig pattern query.

VI CONCLUSION

Twig pattern queries containing complex conditions are popular and critical in XML query processing. In this paper, we proposed an efficient wireless XML streaming method supporting twig pattern queries. The previous work on wireless XML streaming only addressed processing of simple path queries. Thus, they are inefficient for twig pattern queries. In contrast, our scheme provides an energy band latency efficient way to evaluate predicates and twig pattern matching. Specifically, our scheme reduces the size of the XML stream, exploiting the benefits of the structure indexing and attribute summarization. In addition, our scheme reduces the tuning time as it provides an effective way

for selective access of XML elements as well as their attribute values and texts. In this paper, we proposed Lineage Encoding to support queries involving predicates and twig pattern matching. We also defined the relevant operators and functions to efficiently process twig pattern matching. The mobile client can retrieve the required data satisfying the given twig pattern by performing bit-wise operations on the FT codes in the relevant G-nodes. Thus, our scheme can support twig pattern query processing while providing both energy and latency efficiencies.

REFERENCES

[1] S. Acharya, R. Alonso, M. Franklin, and S. Zdonik, "Broadcast Disks: Data Management for Asymmetric Communication Environments," Proc. ACM SIGMOD Int'l Conf. Management of Data Conf., pp. 199-210, Mar. 1995.

[2] S. Al-Khalifa, H.V. Jagadish, N. Koudas, J.M. Patel, D. Srivastava, and Y. Wu, "Structural Joins: A Primitive for Efficient XML Query Pattern Matching," Proc. Int'l Conf. Data Eng. (ICDE), pp. 141-152,Feb. 2002.

[3] M. Altinel and M. Franklin, "Efficient Filtering of XML Documents for Selective Dissemination of Information," Proc. Int'l Conf. Very Large Data Bases (VLDB), pp. 53-64, 2000.

[4] S. Amer-Yahia, S. Cho, L.V.S. Lakshmanan, and D. Srivastava, "Minimization of Tree Pattern Queries," Proc. ACM SIGMOD Int'l Conf. Management of Data Conf., pp. 497-508, 2001.

[5] A. Berglund, S. Boag, D. Chamberlin, M.F. Fernandez, M. Kay, J. Robie, and J. Simeon, "XML Path Language (XPath) 2.0," Technical Report W3C, 2002.

[6] N. Bruno, D. Srivastava, and N. Koudas, "Holistic Twig Joins: Optimal XML Pattern Matching," Proc. ACM SIGMOD Int'l Conf. Management of Data Conf., pp. 310-321, 2002.

[7] S. Chen, H. Li, J. Tatemura, W. Hsiung, D. Agrawal, and K.S. Candan, "Scalable Filtering of Multiple Generalizaed-Tree-Pattern Queries over XML Streams," IEEE Trans. Knowledge and Data Eng., vol. 20, no. 12, pp. 1627-1640, Dec. 2008.

[8] C. Chung, J. Min, and K. Shim, "APEX: An Adaptive Path Index for XML Data," Proc. ACM SIGMOD Int'l Conf. Management of Data Conf., June 2002.