http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 12 (23): 7240-7244, 2013
ISSN 1812-5638 / DOI: 10.3923/itj.2013.7240.7244

Improvement of XML Structural Join Algorithm with Weaving Multi-documents

[1]Jiang Yan,[2]Wang Yu-Xuan,[3]Jin Xin,[2]Li Xin and [4]Pan Ping
[1]Department of Software, Shenyang Universityof Technology, Shenyang, China
[2]Department ofinformation sScience and Technology, Shenyang Universityof Technology, Shenyang, China
[3]Shengyang HeXing Testing EquipmentCo., Ltd., China
[4]Shenyang Conservatory of Music, Shenyang, China

Abstract: At present in the structure of the join under the decision is based on a single document, namely only the node of a document is encoded to achieve structure join. When a document is inserted into the node, or need to be modified, you need to modify part or all of the nodes coding whcih could lead to reduce the query efficiency. Based on the thought of aspect-oriented method of woven into the implementation of data structure join undermulti-documents, so when need to modify the XML document, the code changes to minimum, so as to improve the efficiency of the XML document structure query. This study on the analysis and comparison of the existing structurejoin algorithm, proposed based on woven into the structure of the multi -document join algorithm. Discussed in this article weave set encoding, does not change the original XML document tree node code, only the coding correction, woven into XML document root node and other nodes coding doesn't change.

Key words: Structural join, multi-document, weave, ancestor/descendant, XML

INTRODUCTION

As the core of the XML technology, XML data storage, index and query is an important part of XML database management and XML query is the research focus and hotspot, query language and query method is at the core of all kinds of XML query. At present in the structure of the join under the decision is based on a single document, namely only the node of a document is encoded to achieve structure join. When a document is inserted into the node, or need to be modified, you need to modify part or all of the nodes coding whcih could lead to reduce the query efficiency. And introduced in this study, based on the thought of aspect-oriented method of woven into the implementation of data structure join undermulti-documents, so when need to modify the XML document, the code changes to minimum, so as to improve the efficiency of the XML document structure query (Li *et al.*, 2006; Liu *et al.*, 2008). This study on the analysis and comparison of the existing structure join algorithm, proposed based on woven into the structure of the multi -document join algorithm. (Wei and Wei, 2012; Wu and Yuan, 2009; Wan, 2008).

Encoding nodes is the precondition of query, the original interval coding scheme based on the relative position of the XML document tree node in the document, or some sort of tree traversal sequence to node coding. Range covered by the node that contains the relationship, can determine the ancestor-descendant relationship between nodes. The coding scheme is lack of flexibility, after inserting an XML document node needs to re-encode all nodes. Some coding method with the method of reserve, the enlarged nodes existing interval range in advance, leave room for adding nodes. Discussed in this article weave set encoding, does not change the original XML document tree node code, only the coding correction, woven into XML document root node and other nodes coding doesn't change (Navasa *et al.*, 2009).

DEFINITION OF CODING

Definition 1: Document node is a triple (docID, start, end). docID is the document number, docID ofa set of woven into nodes and a setis woven into nodes is different.start is the value of the preorder traversal of the XML tree, end is the value of second access visit the node after traversing the sub-tree.Let γ'is the set of woven into nodes R' root element, p for the point being woven into the node.

Corresponding Author: JiangYan, Department of Software, Shenyang Universityof Technology, Shenyang, China

Alogrithm 1: Algorithm for getting weaving root code

```
Algorithm:AOEncdoe(p,T,lbrother,rbrother)
{Set the variable curNode to T, set the variable start
and end an initial value of 0;
if (Left brothers is null and right brother is null) {
start=end=start (p) +end (p) ;}
elseif(Left brothers is not null and right brother is
null)
{ The start and end attribute of the element is
start(lbrother)+end(lbrother);}
elseif(Left brothers is null and right brother is not null)
{ The start and end attribute of the element is start(p)+end(rbrother);}
else { The start and end attribute of the element is start (lbrother) +start
(rbrother) ;}
The start attribute of the element is start/2-γ;
The end attribute of the element is end/2+γ;
append(T,start,end) to output;// output range
encoding }
```

- Let γ' have the same parent node p's left brothers u, no right brothers.start(γ') = (end(p)+end(u))/2-λ,end(γ') = (end(p)+end(u))/2+λ,γ∈(0,0.5)
- Let γ' have the same parent node p's right brothers u, no left brothers.start(γ') = (start(p)+start(u))/2-λ,end(γ') = (start(p)+start(u))/2+λ,λγ(0,0.5)
- Let γ' have the same parent node p's left brothers u, right brothers u'.start(γ')=(end(u)+start(u'))/2-λ,end(γ')=(end(u)+start(u'))/2+λ,λγ(0,0.5)
- Let γ' left and right brothers of have not the same parent node.start(γ')=(start(p)+end(p))/2-λ,end(γ') = (start(p)+end(p))/2+λ,λγ(0,0.5)

Definition 2: Non-woven into the document root node is encoded as a triple (docID, start, end). start is the value of the preorder traversal of the XML tree, end is the value of second access visit the node after traversing the sub-tree. The same docID document, any two nodes u and v is the ancestor/descendant relationships, if and only if the begin(u)<begin(v)/\end(v)<end(u). Different docID document, u is the element in the original document, γ' is the set of woven into nodes R' root element. Satisfy begin (u) <begin (γ') /\ end (v) <end (γ'), then R'is u'ssubtree whcih R' elements are all descendants of u. Parent/child relationship structure, In addition to the normal internal connecting within two documents, Parent/child relationships may occur between the original document and the root node of the woven document.

Algorithm 1 Implementation of start and end coding to generate of the root node of the woven document tree and algorithm 2 use SAX technique to sequence traversal for the woven documents, at the same time generate the rest of the node code. Algorithm 2 sent BEGINELEMENT and END_ELEMENT event when access nodes, according BEGIN_ELEMENT events accumulating variable n and n pushed onto the stack. When sending END_ELEMENT events, variable n continue to accumulate, at the same time pop the top element of the stack out of the stack with accumulative variable n for start and end nodes values respectively. The output order is the node sequence of after the root sequence of the document tree.

IMPROVED QUERY ALGORITHM

Xpath query path expression is the core technology of XML query, path expressions to describe the target locating method in XML document tree. In path expression calculation strategy of decomposition, the calculation results of the basic structure relationship between nodes becomes the key to query processing operations. In this study, in a relational database system to realize effective contain join operation was studied, put forward the solution to weave files with the original XML document contains relational structure join algorithm.Alogrithm 2 Algorithm for getting weaving document code

```
Algorithm:AOXMLEncdoe(T,nleft,nright)
{ Define variable curNode store the current access
node;
  Set a variable n=0;
Initialize the Stack Stack, set to null
while(ret= Sequential access XML document tree nodes)≠
FINISHED)
{
  if(Determine whether the current node of ret
is the specified node of curNode)
    {output nleft&nright;}
  Else
    { if (Determine whether the current element of ret
is the element BEGIN_ELEMENT)
      {  n++;
      Push the element N into the stack STACK;
      Output the start attribute of the element ret };
    if (Determine whether the current element of ret is
the element END_ELEMENT)
      {n++;
      Pop the top element of the stack STACK,join
with the element N;
      Output the results of join;}}
return NULL;}
```

The key of based on aspect oriented structure join algorithm is on the basis of implementing the different documents internal join, solve the structural join between documents. In the original ancestor element list Alist and descendant elements list Dlist according to the document number docID divided into a certain number of subsets, then structure join task converted into the join between the corresponding subset inside and subsets.

Alogrithm 3: Algorithm for AO_Join

```
Algorithm: AO_Join(Alist,Dlist)
{Make a and d respectively pointing the current
record of Alist and Dlist;
Set the stack to null;// Record the information of the woven node
while( A and d are not pointing to the bottom of the table ){
Order to find the elements have the same docID in
Alist and Dlist.
if(p.start=0andp.ParentOrder<>0)
Push the P node information into the stack ,
Push the corresponding docID in Dlist into stack;
if(d.begin<a.begin)
Find node d satisfying containment relationship in Dlist
if(a.begin<d.begin)
Find node a satisfying containment relationship
in Alist;
Complete all the containing elements in the table join
with a and output;}
while(stack is not null)
{Pop element a;
Order to query Alist,positioning the element;
Complete a join with d and d follow-up satisfying
containment relationship of the nodes;}}
```

Proposition 1: List by (docID, begin) orderly arrangement, any given a node x∈ List, the node x in the List all descendants of the node in the List are from node x n +1 section of the node List.n is the number of the woven document of next node.

Proposition 2: List by (docID, begin) orderly arrangement, any given a node x∈List,the node x in the list of possible descendants of the first node is first node of first node string of n +1 section nodes string;and x all descendants of nodes may occur in up to n +1 section nodes string, until traversing the all documents tree so far.

Improved merging algorithm: Improved merge join algorithm idea is: Participate join two relational list Alist and DList, Alist of the appearance of the tuples in order to search the Dlist table, complete each document internal structure join and record of the woven document and the woven node information(start=1 and ParentOrder<>1 tuple information), complete the structure join between the woven document and the original document. Each element node is a four-tuple<docID, start,end, ParentOrder>.

AO_Join algorithm is based on the MPMGJN,realize the join operation between elements within the document and pushthe woven node into stack.And then pop the node out of the stack in turn and realize join between the woven node and the woven document.The algorithm repeatedly scanning list to realize the join,the worst case algorithm's time complexity is O(|Alist|*|Dlist|).

Analysis the relationship of the node elements woven into the document and the original document ancestor nodes,all the elements are woven into the document whcih areweaving node descendant nodes,so the join may occur when the sequential scan Alist nodes improved AO_Join algorithm M_AO_Join using one-way scan mode,namely complete the join in the documents at the same time,complete the join of document and the woven document.M_AO_Join algorithm uses the index to locate,achieve by the woven document root node to locate the weaving node,achieve improvement of the algorithm.Algorithm based on the definition of extended XML Schema to realize weave,corresponding relationship of the woven document root node,so as to realize rapid node localization, see algorithm 4.

Alogrithm 4: Algorithm for M_AO_Join

```
Algorithm: M_AO_Join(Alist,Dlist)
Make a and d respectively pointing the current record
of Alist and Dlist ;
Set the stack to null;// Record the information of the
woven node;
According to XML Schema document tree,establish
index relationship between the woven node and the
woven document;
while( A and d are not pointing to the bottom of the table )
{ Order to find the elements have the same docID in
Alist and Dlist.
if(p.start=0andp.ParentOrder<>0)// Judge node for
the woven document;
{Query the specified node in the stack;
Complete the elements in the table join with the
woven node and output;}
if(d.begin<a.begin)
{ Find node d satisfying containment relationship in
Dlist;
if(a.begin<d.begin)
{ Find node a satisfying containment relationship in
Alist;
Complete all the containing elements in the table join
with a and output;}}}
```

Improvedancestor/descendant algorithm: The improved Stack-Tree algorithm is based on the Stack merge join algorithm. Original Stack-Tree algorithm to realize the structure of the join between nodes within a single document,the improved Stack-Tree algorithm will realize the structure join algorithm between XML document and the woven document, called Ext-Stack- Tree algorithm.The algorithm uses indexing technology in the process of structural join algorithm to skip node does not participate in the join as follows algorithm 5.

In order to reflect when the algorithm in dealing with a containment relationship structure join algorithm,using the index to skip the ancestors nodes don't participate in link, achieve the style of Fig. 1 reflect.Coarse line represents the nodes in the ancestorslist AList [begin, end] range, thin line represents the nodes in the descendants list DList [begin, end] range as Fig. 1. Solid lines with arrows indicate woven information, the arrow head is the woven node and the arrow tail is the weave node.Short dash line with arrows indicates the ability to skip of the node,thick dashed line with arrows indicates on the basis of node localization by index.Node a_{1i} represents node of the ancestors' nodes list in document 1, node d_{1i} represents node of the descendants nodes list in document 1. Node a_{2i} represents node of the ancestors' nodes list in document 2, node d_{2i} represents node of the descendants nodes list in document 2.

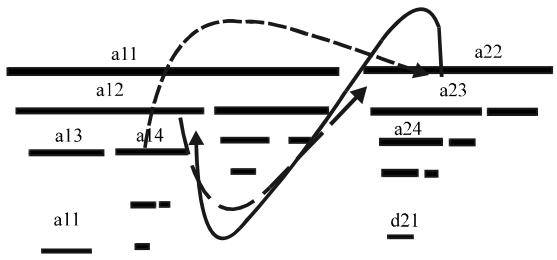

Fig. 1: Skip ancestor node with index

Alogrithm 5: Algorithm for Skipping ancestor node

```
Algorithm: Ext-Stack-Tree (Alist, Dlist, WList)
 STACK=NULL;
 a←the first Element of AList;
 d←the first Element of DList;
 w←the first Element of WList;
 while(a is NULL or d is NULL)
  { if (a is the ancestor of d)
    { tmp=a;
     while( tmp is the ancestor of d)
     {  PUSH(STACK,tmp);
tmp=the Next Element in AList;}
while(Not EMPTY(STACK))
{ OutPut(POP(STACK),d);
 d=the Next Element in DList;}
      else if (a.end<d.begin)
        { if( NOT EMPTY (STACK))
              { while(GetTOP(STACK) is before d)
L=POP(STACK);}
      else   L=a;
Let a be the element in AList and in WList having the first element.
OutPut all of d in weave document with Stack-tree; }}}}
```

The process of the original Anc_Desc_B + algorithm in solving the use of indexing technology can realize the ancestor nodes of skip not participate in join is push all ancestors of the first element d_{11} in the descendants list DList in the ancestor list AList into the stack,and output the results of them join with d_{11}.Pop-up node from the stack in turn,and all the nodes in the list AList detecting whether the d_{12}.Node's ancestors, until is d_{11} ancestors to continue into the stack and output join,end until ancestors list AList and descendants list DList are empty.Ext-Stack-Tree algorithm is use the idea to skip all nodes do not participate in join between the documents.

0125167974412516787201251668480251659264125166028812516613121251662336125166336012516643841251665408125166745612516776961251669504125167052825167155212516725761251673600125167462412516756481251676672125168076 81

As shown as Fig. 2 reflects the algorithm is how to use the index to skip nodes do not participate in join between the documents, the process is as follows:

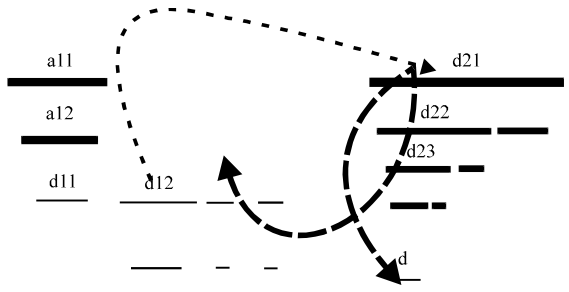

Fig. 2: Skip descedent node with index

- Push a_{11}, a_{12} and a_{13} in the list AList into stack and output the results of them join with d11
- From the stack pop-up a_{13} and a_{12} in turn
- Using the B + tree index, after a_{12} is popped from the stack, locating the a_{22} directly
- Push a_{22} in the list AList into stack and output the results of them join with d_{21}
- End until list AList and list DList are empty.

Processing in the step (3) to realize the nodes between the documents to skip,here node a_{22} is the root element of the woven elements of the list AList is shown as Algorithm 6.

Algorithm 6.Algorithm for Skipping descendant node

```
Algorithm: Ext2-Stack-Tree (Alist, Dlist, Wlist)
STACK=NULL;
       a?the first Element of AList;
     d?the first Element of DList;
   w?the first Element of WList;
   while(a is NULL or d is NULL)
    { if (a is the ancestor of d)
       { tmp=a;
while( tmp is the ancestor of d)
{  PUSH(STACK,tmp);
tmp=the Next Element in AList;}
While (Not EMPTY (STACK))
  { OutPut(POP(STACK),d);
    d=the Next Element in DList;}
     else
       if (NOT EMPTY (STACK))
            Let a be the element in DList        and in WList having the first
element;
      else   d=the Next Element in DList;
OutPut all of d in weave document with Stack-tree; }}
```

RESULTS

To test the rationality, effectiveness and practicability of the algorithm proposed in this study, we design the corresponding experiment.On the same platform using the Access database to establish different documents, comparing and analyzing the structure join algorithm performance.Test platform is Pentium4 PC, clock speed 2.4GHZ, RAM 1GB.The example design XML document automatically generates basically in line with the XML document of XML Schema document to weave node, respectively accounted for 10, 20, 30, 50 and 70% of the entire document node, proportion of different test query time of the algorithm.Test 2 kinds of algorithms for different data sets, running different query performance.Here to select the four test query examples, respectively is the Person/Name, TA/City as follows Fig. 3. It can conclude that the execution time of the algorithmincreaseswith the increase of the proportion of

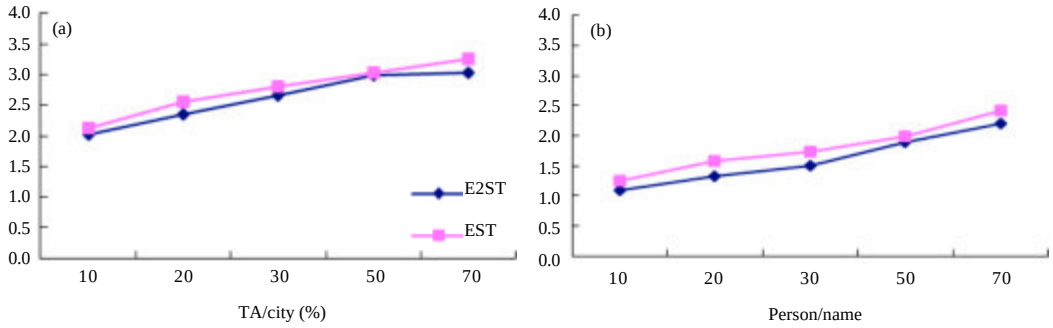

Fig. 3(a-b): Excution time for the different weave ratio

weavingdocument and increasesgreatly. In thefuturestudies, we'll improve the algorithm andraise theimplementingefficiencyofthealgorithm.

ACKNOWLEDGMENT

The authors would like to thank for the support of Natural science Foundation of Liaoning Provinceunder Grant 2013020032. The authors also thankfor the support of the Education Department of Liaoning Province under Grant L2011021.

REFERENCES

Li, G., J. Feng, Z. Yong, T. Na and L. Zhou, 2006. Exploiting even partition to accelerate structure join. Proceedings of the 7th International Conference on Web-Age Information Management Workshops, June 12-13, 2006, The Institute of Electrical and Electronics Engineers, Inc., Hong Kong, China, pp:13..

Liu, L., J.H. Feng, G.L. Li, Q. Qian and J.H. Li, 2008. Parallel structural join algorithm on shared-memory multi-core systems. Proceedings of the 9th International Conference on Web-Age Information Management, July 20-22 2008, The Institute of Electrical and Electronics Engineers, Inc., Zhangjiajie Hunan, pp:70-77.

Navasa, A., M.A. Perez-Toledano and J.M. Murillo, 2009. Developing aspect-oriented software architectures: A framework definition. Proceedings of the 4th International Conference on Software Engineering Advances, September 20-25, 2009, The Institute of Electrical and Electronics Engineers, Inc., Porto, pp: 331-338.

Wan, C.X., 2008. Technology of XML Database. 2nd Edn., Tsinghua University Press, Beijing, China, pp:151-172.

Wei, D.P. and X.L. Wei, 2012. Structural join oriented XML data compression. Proceedings of the 3rd World Congress on Software Engineering, November 6-8, 2012, The Institute of Electrical and Electronics Engineers, Inc., Wuhan, pp: 29-33.

Wu, S.P. and H. Yuan, 2009. Study on a new structural join algorithm for XML query. Proceedings of the International Symposium on Intelligent Ubiquitous Computing and Education, May 15-16, 2009, The Institute of Electrical and Electronics Engineers, Inc., Chengdu, pp: 417-419.