퍼지 및 지능시스템학회 논문지 2007, Vol. 17, No. 6, pp. 766-772

역방향 레이블 경로를 이용한 XML 문서의 선형 경로 질의 처리

Linear Path Query Processing using Backward Label Path on XML Documents

박충희* · 구흥서** · 이상준***
Chung-Hee Park, Heung-Seo Koo, and Sang-Joon Lee

* 제주대학교 컴퓨터공학과
** 청주대학교 정보기술공학부
*** 제주대학교 컴퓨터공학과

요 약

XML의 광범위한 사용으로 XML 저장과 질의 처리에 관한 많은 연구가 이루어지고 있다. 하지만 기존의 경로 질의 처리에 대한 연구들은 한 개의 대규모 XML 문서나 동일한 구조를 가진 문서들에 대한 저장, 검색에 초점이 맞춰져 있어서, 상이한 구조를 가진 대규모 문서들에 대해서 부분 매치 질의(partial match query)를 효과적으로 지원하지 못하는 단점이 있었다. 본 논문에서는 상이한 구조를 가지는 대규모 문서들에 대해서도 부분 매치 질의를 효과적으로 지원할 수 있는 관계형 테이블을 이용한 새로운 인덱스 구조를 제안하였다. 본 방법은 경로 정보를 저장할 때 기존의 연구에서 사용된 순방향 레이블 경로 대신 역방향 레이블 경로를 사용하여 B^+-트리 인덱스를 구축함으로써, 부분 매치 질의 처리시 구축된 인덱스를 이용하여 질의에 해당되는 레이블 경로들을 효율적으로 찾을 수 있도록 하였다.

키워드 : XML, 선형 경로식, 부분 매치 질의, 역방향 레이블 경로, 데이타베이스

Abstract

As XML is widely used, many researches on the XML storage and query processing have been done. But, previous works on path query processing have mainly focused on the storage and retrieval methods for a large XML document or XML documents had a same DTD. Those researches did not efficiently process partial match queries on the differently-structured document set. To resolve the problem, we suggested a new index structure using relational table. The method constructs the B^+-tree index using backward label paths instead of forward label paths used in previous researches for storing path information and allows for finding the label paths that match the partial match queries efficiently using it when process the queries.

Key Words : XML, linear path expression, partial match query, backward label path, database

1. 서 론

XML[1]은 데이타 교환과 문서 표현을 위한 실질적인 표준 언어이다. XML 형식의 데이타를 생성, 변형, 조작하기 위한 여러 도구들과 표준이 등장하면서 다양한 분야에서 많은 XML 문서들이 생성되고 있다. 따라서 우리는 상이한 구조를 가지는 대규모의 XML 문서들을 가지게 되었으며 그것들로부터 원하는 정보를 빠르게 추출하는 것이 중요해졌다.

XML 문서에 대한 질의 언어로 XML-QL, XQL, Quilt, XQuery 등이 제안되어 왔으며 이들 질의 언어들은 계층적인 문서의 구조 검색을 위해 XPath[2]와 같은 정규 경로식 형태의 질의어를 기본으로 삼고 있다. XPath 질의는 여러 경로식을 포함할 수 있어서 각각의 경로식에 대한 효율적인 처리는 전체 질의 성능에 매우 핵심적인 부분을 차지한다. 경로식상에 부모-자식 관계성 '/'만을 가지는 질의를 전체 매치 질의(full match query)라고 질의 1과 같이 경로식상에 '/'외에 조상-자손 관계성 '//'를 가지는 질의를 부분 매치 질의(partial match query)라 한다. 질의 1과 같은 부분 매치 질의는 서로 상이한 구조를 가지는 문서집합에 대해서 특히 유용하다[3].

• 질의 1:"//article//editor/name"

질의 1의 결과노드들을 추출하기 위한 질의 처리는 경로식과 매치하는(matching) 레이블 경로들을 찾는 과정을 수반한다.

본 논문의 목적은 분기 조건을 가지고 있지 않는 분기 경

접수일자 : 2007년 9월 12일
완료일자 : 2007년 11월 10일

로식(Branching Path Expression : BPE)의 특별한 형태인 선형 경로식(Linear Path Expression : LPE)의 효율적인 질의 처리 방법을 제안하는 것이다.

기존의 XML 문서에 대한 질의 처리 방법들은 [3]의 분류에 의거 크게 (1) 문서내 노드의 발생 정보를 사용하는 인스턴스-레벨 방법들[4,5,6,7]과 (2) 문서의 구조적인 정보를 관계 테이블에 저장하여 사용하는 스키마-레벨 방법들[3,8,9,10,11]로 나누어진다. 첫 번째 방법의 연구들은 경로식을 구성하는 각 레이블에 대응되는 모든 데이타를 추출하여 이들을 조인함으로써 결과 데이타를 구한다. 이로 인해 조인 연산을 수행하는 과정에서 많은 비교 연산을 초래한다. 따라서 부분 매치 질의를 지원하지만 질의 처리 효율성이 떨어진다. 두 번째 방법에서 XRel[8]과 XParent[9,10]는 레이블 경로 정보를 이용함으로써 불필요한 노드 인스턴스들간의 조인을 배제하여 빠른 처리를 보장하나 '//'로 시작하는 부분 매치 질의는 해당 경로들을 찾기 위해 레이블 경로 정보를 저장하는 테이블 전체를 탐색해야한다. 따라서 구조가 다른 문서 수가 증가함에 따라 저장되는 경로의 개수가 늘어나 성능이 급격히 저하된다. 그리고 EPIS[11]는 레이블 경로들을 각 레이블별로 분할 저장함으로써 부분 매치 질의 처리시 테이블 전체 스캔에 따른 성능 저하를 개선하였으나 길이가 긴 경로식에 대해 경로식상의 레이블 개수 만큼의 조인 연산을 유발한다. 한편 최근의 눈에 띄는 연구로 XIR-Linear[3]는 정보검색 분야에서 대규모 문서들을 빠르게 찾기 위해 사용해 온 역인덱스(inverted index) 기술을 XML 문서의 구조 검색에 활용한 방법으로 대량의 이질적인 문서 집합에 대해 유효한 성능을 보여 주었다. 그러나 제안된 텍스트 인덱싱 기술은 범용 RDBMS의 지원 기능이 아니며 특히 선형 경로식을 텍스트 기반 검색 조건식으로의 변환이 필요하며 변환 지원 가능 시스템은 더욱 제한적이다.

본 논문에서는 상이한 구조를 가지는 대량의 XML 문서들에 대해 선형 경로식을 효과적으로 처리하기 위하여 off-the-shelf RDBMS 기반 인덱스 구조를 제안한다. 레이블 경로를 저장하기 위한 관계 테이블 구조와 이를 활용한 부분 매치 질의 처리 방법에 대해 논의한다. 그리고 실험을 통해 제안한 기법이 기존의 인덱스들에 비해 높은 검색 성능을 제공함을 증명한다.

본 논문의 구성은 다음과 같다. 2장에서는 기존 XPath 질의 처리 방법들에 대해 살펴보고, 3장에서는 본 논문에서 사용하는 XML 문서 모델을 소개한다. 4장에서는 3장의 문서 모델을 기반으로 XML 인덱스 구조와 질의 처리 방법을 제안한다. 5장에서는 성능평가를 통하여 기존 연구와의 비교 분석 결과를 제시한다. 마지막으로 6장에서는 본 논문의 결론을 내린다.

2. 관련 연구

경로식을 처리하는 방법은 [3]에서 분류한 바와 같이 크게 인스턴스-레벨 방법과 스키마-레벨 방법으로 나누어질 수 있다. 인스턴스-레벨 방법은 노드의 시작(start)과 끝(end)의 위치와 같이 XML 문서에 나타난 단어들의 발생 정보만을 사용한다[4,5,6,7,12,13,14]. 반면 스키마-레벨 방법은 경로식과 매치되는 노드들을 찾기 위해서 레이블 경로와 같은 구조적인 정보를 사용한다. 본 장에서는 제안방식의 기본 구조가 되는 관계형 테이블을 사용하는 스키마-레벨 방법[3,8,9,10,11]에 대해 레이블 경로 정보를 저장하는 테이블 구조와 테이블을 이용한 경로식 처리 방법 중심으로 논의한다.

2.1 인스턴스-레벨 방법

이 방식에서의 질의는 XML 트리내 각 노드에 할당된 영역 정보(시작과 끝 위치)를 비교함으로써 처리된다. 두 개의 노드 영역 값들 간의 단순 포함관계를 비교함으로써 부모-자식이나 조상-자손 관계성을 판별한다. 스키마-레벨 방법이 스키마 정보를 먼저 이용하여 검색 결과에 근접한 인스턴스들로 한정하는 반면, 이 방식은 스키마 정보를 사용할 수 없기 때문에 질의 처리 비용이 훨씬 높다. 그러므로 이 방식은 문서 수가 증가함에 따라 질의 처리 비용이 매우 빠른 속도로 높아지는 단점이 있다.

2.2 스키마-레벨 방법

관계 테이블을 이용한 스키마-레벨 방법은 문서내의 모든 레이블 경로들이 질의 처리 이전에 테이블에 저장되어 질의 처리시 사용된다.

2.2.1 XRel

XRel[8] 방식에서 XML 트리의 구조 정보는 다음과 같이 4개의 테이블에 저장된다.

```
Path (label_path_id, label_path)
Element (document_id, label_path_id,
        start_position, end_position, sibling_order)
Attribute (document_id, label_path_id,
        start_position, end_position, value)
Text (document_id, label_path_id,
        start_position, end_position, value)
```

Path 테이블은 문서들내에 있는 모든 레이블 경로들과 그들의 식별자들을 저장하며 Element 테이블은 엘리먼트들에 대한 정보를 저장한다. 각 노드는 그 노드를 포함하는 문서의 식별자, 그 노드에서 종료되는 레이블 경로의 식별자, 노드의 시작과 끝 위치에 대한 옵셋(offset)들, 노드의 형제(sibling)들 가운데 해당 노드의 순번(order) 등의 정보를 갖는다. Text 테이블은 단말 노드의 값들에 대한 정보를 저장하며 Attribute 테이블은 속성노드에 대한 정보를 저장한다. value 열은 텍스트 값 또는 속성 값을 저장한다.

XRel의 LPE 처리는 먼저 Path 테이블에 대해 SQL의 스트링 매치 연산자인 LIKE 연산자를 사용하여 질의 경로식에 매치되는 레이블 경로들을 찾는다. '//'로 시작하는 형태의 부분 매치 질의 처리는 인덱스 존재와 관계없이 Path 테이블 내에 있는 레이블 경로들을 모두 읽어야 한다. 따라서 레이블 경로에 B^+-tree 인덱스를 사용할 수가 없다. 결과 노드들을 얻기 위하여 XRel은 Element 테이블 내의 label_path_id 컬럼을 경유하여, 레이블 경로들과 매치하는 노드들을 찾는다.

```
SELECT   e1.document_id, e1.start_position,
         e1.end_position
FROM     Path p1, Element e1
WHERE    p1.label_path_id = e1.label_path_id and
  p1.label_path LIKE "#%/article#%/editor#/name" ;
```

그림 1. 질의 1을 위한 XRel SQL 문
Fig. 1. SQL statement in XRel for Query 1

그림 1은 XRel에서 LPE 질의 1을 처리하기 위해 생성된 SQL문으로, XRel SQL은 스트링 매치를 통해서 먼저 해당하는 Path 튜플들의 집합을 찾는다.

2.2.2 XParent

XParent[9,10]에서의 레이블 경로 정보를 저장하는 LabelPath 테이블은 XRel의 Path 테이블과 동일하다. 따라서 LPE 처리는 XRel과 동일한 과정을 수행한다. 다만 분기 경로식 처리 과정이 다르며 BPE를 처리하기 위해 XParent는 equi-join들을 사용한다. XParent의 스키마는 다음과 같다.

```
LabelPath(label_path_id, length, label_path)
Element(document_id, label_path_id, node_id,
        sibling_order)
Data(document_id, label_path_id, node_id,
     sibling_order, value)
Ancestor(node_id, ancestor_node_id, offset_to_ancestor)
DataPath(parent_node_id, child_node_id)
```

Element와 Data 테이블은 영역 정보 대신 노드 식별자를 사용한다는 점을 제외하면 XRel의 스키마와 동일하다. 노드들 간의 조상-자손 관계성과 그들 간의 레벨 수를 유지하는 Ancestor 테이블과 선택적으로 사용할 수 있는, 부모-자식 관계성을 유지하는 DataPath 테이블(일명 Parent 테이블)이 추가 되었다. 그림 2는 XParent에서 질의 1을 처리하기 위해 생성된 SQL문이다.

```
SELECT e1.document_id, e1.node_id
FROM   LabelPath p1, Element e1
WHERE  p1.label_path_id = e1.label_path_id and
   p1.label_path LIKE ".%/article.%/editor./name" ;
```

그림 2. 질의 1을 위한 XParent SQL문
Fig. 2. SQL statement in XParent for Query 1

2.2.3 EPIS

EPIS[11]는 문서들의 레이블 경로 정보들을 XRel, XParent와는 다른 방식으로 저장한다.

```
UniqPathElmTbl(ename, pathid, plevel)
PathOccurrenceTbl(pathid, docid, startpos, endpos)
TermTbl(term, termid)
TermOccurrenceTbl(termid, docid, pathid, position)
```

UniqPathElmTbl 테이블은 XML 문서들로부터 추출된 경로를 기반으로, 각 경로를 구성하는 레이블 단위로 분할하여 정보를 저장한다. 저장되는 정보는 각 레이블의 이름(ename), 레이블이 속해 있는 경로 식별자(pathid), 레이블이 경로에 나타난 위치(plevel)이며 마지막 위치에 나타나는 레이블에는 '#'을 붙여서 경로의 마지막 레이블을 표현한다. 예를 들어 경로식별자 3을 가지고 있는 "/book/author/name" 경로는 <"book", 3, 0>, <"author", 3, 1>, <"name#", 3, 2>와 같이 세 개의 정보로 분할되어 UniqPathElmTbl 테이블에 저장되며, 레이블 이름 저장 열에 대하여 클러스터링 인덱스를 생성한다.

PathOccurrenceTbl 테이블은 모든 발생 경로에 대한 정보를 저장하며 TermTbl 테이블은 XML 문서들에서 사용된 모든 용어들을 저장한다. TermOccurrenceTbl 테이블은 각 용어들에 대한 발생 정보를 저장한다.

EPIS의 LPE 처리는 먼저 경로식에 나타난 각각의 레이블에 대해 인덱스를 탐색하여 해당 레이블을 포함하고 있는 후보 경로 식별자들을 찾는다. 그리고 각각의 레이블을 모두 포함하는 경로를 찾기 위하여 후보 경로 식별자 집합간 조인 연산을 수행한다. 따라서 구조가 상이한 문서의 수가 증가되어 레이블 경로를 저장하는 테이블의 크기가 커지는 상황에서도 전체 테이블 검색 대신 인덱스를 이용하기 때문에 상대적으로 적은 시간을 요구한다. 다만 길이가 긴 경로식을 처리하는 경우에 레이블 개수 만큼의 조인 횟수가 늘어나 응답시간이 길어지는 단점이 있다. 그림 3은 EPIS에서 질의 1을 처리하기 위해 생성된 SQL문이다.

```
SELECT  e1.docid, e1.startpos, e1.endpos
FROM    UniqPathElmTbl p1, UniqPathElmTbl p2,
        UniqPathElmTbl p3, PathOccurrenceTbl e1
WHERE   p1.pathid=p2.pathid and p2.pathid=p3.pathid
and p1.plevel < p2.plevel and p2.plevel+1 = p3.plevel
and p3.pathid=e1.pathid and p1.ename ="article"
and p2.ename="editor"and p3.ename="name#" ;
```

그림 3. 질의 1을 위한 EPIS SQL문
Fig. 3. SQL statement in EPIS for Query 1

2.2.4 XIR-Linear

XIR-Linear[3]는 문서들의 레이블 경로를 저장하는 LabelPath 테이블을 제외한 나머지 테이블 구조는 XRel이나 XParent 모두 사용 가능하다. LabelPath 테이블의 구조는 XRel, XParent와 유사하다. 다만 레이블 경로의 개수가 아주 많은 경우에도 빠른 검색을 지원하기 위하여 저장된 경로 문자열에 대해 텍스트 검색 엔진에서 사용되는 역 인덱스를 적용하여 생성하였다. XIR-Linear의 LabelPath 테이블 구조는 다음과 같다.

```
LabelPath(pid, labelpath)
Inverted Index on label_path of the table LabelPath
```

LabelPath 테이블은 전체 XML 문서들에 포함된 모든 발생 경로들로부터 추출된 서로 다른 레이블 경로들(labelpath)과 그들의 경로 식별자들(pid)을 저장한다. 경로 레이블 저장 시 각 레이블 경로의 시작 레이블과 종료 레이블을 표시하기 위해 프리픽스로써 '$'와 '&'를 덧붙여 저장한다.

XIR-Linear에서 LPE 처리는 먼저 LPE를 키워드 기반의 텍스트 검색 조건(일명 정보 검색 표현식)으로 변환한다. 이때 near(w)라는 근접 연산자를 이용하여 일정한 변환 규칙에 따라 정보 검색 표현식을 생성한다.

```
SELECT    n1.did, n1.nodepath
FROM      LabelPath p1, NodePath n1
WHERE     p1.pid = n1.pid  and
 MATCH(p1.labelpath, "article" NEAR(MAXINT)
 "editor" NEAR(1) "name" NEAR(1) "&name") ;
```

그림 4. 질의 1을 위한 XIR-Linear SQL문
Fig. 4. SQL statement in XIR-Linear for Query 1

생성된 정보 검색식을 이용하여 기존의 스트링 매치 대신

키워드 기반의 텍스트 검색을 수행한다. 그림 4는 XIR-Linear에서 질의 1을 처리하기 위해 생성된 SQL문이다.

3. XML 문서 모델

본 논문에서 사용하는 XML 문서는 루트가 있고, 순서가 있으며, 레이블이 기록된 트리로 가정한다. 이 트리의 노드는 엘리먼트, 애트리뷰트, 값 중 하나를 나타낸다. 또한 이 트리의 에지(edge)는 엘리먼트-서브엘리먼트 관계성, 엘리먼트-애트리뷰트 관계성, 엘리먼트-값 관계성, 애트리뷰트-값 관계성 중 하나를 나타낸다. 엘리먼트와 애트리뷰트 노드는 XML 문서의 구조를 결정하는 요소로, 각 노드에게는 레이블과 고유 식별자를 부여한다. 그림 5는 XML 문서의 XML 트리 예를 보인다. 애트리뷰트는 레이블 앞에 프리픽스 '@'를 사용함으로써, 엘리먼트와 구분한다. 제시된 모델에 대해 정의 1, 2, 3, 4와 같이 레이블 경로와 경로식의 개념을 정의한다.

정의 1. 순방향 레이블 경로(forward label path)

XML 트리 내의 순방향 레이블 경로는 그 트리의 루트 노드로부터 특정 노드 p까지의 노드 레이블 l_1, l_2, ..., l_p($p \geq 1$)들의 연속으로 정의되고 $/l_1/l_2/.../l_p$로 표현된다.

정의 2. 역방향 레이블 경로(backward label path)

XML 트리 내의 역방향 레이블 경로는 그 트리의 특정 노드 p부터 루트 노드까지의 역방향 노드 레이블 l_p, ..., l_2, l_1($p \geq 1$)들의 연속으로 정의되고 $l_p\backslash...\backslash l_2\backslash l_1\backslash$로 표현된다.

정의 3. 순방향 선형 경로식(forward linear path expression)

순방향 선형 경로식은 $l_0o_1l_1o_2l_2...o_nl_n$로서 정의된다. 이때 l_i(i=1, 2, ..., n)은 그 경로내에 존재하는 i번째 레이블이고, o_j(j=1, 2, ..., n)가 '/'이면 l_{j-1}는 l_j의 부모이며, '//'이면 l_{j-1}는 l_j의 조상임을 나타낸다. 여기서, l_0는 모든 XML 문서들의 집합을 나타내는 XML 트리의 루트 노드로서 생략될 수 있다.

정의 4. 역방향 선형 경로식(backward linear path expression)

역방향 선형 경로식은 $l_no_n...l_2o_2l_1o_1l_0$로서 정의된다. 이때 l_i(i=1, 2, ..., n)은 그 경로내에 존재하는 i번째 레이블이고, o_j(j=1, 2, ..., n)가 '\'이면 l_j는 l_{j-1}의 자식이며, '\\'이면 l_j는 l_{j-1}의 자손임을 나타낸다. 여기서, l_0는 생략될 수 있다.

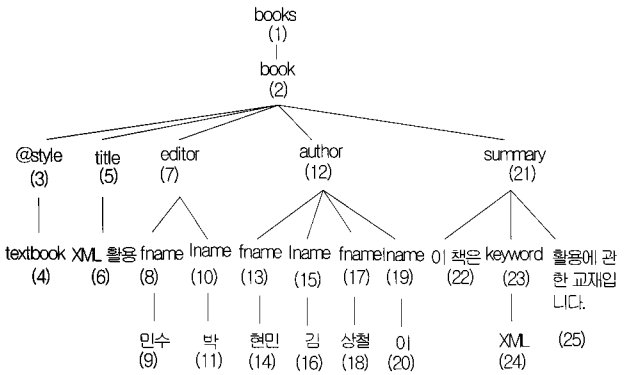


그림 5. XML 문서의 XML 트리 예

Fig. 5. XML Tree Example of a XML Document

4. XML 인덱스 구조와 질의 처리

4.1 XML 인덱스 구조

본 절에서는 효과적인 레이블 경로 탐색을 위하여 RDBMS 기반의 레이블 경로 정보 저장 구조를 제안한다. 먼저 경로 질의 처리 과정의 분석결과는 다음과 같다. (1) XPath의 경로 질의는 경로식 선두나 혹은 중간에는 조상-자손 관계성 '//'이 존재하나 경로식 맨 끝부분에는 나타나지 않는다. (2) 관계 데이터베이스의 스트링 매치 연산자 LIKE는 찾고자 하는 문자열 패턴이 와일드 카드 '%'로 시작하면 해당 열의 인덱스 존재와 관계없이 테이블 전체를 스캔한다. (3) 테이블에 저장된 모든 순방향 레이블 경로는 항상 동일한 루트 레이블로 시작하는 조상의 프리픽스 문자열들을 가지고 있어서 문자열 검색시 선택율이 떨어진다. 본 논문에서는 이런 분석 결과를 토대로 기존의 순방향 레이블 경로 문자열 대신 역방향 레이블 경로 문자열을 저장하는 방식을 제안한다. 이 방식은 순방향 형태의 기존 부분 매치 질의 대신 역방향 부분 매치 질의를 이용함으로써 RDBMS 스트링 매치 연산시 검색 문자열 패턴의 선두에 '%'가 나오는 것을 방지할 수 있다. 또한 현재 노드의 레이블부터 루트 레이블 순으로 저장하는 역방향 레이블 경로 문자열은 선택율이 높다. 제안하는 레이블 경로 저장 구조는 다음과 같다.

LabelPath(label_path, label_path_id, plevel)

LabelPath 테이블은 역방향 레이블 경로 정보를 저장하는 label_path 열과 경로 식별자인 label_path_id, 그리고 현재 레이블의 레벨을 저장하는 plevel로 구성되며 label_path 열에 대하여 클러스터링 인덱스(B^+-tree)를 생성하여 데이터의 순서를 물리적으로 유지한다.

LabelPath 테이블은 먼저, XML 문서에 포함된 모든 레이블 경로(순방향 레이블 경로)들을 추출하여 이에 해당하는 역방향 레이블 경로들을 저장한다. 이 때 레이블의 구분자를 위해 "\" 대신 "#\"를 사용한다. 예를 들어 순방향 레이블 경로 "/books/book/editor"가 경로 식별자 5를 가지고 있다고 가정하면 이 경로는<"editor#\book#\books#\",5,3>와 같은 역 방향 레이블 경로 형태로 저장된다.

label_path_id	label_path
1	#/books
2	#/books#/book
3	#/books#/book#/@style
4	#/books#/book#/title
5	#/books#/book#/editor
6	#/books#/book#/editor#/fname
7	#/books#/book#/editor#/lname
8	#/books#/book#/author
9	#/books#/book#/author#/fname
10	#/books#/book#/author#/lname
11	#/books#/book#summary
12	#/books#/book#summary#/keyward

그림 6. XRel의 Path 테이블

Fig. 6. Path Table in XRel

label_path	label_path_id	plevel
@style#\book#\books#\	3	3
author#\book#\books#\	8	3
book#\books#\	2	2
books#\	1	1
editor#\book#\books#\	5	3
fname#\author#\book#\books#\	9	4
fname#\editor#\book#\books#\	6	4
keyword#\summary#\book#\books#\	12	4
lname#\author#\book#\books#\	10	4
lname#\editor#\book#\books#\	7	4
summary#\book#\books#\	11	3
title#\book#\books#\	4	3

그림 7. 제안방식의 LabelPath 테이블
Fig. 7. LabelPath Table in the proposed method

그림 6, 7은 그림 5의 XML 문서 트리내 레이블 경로 정보들을 XRel의 Path 테이블과 제안방식의 LabelPath 테이블로 저장한 예를 보여주고 있다.

4.2 질의 처리

레이블 경로 정보를 담는 LabelPath 테이블을 제외한 나머지 테이블들의 구조는 XRel, XParent, EPIS 등 여러 가지가 선택 가능하며 여기서는 XParent 방식을 사용하여 질의 처리하는 예를 보인다. 먼저, 순방향 부분 매치 질의 "//article//editor/name"를 역방향 부분 매치 질의를 "name\editor\\article\\"형태로 변환하며 이때 SQL의 LIKE 연산을 위해 '\'는 '#\'로, '\\'는 '#%\'로 변환하여 SQL문을 작성한다. 그림 8은 제안된 LabelPath 테이블을 적용하여, LPE 질의 1을 처리하기 위해 생성된 SQL문이다. 제안된 구조는 전체 매치 질의는 물론 특히 '//'로 시작하는 부분 매치 질의에 대해 전체 테이블 검색 대신 인덱스를 사용하여 문자열 패턴에 해당하는 데이터만을 추출함으로써 빠른 응답시간을 제공한다.

```
SELECT   e1.document_id, e1.node_id
FROM     LabelPath  p1, Element e1
WHERE    p1.label_path_id = e1.label_path_id and
   p1.label_path LIKE "name#\editor#%\article#%" ;
```

그림 8. 질의 1을 위한 제안 구조의 SQL문
Fig. 8. SQL statement in the proposed method for Query 1

제안하는 저장 구조는 레이블 경로 수가 늘어나 경로 정보를 담고 있는 테이블 크기가 증가하는 상황에서도 빠른 탐색을 지원한다. 또한 EPIS의 단점인 긴 경로 질의에 대해 많은 수의 조인 연산 대신 단순히 해당 레이블 경로 문자열을 찾는 스트링 매칭 과정만을 수행함으로써 개선된 성능을 보인다.

4.3 기존 방식과의 질의 처리 방법 비교

표 1은 레이블 경로 정보를 저장하는 테이블들 XRel의 Path, XParent의 LabelPath, EPIS의 UniqPathElmTbl, 그리고 제안방식인 LabelPath 테이블의 구현 방법과 성능을 비교한 표이다. 본 논문의 목적은 순수한 RDBMS 기반의 인덱스 구조를 제안하는 것이기 때문에 XIR-Linear는 비교 대상에서 제외하였다.

표 1. XRel, XParent, EPIS, 제안방식의 비교
Table 1. Comparison of XRel, XParent, EPIS, Proposed method

성능관련 요소	XRel	XParent	EPIS	제안방식
레이블경로 탐색방법	테이블 스캔	테이블 스캔	인덱스 스캔 + 조인	인덱스 스캔
노드 식별방법	영역	노드 식별자	영역	영역 또는 노드식별자
레이블경로 탐색 비용	O(n)	O(n)	O(log(n x m))	O(log n)
경로 탐색 조인 횟수	0	0	m -1	0

n : 레이블 경로 테이블에 나타난 레이블 경로의 개수,
m : 단일 레이블 경로에 나타난 레이블의 개수

제안 방식의 성능이 XRel의 Path나 XParent의 LabelPath 보다 우수한 이유는 '//'로 시작하는 선형 경로식에 대해 해당 경로를 찾기 위해 전체 테이블을 스캔하지 않고 B^+-tree 인덱스를 이용하여 처리하기 때문이다. 그리고 긴 경로식에 대해서도 조인을 요구하지 않아 EPIS보다 우수하다. 레이블 경로 테이블에 저장된 레이블 경로의 갯수를 n, 단일 레이블 경로에 나타난 레이블의 개수를m이라 할 때, XRel, XParent, EPIS의 레이블 경로 매치 수행시간은 각각 O(n), O(n), O(log(n x m))이나 제안방식의 수행시간은 O(log n)으로 줄어든다.

5. 성능 평가

본 장에서는 제안방식의 질의 처리 성능을 검증하기 위해 질의 처리 효율성과 문서 증가에 따른 확장성 두 종류의 실험을 하였다.

5.1 실험 설정

본 실험에서는 Pentium3 930MHz CPU에 256 메모리를 가진 PC와 MS SQL Server 2000을 사용하였다. 본 논문에서 제안한 방법의 실험을 위해 필요한 모든 것들은 Java와 SAX2.0을 이용하여 구현하였다. 레이블 경로 개수의 증가에 따르는 확장성 실험을 위해서는 충분한 정도의 레이블 경로 개수가 필요한데 수집된 XML 문서들을 분석한 결과 한 개의 XML 문서 당 평균 8개 이하의 상이한 레이블 경로를 가지고 있었다. 따라서 실험 환경을 충족하기 위한 상이한 구조를 가지는 대량의 XML 문서는 대략 수십만 개에서 수백만 개의 문서들이 필요하여 본 실험에서는 일부 수집된 문서와 자체적으로 생성한 XML 문서들의 집합을 가지고 수행하였다. 본 연구의 성능 평가를 위해 우리는 XRel[8], EPIS[11]를 비교대상으로 선정하였다. XParent[9,10]는 XRel과 동일한 방법으로 선형 경로식을 처리하기 때문에 비교 대상에서 제외한다. 비교 대상내 경로 테이블에 대해 완전 클러스터 인덱스(Full Clustered Index)를 제공하였으며 사용된 질의의 SQL 변환은 수작업을 통해 수행하였다.

표 2. 실험 질의들
Table 2. Experimental queries

유형	XPath 질의
Q1	/PLAY/ACT
Q2	/PLAY/ACT/SCENE/SPEECH/LINE/STAGEDIR
Q3	/PLAY//SCENE/STAGEDIR
Q4	//ACT
Q5	//SCENE//LINE
Q6	//ACT//SPEECH//SPEAKER
Q7	//ACT/SCENE//SPEECH/LINE/STAGEDIR

표 2는 실험에서 사용된 질의이며 질의를 구성하는 경로의 길이, 조상-자손 관계 '//'의 개수를 고려하여 질의를 선정하였다.

5.2 실험 결과

그림 9는 사용된 실험 질의들을 수행한 결과 그래프이다. XRel은 EPIS보다 전체 매치 질의(Q1, Q2)와 '//'로 시작하지 않는 부분 매치질의(Q3)에 대해서 우수한 성능을 보였으나 '//'로 시작하는 부분 매치 질의(Q4, Q5, Q6, Q7)에 대해서는 항상 전체 경로 테이블을 읽기 때문에 EPIS보다 열악한 결과를 보여주고 있다. EPIS는 경로 식별자를 찾기 위해 먼저 레이블별로 인덱스 탐색을 통해 후보 경로 식별자들을 찾는다. 이후에 최종 경로 식별자를 찾기 위해 구한 후보 경로 식별자간 조인을 수행한다. 제안방식은 전체 매치 질의에 대해서는 XRel과 동일한 수준의 성능을 보이고 있으며 부분 매치 질의에 대해서는 EPIS보다도 더 우수한 성능을 보이고 있다. 이런 결과는 제안방식이 전체 매치 질의, 부분 매치 질의 즉 질의 유형과 관계없이 경로를 찾기 위하여 추가적인 조인연산 없이 항상 경로 테이블 탐색 대신 인덱스를 탐색하기 때문이다. 한편, 상이한 구조를 갖는 문서 개수의 증가함에 따라 레이블 경로 개수가 증가된다.

그림 10, 그림 11은 이러한 상황에서도 제안방식이 우수한 성능을 갖는다는 것을 보이기 위한 확장성 검증 실험이다. 그림 10은 긴 경로식을 갖는 전체 매치 질의 Q2에 대한 질의 처리 비용 그래프이고, 그림 11은 '//'로 시작하는 부분 매치 질의 Q6에 대한 그래프이다.

제안방식은 레이블 경로의 개수가 커지는 상황에서도 질의 유형에 관계없이 질의 비용이 거의 상수적(nearly constant)으로 증가한다. 제안방식은 레이블 경로를 저장하는 경로 테이블 크기가 증가함에 따라 그 성능의 차이는 더욱 더 커짐을 보인다.

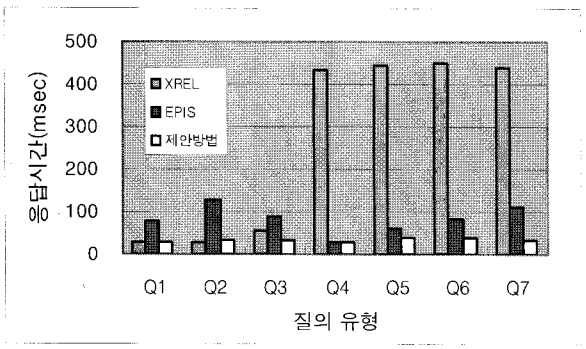


그림 9. 실험 질의 비용
Fig. 9. Costs of the experimental queries

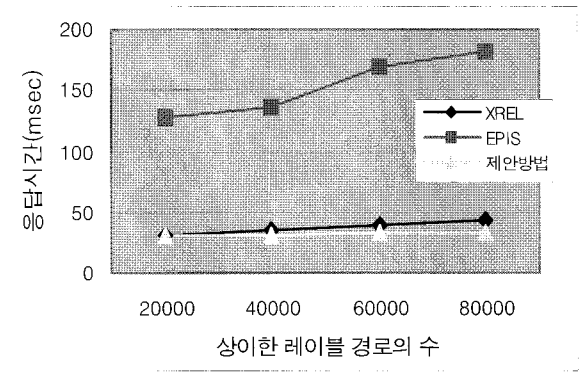


그림 10. 레이블 경로 개수 증가에 따른 Q2를 위한 질의 비용
Fig. 10. Query costs for Q2 by increasing number of label paths

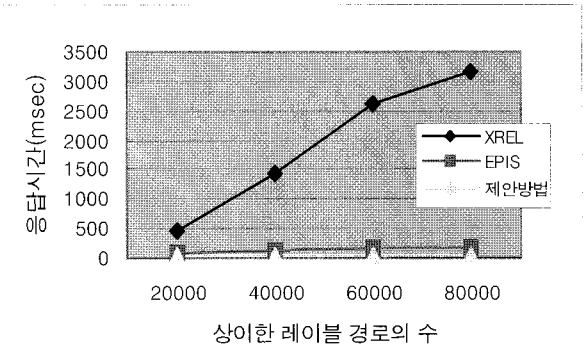


그림 11. 레이블 경로 개수 증가에 따른 Q6을 위한 질의 비용
Fig. 11. Query costs for Q6 by increasing number of label paths

6. 결 론

웹 환경에서 상이한 문서 구조를 가지는 XML 형식의 문서들이 빠르게 증가하고 있고 따라서 이러한 문서들로부터 사용자가 원하는 정보들을 추출하는 질의 처리가 매우 중요해지고 있다. 본 논문에서는 이러한 환경에서도 부분 매치 질의를 빠르고 정확하게 처리하기 위한 RDBMS 기반 인덱스 저장 구조를 제안하였다. 제안된 인덱스 저장 구조는 문서들의 레이블 경로 정보에 대해 역방향 레이블 경로를 저장하고 그 열에 대해 클러스터링 인덱스(B^+-tree)를 생성함으로써, 향후 경로식 처리시 전체 테이블을 스캔하지 않고 인덱스를 통한 빠른 접근을 허용하였다. 그리고 상이한 문서들이 급격히 증가하는 상황에서도 안정적인 검색 성능을 보장해 주었다. 우리는 제안방식의 성능을 XRel, EPIS와 비교, 실험하였다. 실험 결과는 XRel, EPIS보다 전체 매치 질의는 물론 부분 매치 질의에 대해서도 훨씬 빠른 질의 처리 성능을 보였다. 특히 상이한 구조를 가지는 문서들이 증가할수록 더 높은 성능을 보인다는 것을 실험을 통해 입증하였다. 그리고 인덱스를 데이타베이스 테이블로 저장, 관리하여 방식을 제공함으로써, 기존의 데이터베이스 시스템 변경이나 모듈 추가 없이 그대로 이용할 수 있도록 하였다. 향후 연구로

XML 문서 검색시 분기 경로식 형태의 복잡한 질의 처리를 위한 인덱스 성능 향상 연구를 진행하고 있다.

참 고 문 헌

[1] eXtensible Markup Language(XML), http://www.w3.org/XML/.

[2] J. Clark and S. DeRose, XML Path Language (XPath), W3C Recommendation, http://www.w3.org/TR/xpath, Nov. 1999.

[3] 박영호, 한욱신, 황규영, "정보 검색 기술을 이용한 대규모 이질적인 XML 문서에 대한 효율적인 선형 경로 질의 처리," *정보과학회논문지: 데이타베이스*, 제31권, 제5호, pp. 540-552, 2004.

[4] Q. Li and B. Moon, "Indexing and Querying XML Data for Regular Path Expressions," *In Proc. of VLDB*, pp. 361-370, 2001.

[5] C. Zhang et al., "On Supporting Containment Queries in Relational Databases Management Systems," *In Proc. of SIGMOD*, pp. 425-436, 2001.

[6] N. Bruno, N. Koudas and D. Srivastava, "Holistic Twig Joins: Optimal XML Pattern Matching," *In Proc. of SIGMOD*, pp. 310-321, 2002.

[7] 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," *In Proc. of IEEE ICDE*, pp. 141-152, 2002.

[8] M. Yoshikawa et al., "XRel: A Path-Based Approach to Storage and Retrieval of XML Documents Using Relational Databases," *ACM Trans. on Internet Technology*, Vol. 1, No. 1, pp. 110-141, 2001.

[9] H. Jiang et al., "Path Materialization Revisted: An Efficient Storage Model for XML Data," *In Proc. of ADC*, pp. 85-94, 2002.

[10] H. Jiang et al., "XParent: An Efficient RDBMS-Based XML Database System," *In Proc. of ICDE*, pp. 335-336, 2002.

[11] 민경섭, 김형주, "상이한 구조의 XML 문서들에서 경로 질의 처리를 위한 RDBMS 기반 역 인덱스 기법," *정보과학회논문지: 데이타베이스*, 제30권, 제4호, pp. 420-428, 2003.

[12] H. Jiang et al., "XR-Tree: Indexing XML Data for Efficient Structural Joins," *In Proc. of ICDE*, pp. 253-264, 2003.

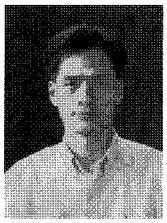
[13] H. Jiang et al., "Holistic Twig Joins on Indexed XML Documents," *In Proc. of VLDB*, pp. 273-284, 2003.

[14] I. Tatarinov, "Stroring and Querying Ordered XML Using a Relational Database System," *In Proc. of SIGMOD*, pp. 204-215, 2002.

저 자 소 개

박충희(Chung-Hee Park)
1989년 인하대 전자계산학과 학사.
1994년 동 대학원 전자계산공학과 석사.
2002년~현재 제주대 대학원 컴퓨터공학과 박사과정.

관심분야 : XML, 데이타베이스
Phone : 064-754-0310
Fax : 064-754-0313



구흥서(Heung-Seo Koo)
1985년 인하대 전산학과 학사..
1989년 동 대학원 전산학 전공 이학석사.
1993년 동 대학원 전산학 전공 이학박사.
1994년~현재 청주대 IT학부 교수.

관심분야 : XML 데이터베이스, u-헬스케어, ECM
Phone : 043-229-8492
Fax : 064-229-8432

이상준(Sang-Joon Lee)
1984년 중앙대 컴퓨터공학과 학사.
1989년 동 대학원 컴퓨터공학과 석사.
1992년 동 대학원 컴퓨터공학과 박사.
1993년~현재 제주대 컴퓨터공학과 교수.

관심분야 : Heuristic Algorithm, Web Database, Agent, XML 응용
Phone : 064-754-3655
Fax : 064-755-3620