퍼지 및 지능시스템학회 논문지 2007, Vol. 17, No. 3, pp. 430-433

XSTAR: XML 질의의 SQL 변환 알고리즘

XSTAR: XQuery to SQL Translation Algorithms on RDBMS

홍동권*, 정민경**

Dong-Kweon Hong and Min-Kyoung Jung

* 계명대학교 컴퓨터 공학과 교수

** 계명대학교 컴퓨터 공학과 대학원

요 약

XML이 다양한 분야에 널리 이용되면서 대용량의 XML을 효과적으로 관리하는 여러 가지 방법들이 연구되고 있다. 특히 지금까지 상업적, 기술적으로 성공적인 데이터 모델인 관계형 데이터베이스를 기반으로 한 여러 가지 방법들이 연구되고 있다. 본 논문은 관계형 DBMS를 사용하여 XML 질의어인 XQuery를 SQL로 변환하는 알고리즘인 XSTAR(XQuery to SQL Translation Algorithms on RDBMS)를 설계 및 구현한다. 본 연구의 XSTAR 알고리즘은 기본적인 XPath 뿐만 아니라 XQuery FLWOR 표현식, XQuery함수, 그리고 전문 검색(Fulltext 검색[8])과 관련된 몇몇 특수한 기능을 효율적으로 지원할 수 있으며, 질의의 결과 값을 XML 형태로 재생성하여 사용자에게 반환한다. 본 논문에서 제안하는 XSTAR 알고리즘은 현재 웹상에서 공개적으로 시범 운용 되고 있는 XML 문서의 관리 및 질의 처리 시스템인 XPERT(XML Query Processing Engine using Relational Technologies, http://dblab.kmu.ac.kr/project.jsp")의 질의 처리 엔진으로 사용되고 있다.

키워드 : XML 질의, SQL 변환, 전문 검색, 중간 결과값, 새로운 엘리먼트 구성 작업

Abstract

There have been several researches to manipulate XML Queries efficiently since XML has been accepted in many areas. Among the many of the researches majority of them adopt relational databases as underlying systems because relational model which is used the most widely for managing large data efficiently. In this paper we develop XQuery to SQL Translation Algorithms called XSTAR that can efficiently handle XPath, XQuery FLWORs with nested iteration expressions, element constructors and keywords retrieval on relational database as well as constructing XML fragments from the transformed SQL results. The entire algorithms mentioned in XSTAR have been implemented as the XQuery processor engine in XML management system, XPERT, and we can test and confirm it's prototype from "http://dblab.kmu.ac.kr/project.jsp".

Key Words : XML query, SQL translation, full-text retrieval, intermediate results, element constructors

1. 서 론

W3C(World Wide Web Consortium)에 의해 제안된 XML(eXtensible Markup Language)이 문서교환의 표준으로 지정된 후 XML은 인터넷, 음악, 과학, 디지털 도서관 등과 같은 매우 다양한 분야에서 활용되고 있다. 이렇듯 폭넓은 XML의 활용으로 인해 이를 효율적으로 저장, 검색하기 위한 연구가 요구되고 있다. 본 논문에서는 사용자가 필요한 기능들을 추가 및 수정할 수 있고 동시에 관계형 데이터베이스 시스템에서 제공하는 고급기능들까지 사용할 수 있는 관계형 환경을 기반으로 한 XML 질의 처리 알고리즘(XSTAR: XQuery to SQL Translation Algorithms on RDBMS)을 설계하고 구현한다. 본 연구의 XSTAR 알고리즘은 기본적인 XPath 및 XQuery FLWOR 표현식 외에도, XQuery함수, 그리고 전문 검색(FullText 검색[8])과 관련된 기능들을 효율적으로 지원할 수 있으며, 현재 웹상에서 시범 운용중인 XML 관리 및 처리 시스템 XPERT[16](XML Query Processing Engine using Relational Technologies)의 질의 처리 엔진으로써 존재한다.

본 논문의 구성은 다음과 같다. 2장에서는 XSTAR의 구조를 살펴본 뒤 알고리즘을 설명한다. 3장에서는 XSTAR 알고리즘의 기능 및 성능을 평가하고 4장에서는 결론 및 앞으로의 향후 과제에 대해 언급한다.

접수일자 : 2006년 11월 1일
완료일자 : 2007년 5월 3일

2. XSTAR의 설계 및 구현

본 장에서는 XSTAR 알고리즘의 전체 구조도, 테이블에 대해 살펴본 후 이를 설계 및 구현 하는 과정을 소개한다.

2.1 XSTAR의 색인 테이블

XSTAR 기법에서는 XPERT 시스템의 XML_Analyzer 엔진으로 생성되는 [그림 1] 구조의 색인 테이블[16]을 사용

한다. 우선 여러 개의 XML 문서들을 쓰임새에 따라 분류하여 하나의 컬렉션 단위로 관리한다. 이러한 각각의 컬렉션에는 다음과 같은 서로 다른 색인 테이블 집합이 생성되며 모든 질의는 사용자가 선택한 현재 컬렉션 (current collection)에 대해 이루어진다.

```
Collections (collection_id, Cname)
Cname_xml(doc_id, doc_name)
Cname_location(doc_id, path_id, path, depth, path_cnt)
Cname_element(doc_id, e_id, name, sibord, path_id, key_count, p_id, value, info, numbering)
Cname_attribute(doc_id, a_id, e_id, a_name, a_value)
Cname_word(word, doc_id, e_id, path_id, position, depth)
```

그림 1. XSTAR의 색인 테이블
Fig 1. XSTAR's index table

2.2 XQuery FLWOR 처리 알고리즘

본 연구에서는 [그림 2]와 같은 XQuery FLWOR 식을 SQL문으로 처리할 수 있는 알고리즘을 제시한다.

```
for $p in doc('books.xml')//book[field =
'Information Technology']
let $q := $p/author
where $p/title[contains(text(), 'XQuery')]
return <item price = "{$p/price/text()}"> {$q/last}
</item>
```

그림 2. XQuery FLWOR 표현식
Fig 2. XQuery FLWOR example

사용자가 XQuery FLWOR식을 입력할 경우 먼저 이를 분석하여 AST형태로 변환한다. 그리고 트리를 깊이 우선 탐색하되 각 서브트리 단위로 XSTAR의 XPath 처리 알고리즘[12,14]을 적용하여 XQuery FLWOR식의 각 바인딩 변수에 위치할 중간 결과값을 검색하는 SQL문을 생성한다. 예를 들면 [그림 3]과 같으며 p, q라는 2개의 바인딩 변수가 존재한다.

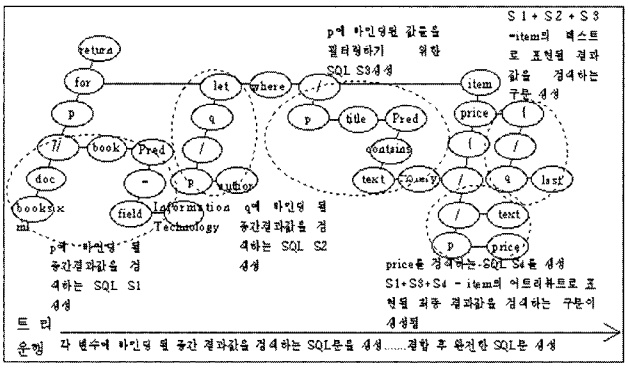


그림 3. AST를 운행하며 SQL문을 생성 및 결합
Fig 3. Constructing of SQL sentences

1) AST를 순회하면서 새로운 노드의 속성 값으로 표현될 노드를 만날 경우 현재까지 생성된 SQL문들을 적절히 결합하여 attr_table라는 뷰를 생성한다. [그림 3]인 경우 현재까지 생성된 SQL문들 중 변수 p에 바인딩 될 값을 질의하는 구문끼리 결합하여 item 노드의 속성 값으로 표현될 price노드와 관계형 데이터베이스 시스템에서 제공하는 rownum값을 검색하여 뷰를 생성한다.

표 1 attr_table 뷰
Table 1.View of attr_table

docid	eid	name	numbering	value	.....	row_id
1	76	price	#1#5#5#	49.99	.....	1
1	95	price	#1#6#6#	39.99	.....	2

2) AST를 순회하면서 새로운 노드의 텍스트 값으로 표현될 노드를 만날 경우 현재까지 생성된 SQL문들을 적절히 결합하여 text_table라는 뷰를 생성한다. [그림 36]의 경우 변수 p에 바인딩 된 노드들(book 엘리먼트와 그의 자손노드들) 중에서 last엘리먼트를 검색하여 뷰를 생성한다.

표 2 text_table 뷰
Table 2.View of text_table

doc id	eid	name	numbering	value	....	sibord
1	71	last	#1#5#3#1#	Brundage		1
1	72	email	#1#5#3#1#1#	Michael@person.....		1
1	86	last	#1#6#3#1#	Katz		1
1	87	email	#1#6#3#1#1#	Howard@pearson.....		1

3) 본 과정부터는 AST를 순회하지 않고 엘리먼트 구성 작업이 삽입된 구문이라면 모두 동일한 과정을 거치게 된다. text_table 뷰의 행들을 book 노드의 Dewey 순서 값으로 그룹핑한 뒤 Dewey 인코딩 기법으로 각 그룹의 루트 노드(last엘리먼트)들을 찾고 그들의 인코딩된 Dewey 순서 값들을 추출하여 Group_view 뷰를 생성한다. 이는 item 노드와 문자열로 결합될 last노드들을 정확하게 검색하기 위함이다.

표 3 Group_view 뷰
Table 3.View of Group_view

Child_numbering
1000050000300001
1000060000300001

4) Group_view 뷰의 값들을 정렬한 뒤 각 행의 rownum 값을 추출하여 Group_seq라는 뷰를 생성한다. 이 뷰를 통해 다음 단계에서 item 노드의 속성으로 삽입될 값과 텍스트로 삽입될 값들을 정확하게 결합시킬 수 있다. 단 새로운 엘리먼트의 애트리뷰트를 생성하는 구문이 없는 질의인 경우 본 단계를 생략할 수 있다.

표 4 Group_seq 뷰
Table 4. view of Group_seq

child_numbering	row_id
1000050000300001	1
1000060000300001	2

5) 전 단계에서 생성한 Group_seq 뷰의 child_numbering 컬럼 값과 동일한 값을 가진 엘리먼트들을 text_table 뷰에서 검색하고 row_id 컬럼 값과 동일한 값을 가진 엘리먼트들의 텍스트들을 attr_table 뷰에서 검색한 뒤 이들의 name 컬럼 값을 결합하여 Concatenation_table 라는 뷰를 생성한다. 이러한 기법으로 새로운 엘리먼트 구성작업으로 인한 불필요한 갱신현상을 줄이게 된다.

표 5. Concatenation_table 뷰
Table 5. View of Concatenation_table

name	t.numbering	t.value	.....	sibord
<item price = "49.99">last	#1#5#3#1#	Brundage	.....	1
<item price = "39.99">last	#1#6#3#1#	Katz	.....	1

6) Concatenation_table 뷰와 text_table 뷰를 결합하고 중복된 행을 제거함으로써 item 노드의 자식 노드로 위치할 값들을 표현한 뒤 최종 결과 뷰를 생성한다.

표 6. last_table 뷰
Table 6.View of last_table

docid	eid	name	numbering	value	info
1	71	<item price = "49.99">last	#1#5#3#1#	Brundage	71
1	72	email	#1#5#3#1#1#	Michael@pear..	72
1	86	<item price = "39.99">last	#1#6#3#1#	Katz	86

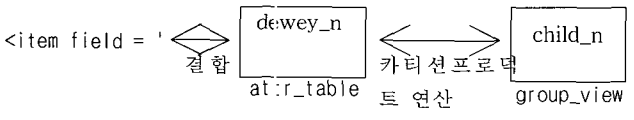
마지막으로 XML 재생성 과정을 거쳐 표 6을 XML 형태로 생성 및 출력한다.

지금까지 살펴본 내용과는 달리 2중 for문인 XQuery 경우 5단계로 나눠 처리한다. group_view를 생성하는 단계까지는 1중 for문의 처리방식과 동일하나 group_seq뷰를 생성하는 과정은 거치지 않고 concatenation_table뷰와 최종 last 뷰를 생성한다. 이때 카티션 프로덕트 연산을 하여 처리한다는 점이 1중 for문의 XQuery 처리방식과 다르다.

nested XQuery To SQL(AST node)

1)~3) XQuery To SQL 알고리즘과 동일

4) group_view의 child_n값과 동일한 값을 가진 행을 text_table(item노드가 삽입될 위치)에서 검색한 뒤 attr_table의 행(item노드의 attribute로 들어갈 값)의 Dewey_n값을 인코딩한 값을 기준으로 카티션 프로덕트 연산을 하여 2중 for문을 수행한 것과 동일한 개수의 item노드를 생성한다.



conctenation_table(t.name, t.dewey_n, t.value,,,,t.sibord, a.row_id)

5) item노드의 자식으로 출현할 서브트리들을 text_table로부터 검색하되 attr_table과 카티션 프로덕트 연산을 한다. 그리하여 4)과정에서 생성한 item노드와 동일한 개수의 서브트리를 생성하고 concatenation 테이블과 UNION All 연산을 한 뒤 최종 결과뷰를 생성한다. 단 아래의 최종 결과 뷰는 서로 동일한 Dewey_n값을 가지게 되는데 이때 row_id 컬럼을 기준으로 정렬함으로써 결과 뷰를 XML 형태로 재생성할 때 XML문서에 내재된 순서대로 데이터를 로드할 수 있다.

last_table(docid, eid, name, dewey_n, value, info, row_id)

3. 성능 평가

본 장에서는 XSTAR 알고리즘의 질의 처리 성능을 평가한다. 우선 비교대상이 되는 방식은 Dynamic Interval Encoding 기법[10]의 기본 XQuery 처리 알고리즘을 본 연구의 색인 테이블에 맞게 적용한 것으로 중간 결과 값들이 발생될 때마다 뷰를 생성하여 처리한다. 이러한 기법을 앞으로 편의상 d_XSTAR라고 부른다. 이 기법의 XQuery 처리 알고리즘은 jdk1.4.05와 JDBC, 결과값을 재생성하는 저장 프로시져는 PL/SQL를 이용하여 XSTAR 기법과 동일한 방식으로 구현하였다. 또한 차트의 데이터 테이블에 표기된 'DNF(Did not Finish)'는 실행을 마치지 못한 경우를 뜻한다.

성능 평가에 사용된 XML문서는 도서에 관한 내용을 담고 있는 books.xml문서와 한글이 포함된 books_korean.xml 이며 이들은 모두 'DBLab'이라는 컬렉션에 존재한다. 또한 본 평가에서 사용된 질의는 아래와 같다.

```
Q1: for $p in doc('books.xml')//book
    return <item title = "{$p/title/text()}"> {$p}
    </item>
Q2: for $p in doc('books.xml')//book
    where $p/@year = '2000'
    return <item person = "{$p//title/text()}">
    {$p//last} </item>
Q3: for $p in doc('books.xml')//book
    let $q := $p/title[contains(text(), 'Illustrated')]
    where $p/@year = '1994'
    return <item person = "{$p//last/text()}"> {$q}
    </item>
Q4:for $p in doc('books.xml')//book[publisher =
'Addison-Wesley' and field =
    'Information Technology']
    let $q := $p/contents
    where exists($p//rank/text() = "1")
    return <item> {$q/notion} </item>
Q5: for $p in doc('books.xml')//book[field =
'Information Technology']
    let $q := $p/contents[subjects = 'XML Query
    Language']
    where $q/notion[contains(text(), 'XQuery')]
    return $q/homepage
Q6: for $p in doc(books_korean.xml)//book
    for $q in doc(books.xml)//book[title = 'Data on
    the Web']
    let $k := count($q/author)
    where $k > 1
    return <contents subject =
    "{$p//subjects/text()}'> {$q/field} </contents>
```

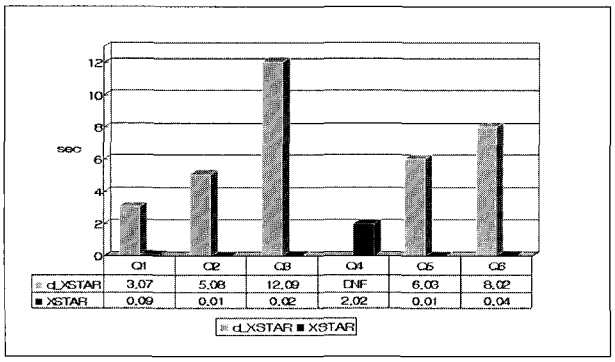


그림 4. XQuery FLWOR식 처리시간 비교
Fig 4.Processing time of XQuery FLWOR expression

[그림 4]를 통해 알 수 있듯이 본 논문에서 제안한 XSTAR방식이 훨씬 우수한 성능을 발휘한다. 이는 XSTAR 알고리즘이 여러 조건을 만족하는 최종 결과 엘리먼트들을 한꺼번에 필터링함으로써 XQuery의 중간 결과 값을 처리하기 위한 복잡한 과정들을 줄이기 때문이다.

4. 결 론

인터넷에서 급격히 사용이 늘어난 XML을 효과적으로 관리하는 데이터베이스 기술의 요구 사항을 만족시키기 위하여 관계형 데이터베이스를 이용하여 XML을 관리하는 많은 기술들이 연구되고 있다. 본 논문에서는 다음과 같은 여러 실제적인 문제들에 대한 해결방법을 제공하였다. 1) 사용자가 선택할 수 있는 여러 개의 XML문서들을 컬렉션으로 선택하고 모든 질의가 컬렉션 단위로 이루어지게 하였다. 2) 기존의 방식에서 제안한 복잡한 중간 과정들과 질의 횟수를 줄이고 질의 처리 성능을 극대화 하였다. 3) 입력되는 XQuery FLWOR식을 이미 정해진 과정을 통해 일정하게 처리함으로써 주어지는 질의에 따라 예측할 수 없는 복잡한 과정으로 처리하는 기존의 연구 기법들과 차별성을 두었다. 4) 새로운 엘리먼트를 삽입하여 최종 결과 값을 형성할 경우 새로운 엘리먼트의 이름을 기존의 엘리먼트명과 단순히 문자열을 결합시킴으로써 삽입으로 인한 불필요한 갱신현상을 줄였다.

본 논문에서 제안하고 구현한 XSTAR 알고리즘은 XML과 관계형 응용 모두에 적용될 수 있는 장점을 가지고 있다.

참 고 문 헌

[1] Meier, W., "eXist open source native XML database," http://eXist.sourceforge.net/

[2] Rotterdam, J., Veen, M., ed., "X-Hive/DB" http://www.x-hive.com/ 2004년 10월 검색

[3] Chen, C., Chio, B., Gapeyev, V., "Galax XML DB release0.5.0" http://www.galaxquery.org

[4] Oracle Home page "Oracle 10g SQL/XML" http://www.oracle.com/technology/products/database/application_development/sqlxml/index.html

[5] SQL Sever Home page "SQL Server 2000 XML Overview" http://www.microsoft.com/technet/prodtechnol/sql/2000/evaluate/xmlsql.mspx 2005년 7월 검색

[6] Dehaan, D., Toman, D., Consens, M., Ozsu, T., "A Comprehensive XQuery to SQL Translation using Dynamic Interval Encoding" SIGMODE 2003

[7] Grust, T., Sakr, S., Teubner, J., "XQuery on SQL Hosts" VLDB Conference 2004

[8] Buxton, S., Rys, M., "XQuery and XPath Full-Text Requirements" http://www.w3c.org/TR/xquery-full-text-requirements/

[9] "XQuery 1.0 Grammar Test Page" http://www.w3.org/2005/04/qt-applets/xqueryApplet.html

[10] Stylus Studio homepage http://www.stylusstudio.com/xml_download.html

[11] XPath axes http://www.w3.org/TR/xpath#axes

[12] Jung, M.K., Hong, D.K., "Design and Implementation of XQuery Processor on the RDBMS using Dewey order" 정보처리학회 춘계학술대회 2005. 5월

[13] Jung, M.K., Hong, D.K., Kim, K.Y., "Developing an XML query processing system using a relation database" ALPIT conference 2005, 6월

[14] Jung, M.K., Hong, D.K., "Design and Implementation of XQuery Processor on the RDBMS" 정보처리학회 추계학술대회 2005, 11월

[15] Jung, M.K., Hong, D.K., Nam, J.Y., "Design and Implementation of XML Analyzer on RDBMS" 정보과학회 춘계학술대회 2005, 7월

[16] Jung, M.K., Hong, D.K., Kim, K.Y., "XPERT: XML Query Processing System using Relation Databases" APIS conferentce 2006

저 자 소 개

홍동권(Dong-Kweon Hong)

2004년 14권 6호

정민경(Min-Kyoung Jung)
2004년 : 계명대 컴퓨터 공학과 석사
2006년~현재 : 계명대 컴퓨터 공학과 박사과정

관심 분야: 데이터베이스, XML
E-mail : time_mk@hanmail.netp