비트 분할 시그니처 화일을 이용한 XML 인덱스 구조

강인선[0], 홍석진, 이태원, 이석호
서울대학교 전기컴퓨터공학부
{eggie[0],jinny,warrior}@db.snu.ac.kr, shlee@cse.snu.ac.kr

XML Indexing Structure Using Bit-Sliced Signature File

Insun Kang[0], Seokjin Hong, Taewon Lee, Sukho Lee
School of Electrical Engineering and Computer Science, Seoul National University

요약

데이터베이스에 저장된 많은 양의 XML 데이터를 빠르게 검색하는 과정에서, 경로식을 만족하는 노드를 추출하는 부분은 가장 많은 비용을 요구한다. 기존 방법은 여러 번의 조인을 통해 이를 처리하기 때문에 많은 비용이 드는 단점이 있다. 본 연구에서는 각 노드의 경로에 대한 시그니처를 만들고, 질의문의 경로식에 대한 시그니처와의 비트연산을 통해 후보 경로 집합을 선택하는 방법을 제안한다. 이 방법은 조인 연산 없이 경로식을 처리하기 때문에 기존의 조인 연산 비용을 줄일 수 있으며, 기존 관계형 데이터베이스에 쉽게 적용시킬 수 있는 장점이 있다.

1. 서론

XML은 웹 문서의 표준으로서 뿐만이 아니라, 인터넷 상에서 주고 받는 데이타의 표준으로 자리잡고 있다. XML 형식으로 데이타를 표현함으로써 이종 시스템 사이에서도 쉽게 데이타를 주고 받을 수 있다. 이러한 XML의 장점 때문에 크고 작은 양의 데이타들이 XML 문서로 만들어져 왔다. 그런데, XML로 표현된 데이타가 DBLP 문서와 같이 방대한 양의 데이타인 경우에는, 빠른 데이타 검색과 추출을 위한 효율적인 인덱스 구조가 필요하다.

이러한 필요성 때문에 최근 XML 데이타의 효율적인 저장 및 검색 시스템에 관한 연구가 활발하게 이루어지고 있다. 그러나 기존의 연구에서 제안한 저장, 검색 구조들은 질의문 내의 노드 경로식을 처리하기 위해 여러 번의 조인이 불가피하다는 공통적인 문제점을 안고 있다.

본 논문에서는 비트 분할 시그니처 화일(BSSF; Bit-Sliced Signature File) 기법[1]을 이용, 노드의 경로를 표현하여, 노드 검색 속도를 빠르게 하는 방법을 제안한다. 질의문에서 찾고자 하는 노드의 경로식을 시그니처로 표현하여 저장된 노드 경로들의 시그니처와 비트 연산을 하면, 여러 번의 조인을 하지 않아도 원하는 노드를 효율적으로 찾을 수 있다.

본 논문의 구성은 다음과 같다. 먼저 2장에서 관련 연구들을 살펴보고, 3장에서 본 논문이 제안하는 XML 저장 및 검색 시스템에 대해 설명한 뒤, 4장에서 결론을 맺는다.

2. 관련연구

XML 문서에 대한 질의어로 Lore, XML-QL, XPath, XQuery 등이 소개 되었는데, 특히 W3C에서 작성 중인 XQuery[6]는 점차 대표적인 XML 질의어로 인식되고 있다. 그림 1은 W3C "XML Query Use Cases"[6]의 예제 문서이다.

```
<chapter>
    <title>Data Model</title>
    <section>
        <title>Syntax For Data Model</title>
    </section>
    <section>
        <title>XML</title>
        <section>
            <title>Basic Syntax</title>
        </section>
        <section>
            <title>XML and Semistructured Data</title>
        </section>
    </section>
</chapter>
```

그림 1 XML 문서 book.xml

다음 질의문 Q는 그림 1의 문서에 대해 작성한 간단한 XQuery 질의문이다.

```
Q:  for $t in /chapter/section/section/title
    where $t = "Basic Syntax"
    return $t
```

데이타베이스에 저장된 XML 데이타에 대해 질의를 처리할 때, 가장 큰 문제는 경로식으로 표현된 노드를 빠르게 탐색하는 방법이다. 질의 Q의 경우 "/chapter/section/section/title"에 해당하는 노드를 빨리 찾는 것이 관건이다. 따라서 본 논문의 목표는 XQuery에서의 경로식의 효율적인 처리에 있다. XML 데이타를 데이타베이스에 저장하는 기존의 방법들은 다음과 같은 문제점이 있다.

[4]에서 제안한 인라이닝(inlining) 기법은 DTD 정보를 통해 XML 문서에 적합한 관계형 데이타베이스 스키마를 구성하여 XML문서를 저장한다. 그러나, 인라이닝 기법은 트리상에서 높이 차이가 큰 두 노드를 거치는 경로를 처리 해야 할 때에는 여러 번의 조인 연산이 필요하다는 문제가 있다. 예를

들어, 그림1의 문서에 대해, chapter 테이블이 title을 애트리뷰트로 포함하고 있고, section 테이블이 title을 포함하고 있을 때, 질의문 Q를 처리하기 위해서 chapter 테이블과 section 테이블과 또 한번의 section 테이블 조인이 필요할 것이다.

[2]와[5]에서 제안한 numbering scheme 방법은 각 노드에 숫자 쌍을 부여하여, 임의의 두 노드의 포함관계를 숫자 쌍이 다른 숫자 쌍을 포함하는지의 여부로 판단한다. 이 방법은 질의문을 SQL로 변환하기 쉽다는 장점이 있지만, 노드의 경로식이 길어질 경우, 각 엘리멘트의 숫자쌍을 비교하기 위해 여러 번의 조인 연산이 필요하다는 문제는 다른 기법들과 마찬가지이다.

[3]에서도 시그니처를 이용한 XML 질의 처리 기법을 소개하고 있는데, 본 연구가 경로에 대해 시그니처를 작성한 것과는 달리 트리 구조상에서 노드 탐색시 하위 노드의 시그니처를 상위 노드가 소유하여 탐색시간을 단축하는 방법을 이용하고 있다. 그러나, 이 경우 트리를 탐색하면서 한 노드에 이를 때마다 질의 경로식과 시그니처를 비교연산을 하기 때문에 비용 절감에 한계가 있다.

본 논문에서는 이러한 기존의 기법들의 한계를 극복하여 노드의 경로식 처리 속도를 빠르게 하는 인덱스를 제안하여, 궁극적으로 질의문 처리 속도를 개선하고자 한다.

3. 비트 분할 시그니처 화일을 이용한 XML 경로 인덱스

3.1. 구조

본 논문에서 제안하는 시스템에서 XML 문서 데이타는 그림 2와 같이 시그니처 화일, 경로 테이블, 노드 테이블, 값 테이블, 문서 테이블에 저장된다. 시그니처 화일은 노드의 절대 경로에 대한 시그니처들을 저장하고 있고, 경로 테이블은 각 시그니처 경로 시그니처 값들이 정확히 어떤 경로를 의미하는지에 대한 정보를 저장하고 있다. 질의 경로식이 입력되었을 때, 이 두 가지 정보를 이용하여 정확한 경로를 경로 테이블에서 추출하고, 노드 테이블과의 조인을 통해 정확하게 원하는 노드를 선택한다. 값 테이블과 문서 테이블은 노드들의 값을 가지고 있는 것으로, 경로식을 처리할 때에는 참여하지 않는 요소이다.

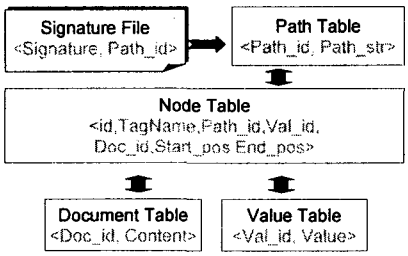

그림 2 저장 구조

3.1.1. 시그니처 화일(Signature File)의 구성

본 논문에서 작성하는 시그니처 화일은 노드 경로에 대한 시그니처이다. [1]에서 제안한 비트 분할 시그니처 화일 방법(BSSF; bit-sliced signature file)을 이용한다. 루트 노드에서 각 노드까지의 절대 경로에서, 절대 경로가 거쳐가는 노드들의 시그니처 값을 'OR' 비트 연산한 것이 노드의 경로 시그니처 이다.예제 XML 문서 D의 각 노드 이름에 대한 해쉬 값이 표1에 주어졌다.

chapter	10000001	title	00010100
section	01100000		

표 1 노드 이름에 대한 해쉬 값

노드 이름에 대한 해쉬값을 바탕으로, 루트에서 각 노드까지의 경로에 대해 시그니처를 만들면 표2와 같다.

/chapter	1	0	0	0	0	0	0	1
/chapter/title	1	0	0	1	0	1	0	1
/chapter/section	1	1	1	0	0	0	0	1
/chapter/section/title	1	1	1	1	0	1	0	1
/chapter/section/section	1	1	1	0	0	0	0	1
/chapter/section/section/title	1	1	1	1	0	1	0	1

표 2 노드의 경로와 그에 해당하는 시그니처

질의 처리시 검색 속도를 높이기 위해, 경로 시그니처의 집합을 비트 분할 시그니처 화일 형식으로 저장한다. 비트 분할 시그니처 화일의 i번째 행은, 각 시그니처의 i번째 비트들로 이루어진 시퀀스이다. 즉 시그니처의 길이를 M, 경로의 수를 N이라 하면, 표2에서 만들어진 경로 시그너처는 N×M 의 0과 1의 행렬 (matrix)로 볼 수 있고, 비트 분할 시그니처 화일은 M×N으로 변환된 행렬이라 할 수 있다. 따라서 문서 D를 저장할 때 구성되는 시그니처 화일은 표3과 같다.

1	1	1	1	1	1
0	0	1	1	1	1
0	0	1	1	1	1
0	1	0	1	0	1
0	0	0	0	0	0
0	1	0	1	0	1
0	0	0	0	0	0
1	1	1	1	1	1

표 3 비트 분할된 시그니처 화일

3.1.2. 경로 테이블(Path Table)의 구성

경로 테이블은 표2의 첫번째 열에서 사용한 경로들과, 경로 ID로 이루어진다. 표4는 예제 문서 D에 대한 경로 테이블이다.

1	/chapter
2	/chapter/title
3	/chapter/section
4	/chapter/section/title

5	/chapter/section/section
6	/chapter/section/section/title

표 4 경로 테이블

3.1.3. 노드 테이블 (Node Table)의 구성

노드 테이블은 표5와 같이 Tag이름, 경로 ID, 값 ID, 문서 ID, 시작 위치, 끝 위치로 구성된다. 경로식을 처리할 때 핵심이 되는 것은 TagName, Path_ID 이다. Val_ID는 노드의 값을 비교해야할 때에 값 테이블과 조인하기위해 사용되며, Doc_ID, Start_Pos, End_Pos는 문서에서 노드 전체를 가져올 때 이용한다.

1	chapter	1	NULL	1	1	143
2	title	2	1	1	10	34
3	section	3	NULL	1	35	89
4	title	4	2	1	44	79
5	section	3	NULL	1	90	233
6	title	4	3	1	99	116
7	section	5	NULL	1	117	162
8	title	6	4	1	126	152
9	section	5	NULL	1	163	223
10	title	6	5	1	172	213

표 5 노드 테이블

3.2. 질의 경로식 처리 과정

3.1의 구조를 이용하여 질의문 내의 경로식을 처리하는 과정은 크게 다음과 같이 크게 세 단계를 거친다.

1. 경로 테이블에서 경로식을 만족하는 후보 집합 만들기
   - 해쉬 테이블을 이용해 질의 경로식에 해당하는 질의 시그너처를 만든다.
   - 질의 시그너처에서 1로 세팅된 모든 i번째 비트에 대해, 시그니처 화일의 i번째 행들을 서로 AND 연산하여, 비트 맵을 만든다.
   - 비트 맵에서 1로 세팅된 모든 j번째 비트에 대해 해당하는 경로를 경로 테이블에서 찾아, 후보 경로 집합을 만든다.
2. 실제로 질의를 만족하는 경로만 선택하기
   - NFA를 이용해 후보 경로 집합에서, 실제 질의 경로식을 만족하는 경로들을 찾아낸다.
3. 노드테이블과 조인하기
   - 2의 과정에서 정확하게 질의 경로식을 만족하는 경로들을 선택하였으므로, 그 경로들로 이루어진 집합과, 노드 테이블을 조인하여 질의 경로식에 해당하는 노드에 접근할 수 있다.

처리 과정의 구체적인 예는 3.2.1과 같다.

3.2.1. 질의 경로 처리의 예

다음 경로식은 앞서 소개했던 질의문 Q에서의 경로식이다.

/chapter/section/section/title

chapter, section, title 의 시그너처를 OR연산하여, 질의 시그너처 QS = "11110101"를 만든다. QS는 1,2,3,4,6,8 번째 비트가 1로 세팅되어 있으므로, 시그니처 화일의 1,2,3,4,6,8 번째 행을 선택하여 AND 연산한다. 그 결과로 비트 맵 "000101"을 얻는다. 비트 맵의 4,6 번째 비트가 1로 세팅되어 있으므로, 경로 테이블에서 4,6번째 튜플을 후보로 선택한다. 즉, 후보 경로 집합은 표6과 같다.

Path_ID	Path_Str
4	/chapter/section/title
6	/chapter/section/section/title

표 6 후보 경로 집합

단지 두 번의 비트 연산을 통해 두 개의 후보 경로를 선택하였다. 후보 집합에서 실제로 질의를 만족하는 경로만 선택해야 하는데, 질의 경로식은 일종의 정규식이므로 질의 경로식에 대한 NFA를 작성하여, 후보 집합의 각 튜플을 검증한다.

검증을 통해 질의 경로식을 만족하는 6번 경로 튜플만을 선택하게 되고, 이를 노드 테이블과 조인하여 질의 경로식 처리를 마치게 된다.

4. 결론

대용량의 XML문서에서 원하는 데이터를 검색하는데 있어서 질의의 경로식 처리는 가장 비용이 많이 드는 부분이다. 기존의 방법은 많은 조인 연산을 통해 이를 처리하여 비용이 많이 들었으나, 본 연구에서는 노드의 시그니처 정보를 통해 조인 연산 없이 경로식을 처리하여 질의 수행 성능을 높일 수 있다. 또한 개념이 간단하며, 기존 관계형 데이터베이스에 쉽게 적용시킬 수 있다는 장점이 있다.

참고문헌

[1] C. Faloutsos, R. Chan, "*Fast Text Access Methods for Optical and Large Magnetic Disks: Designs and Performance Comparison*," VLDB, pp.280-293,1988

[2] Q. Li and B. Moon, "*Indexing and querying XML data for regular path expressions*," VLDB, pp.361-370, 2001

[3] Sangwon Park, et al., "*A New Query Processing Technique for XML Based on Signature*," DASFAA, pp.22-29, April 2001

[4] J. Shanmugasundaram, et al., "*Relational databases for querying XML documents: Limitations and opportunities*," VLDB, pp.302-314, 1999

[5] Chun Zhang, et al., "*On Supporting Containment Queries in RDBMS*," SIGMOD, 2001

[6] http://www.w3.org/XML/Query#specs, August 2002.