XML 문서에서의 엘리먼트 정보를 이용한 스키마 추출방법

김 성 림† · 윤 용 익††

요 약

인터넷상에서 데이터를 표현하고 교환하는 새로운 표준으로 등장하는 XML 문서는 정해진 스키마를 가지고 있지 않다. XML 문서를 기존의 SQL이나 OQL에 바로 적용하기에는 부적합하여 이러한 XML 문서에 대해 스키마를 추출하는 방법과 질의어에 대한 연구가 활발히 진행되고 있다. 본 논문에서는 XML문서에 대해 엘리먼트 정보를 이용하여 스키마를 추출하고, 추출된 스키마를 바탕으로 데이터 빈도수에 따라 새로운 여러 단계의 스키마를 추출하는 방법을 제시하고 실험한다.

A Schema Extraction Method using Elements Information in XML Documents

Sungrim Kim† · Yoon Yong Ik††

ABSTRACT

XML documents, which are becoming new standard for expressing and exchanging data in the Internet, don't have defined schema. It is not adequate to directly apply XML documents to the existing SQL or OQL. Research on how to extract Schema for XML documents and query language is going on actively. For users' query, the results could be too many or too less. It is important to give the users adequate results. This paper suggests the way to extract many levelized schema according to the frequency of element occurrence in XML documents. The Schema can be reduced or extended to correspond to the users' query more flexibly.

키워드 : XML, 문서(document), 스키마 추출(schema extraction), 빈도(frequency)

1. 서 론

XML(eXtended Markup Language)는 인터넷상에서 데이터를 표현하고 교환하는 새로운 표준으로 등장하고 있다[1]. HTML과 마찬가지로 XML은 SGML의 부분집합이지만 HTML 태그가 데이터 아이템의 표현에 중점을 둔 것이라면, XML 태그는 데이터 자체를 기술한다[5, 8, 11].

XML의 데이터 모델은 구조상 기존의 데이터베이스와 많은 차이점이 있고, 또한 기존 SQL이나 OQL을 바로 적용하기에 부적합하다. 따라서 이러한 XML 문서들에 대해 스키마를 추출하는 방법과 질의어에 대한 연구가 활발히 진행되고 있다[5, 8, 9, 11].

본 논문에서는 XML 문서의 엘리먼트 정보를 이용하여 스키마를 추출하고, 추출된 스키마를 바탕으로 데이터 빈도 수에 따라 새로운 여러 단계의 스키마를 추출하는 방법을 제시하고 이를 실험하여 결과를 분석해 본다. 사용자 질의에 대해 질의 수행 결과가 너무 적거나 많을 때 여러 단계의 스키마에 적용해서 질의를 수행함으로써 사용자의 요구를 효율적으로 반영할 수 있게 한다.

본 논문의 구성은 다음과 같다. 2장에서는 스키마 추출에 대한 관련 연구를 살펴보고, 3장에서는 본 논문의 배경이 되는 이론들을 서술한다. 4장에서는 본 논문에서 제안하고자 하는 스키마를 추출하는 알고리즘과 예제를 보이고, 5장에서 제안한 방법을 실험하여 그 결과를 분석해 보고, 6장에서 결론을 맺는다.

2. 관련 연구

본 장에서는 발생 빈도에 따라 스키마를 추출하는 기존의 방법에 대해서 살펴본 뒤, 기존 연구의 제한점을 설명한다.

2.1 트리 표현식의 발생 빈도수에 따른 스키마 추출

트리 표현식들의 발생 빈도에 따라 최대의 트리 표현식으

† 준 회 원 : 동덕여자대학교 정보학부 컴퓨터학전공 강의전임교수
†† 종신회원 : 숙명여자대학교 정보학부 멀티미디어학과 교수
논문접수 : 2001년 8월 5일, 심사완료 : 2002년 2월 28일

로 공통적인 스키마를 추출하는 방법이 있다[12, 13]. 반구조적(semi-structured) 데이터같이 정형화되지 않은 데이터를 트리 표현식으로 표현하고, 이 들 중에서 사용자 정의 지지도보다 높고, 많은 정보를 표현하는 트리 표현식을 추출되는 스키마로 정의하는 방법이다.

자주 발생되는 비슷한 질의에 대해서 트리 표현식을 만들고, 이를 바탕으로 스키마를 추출함으로써 유사한 질의에 대해 효율적으로 실행할 수 있다는 장점이 있지만 문서 전체에 대한 스키마를 찾기 힘들다는 제약이 있다.

2.2 발생 빈도 패턴 트리

발생 빈도 패턴을 찾는 방법은 트랜잭션 데이터베이스, 시계열 데이터베이스등 많은 데이터베이스 분야에서 연구되어 왔다. 여러 방법 중에서 발생 빈도 패턴 트리(*FP-tree* : Frequent Pattern Tree)를 구축하여 최대 패턴을 구하는 방법이 제시되었다[3]. *FP-tree*에서는 발생 빈도 패턴에 대한 정보를 저장하고, 이를 이용하여 조건 *FP-tree*를 형성함으로써 빈도 패턴을 찾아내는데 보다 효율성을 증가시켰다.

사용자 정의 발생 빈도 수를 바탕으로 스키마가 다양하게 추출될 수 있다는 장점은 있지만 전체 스키마보다는 특정 패턴에 대한 스키마가 추출이 되는 제약과 특정 패턴과 발생 빈도 수에 대한 사용자 정의 값에 대해 매번 스키마 추출 단계를 반복해야 한다는 제약점이 있다.

2.3 Lore 시스템의 DataGuide

Lore는 스탠포드대학에서 개발한 XML을 위한 데이터베이스 관리 시스템이다[2, 6]. Lore는 미리 정의된 스키마를 가지고 있지 않기 때문에 태그나 애트리뷰트 패턴이 없는 경우에 사용자가 의미있는 질의를 만들기는 어렵다. 또한 질의 엔진도 질의를 효율적으로 수행하기 위해서는 데이터베이스의 구조를 어느 정도 이해하고 있어야 한다. 이러한 기능을 위해 Lore에서는 DataGuide를 제공한다.

DataGuide는 XML 데이터베이스에 대해 정확하고, 동적으로 정리된 구조를 표현해줌으로써 데이터베이스 스키마나 DTD 역할을 수행하게 된다. 사용자는 DataGuide를 통해 데이터베이스의 전체적인 구조를 파악하여 질의를 만들 수 있게 된다.

DataGuide에서는 공통적인 구조만으로 구성된 최소 경계 스키마와 모든 구조 정보로 구성된 최대 경계 스키마가 있다. 최대 경계 스키마의 경우는 모든 구조 정보를 가지고 있어 스키마의 범위가 최대가 되고, 따라서 전체 문서에 대해 질의 수행을 해야 하는 제약점이 있다. 그리고 최소 경계 스키마의 경우는 공통적인 구조로만 구성되어 질의 범위를 최소화 할 수 있다. 최소 경계 스키마와 최대 경계 스키마 사이에 다른 스키마를 추출할 수 있다면 질의 검색 범위를 축소, 확장 가능하게 된다.

3. 모델링

본 장에서는 본 논문에서 제안하는 스키마 추출방법에 필요한 기본 개념을 살펴보고, 몇 가지 정의를 설명한다.

3.1 Edge Labeled Graph

XML 문서를 반구조적 데이터처럼 방향성있는 edge-labeled graph로 표현할 수 있다. Edge-labeled graph에서 엘리먼트는 객체(노드)로 표현되고, 각 객체는 &01같은 객체 식별자 *oid* (object identifier)를 갖고, 단순 객체 또는 복합 객체의 형태를 갖는다. Edge-labeled graph에서는 객체들간에 간선이 존재하고, 각 간선마다 엘리먼트 이름으로 레이블이 있고, 서브 엘리먼트를 표현하는 방향성을 갖는다[2].

3.2 스키마 추출을 위한 그래프

스키마 추출을 위하여 두 개의 그래프를 다음과 같이 정의한다.

정의 1 : *데이터 그래프(Data Graph)*

XML 문서의 모든 데이터가 표현되는 edge labeled directed graph를 '*데이터 그래프(Data Graph)*'라고 정의한다. 루트 노드로부터 하위노드로 방향성 있는 간선이 있고, 간선의 레이블은 엘리먼트 이름이 된다. 그리고 각 노드는 *oid*를 갖고, 노드는 단순 객체 또는 복합 객체의 형태이다.

정의 2 : *스키마 그래프(Schema Graph)*

XML 문서에 대한 데이터 그래프에서 깊이 우선 탐색 기법을 바탕으로 모든 경로가 단 한번만 표현될 수 있도록 만들어진 그래프를 '*스키마 그래프(Schema Graph)*'라고 정의한다. Lore 시스템[6]의 DataGuide[2]처럼 스키마 그래프에서는 모든 레이블 경로가 유일하고(*concise*), XML 문서에 있는 모든 데이터는 표현되어야 하고(*accuracy*), 각 노드의 구성이 어떻게 되어있는지(*convenience*) 알 수 있도록 한다.

데이터 그래프는 XML 문서의 구성을 쉽게 파악할 수 있게 하고, XML문서에 있는 모든 요소가 표현되어 중복적으로 나타나는 경우도 있다. XML 문서의 요소가 단 한번만 표현되는 스키마 그래프는 스키마 추출을 위해 기본적으로 필요한 그래프이다. 이 스키마 그래프를 레이블 경로의 빈도수에 따라 여러 개로 추출 가능하게 함으로써 사용자 질의에 보다 효율적으로 처리할 수 있도록 하였는데 이를 위해서는 다음 절에서 설명하는 레이블 경로 인덱싱을 사용한다.

3.3 비트맵 인덱싱을 이용한 레이블 경로 인덱싱

비트맵 인덱싱의 기본 개념은 애트리뷰트가 어떤 특별한 값을 갖고 있느냐 없느냐를 0/1의 비트로 표현하는 것이다[14]. 비트맵 인덱싱의 장점은 bitwise-AND, OR, NOT 연산이 하드웨어적으로 가능함으로써 수행속도가 빨라질 수 있다는 것이다. 이러한 장점으로 의사결정 시스템, 데이터웨어하우징에서 많이 사용되고 있다. 그리고 적은 카디날리티를 갖는 경우 적은 공간을 차지함으로써 효율적이 될 수 있다[4, 7, 10].

본 논문에서는 아래와 같이 정의한 XML 문서에서의 레이블 경로를 비트맵 인덱싱하여 보다 유동적인 스키마 그래프를 생성한다.

정의 3 : *레이블 경로(Label Path)*

데이터 그래프 혹은 스키마 그래프에서 한 노드에서 어떤 하위 노드로의 경로를 '*레이블 경로*' 라고 정의한다. 그래프에서 노드와 노드사이에는 간선이 존재하고, 간선은 엘리먼트 이름으로 레이블이 존재한다. 그리고 루트 노드에서 리프 노드까지의 경로에서 나타나는 중간 노드는 . (dot)을 사용하여 표현한다.

레이블 경로는 스키마 그래프에서 루트 노드에서 리프 노드까지 깊이 우선 탐색 기법을 이용하여 단 하나씩 구하게 된다. 이러한 비트맵 인덱스 기법을 스키마 그래프에서 레이블 경로에 적용하였다[15]. 스키마 그래프에서 구해진 레이블 경로를 각 XML 문서에 적용하여 레이블 경로의 존재 여부를 비트로 표현하게 된다. 즉, 레이블 경로가 해당 XML문서에 존재하면 1의 값을, 존재하지 않으면 0의 값을 갖는다. 모든 XML 문서의 비트맵 스트링에 대해 bitwise-OR 연산을 수행하게 되면 모든 레이블 경로를 포함하게 되는 스키마 그래프가 생성되는데 이는 DataGuide의 최대 경계 스키마에 해당될 수 있다. 만약 bitwise-AND를 하게 되면 모든 XML 문서에 공통적으로만 존재하는 레이블 경로로만 구성된 스키마 그래프가 생성되는 데 이는 DataGuide의 최소 경계 스키마에 해당될 수 있다.

4. 스키마 추출

본 장에서는 3장에서 설명한 기본 개념을 바탕으로 영화에 대한 정보를 표현하는 DTD와 XML문서를 가지고 예를 들어 설명한다.

4.1 예제 DTD와 XML 문서

영화 정보에 대한 내용(http : //www.imdb.com/)으로 <표 4.1>과 같은 DTD와 이를 바탕으로 생성된 XML문서((그림 4.1))를 가정한다[16].

〈표 4.1〉 DTD 예제 : movie.dtd

```
<!ELEMENT movie (title, year, director+, writer+, genre+, cast+,
                 language*, country*, color*, keywords*) >
<!ELEMENT title (#PCDATA) >
<!ELEMENT year (#PCDATA) >
<!ELEMENT director ((lastname, firstname) | fullname) >
<!ELEMENT writer ((lastname, firstname) | fullname) >
<!ELEMENT lastname (#PCDATA) >
<!ELEMENT firstname (#PCDATA) >
<!ELEMENT fullname (#PCDATA) >
<!ELEMENT genre (#PCDATA) >
<!ELEMENT cast (name,role, (award, category)*, spouse* >
<!ELEMENT name (#PCDATA) >
<!ELEMENT role (#PCDATA) >
<!ELEMENT award (#PCDATA) >
<!ELEMENT category (#PCDATA) >
<!ELEMENT spouse (name, occupation)* >
<!ELEMENT occupation (#PCDATA) >
<!ELEMENT language (#PCDATA) >
<!ELEMENT country (#PCDATA) >
<!ELEMENT color (#PCDATA) >
<!ELEMENT keywords (#PCDATA) >
```

• XML 문서 : d1.xml

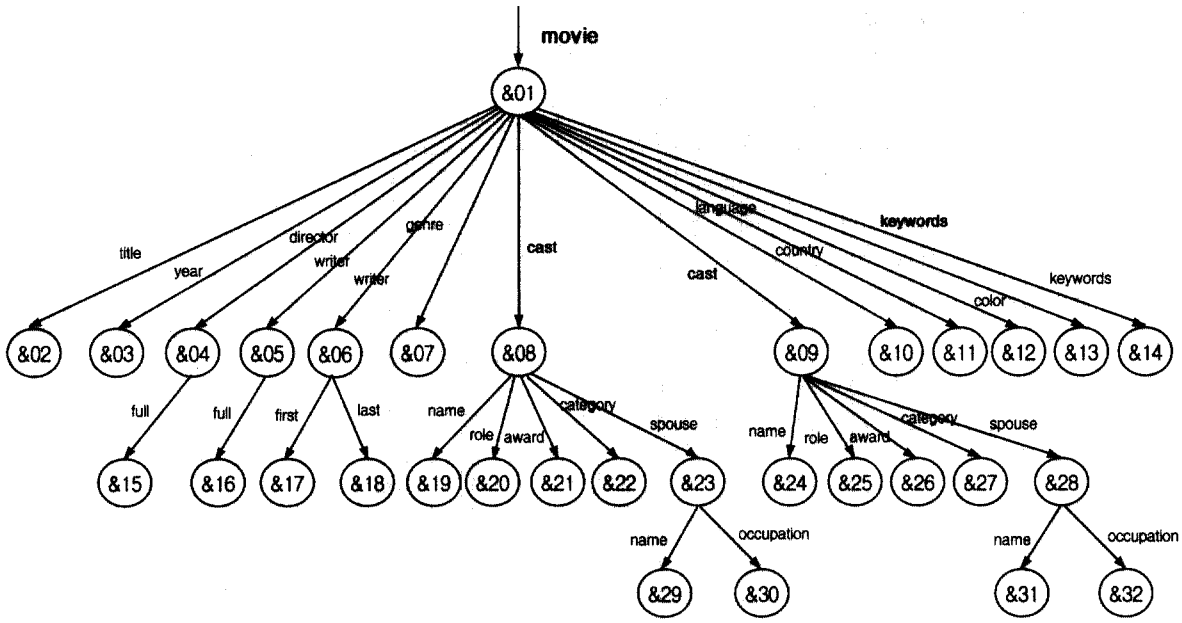
```
<movie>
 <title>Citizen Kane </title><year>1941</year><director>
 Orson Welles</director>
 <writer>Herman J. Mankiewicz </writer>
 <writer><firstname>Orson</firstname><lastname>Welles
 </lastname></writer>
 <genre>Drama</genre>
 <cast>
   <name>Orson Welles</name><role>Charles Foster Kane
   </role>
   <award>Oscar</award>
   <category>Best Writing, Original Screenplay</category>
   <spouse><name>Rita Hayworth </name>
           <occupation>divorced</occupation></spouse>
 </cast>
 <cast>
   <name>Dorothy Comingore</name><role>Susan Alexander
   Kane </role>
   <spouse><name>Richard Collins (I) </name><occupation>
   ?</occupation></spouse>
 </cast>
 <language>English</language><country>USA</country>
 <color>Black and White</color>
 <keywords>sled </keywords> <keywords>dying-words
 </keywords>
</movie>
```

(그림 4.1) XML 문서

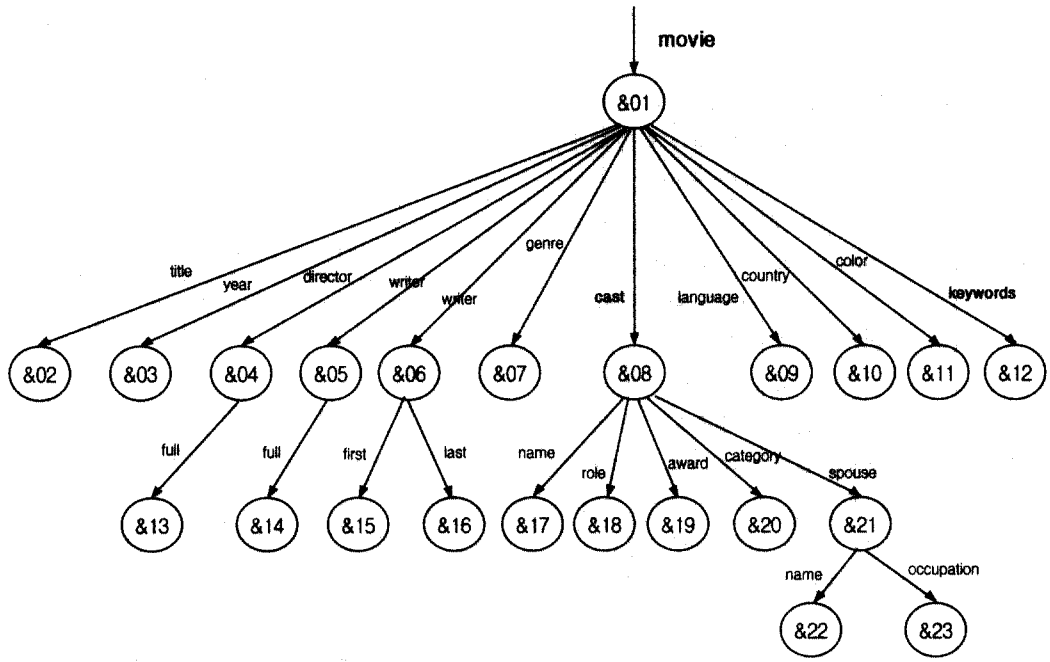
4.2 데이터 그래프와 스키마 그래프

(그림 4.1)의 예제 XML 문서를 데이터 그래프로 표현하면 (그림 4.2)와 같다. 3.2에서 정의한 바와 같이 (그림 4.2)의 데이터 그래프는 예제 XML문서에 있는 모든 엘리먼트가 표현되어 있다.

(그림 4.1)의 예제 XML 문서를 스키마 그래프로 표현하면 (그림 4.3)과 같다. 3.2에서 정의한 바와 같이 (그림 4.3)의 스키마 그래프는 모든 레이블 경로가 단 한번씩만



(그림 4.2) XML 문서에 대한 데이터 그래프



(그림 4.3) XML 문서에 대한 스키마 그래프

표현되게 된다. 데이터 그래프에서 깊이 우선 탐색 기법을 이용하여 모든 레이블 경로가 단 한번만 존재하게 하고, XML 문서의 엘리먼트는 반드시 나타날 수 있도록 하였다.

4.3 비트맵 인덱싱을 이용한 스키마 추출

XML문서를 바탕으로 3.2에서 정의한 스키마 그래프를 생성한 후 레이블 경로를 구하고, 각 XML문서에서의 레이블 경로의 존재여부를 비트 벡터로 표현한 후 그 빈도 수에 따라 다양한 스키마 그래프를 재생성할 수 있다.

4.3.1 레이블 경로

스키마 그래프에 대해 레이블 경로는 루트 노드부터 리프 노드까지 깊이 우선 탐색 기법을 바탕으로 중간노드의 레이블이 레이블 경로에 추가되는데 그 알고리즘은 (알고리즘 1)과 같다.

```
MakeLabelPath (treeNode u)
 begin
       path = u.label
       while (u isNot leafNode)
         begin
              v = u.childNode
              path = path + v.label
             u = v
         end
 end
```

(알고리즘 1) 레이블 경로 구하기

```
CalcBitVector(document d)
begin
        while (pi isNot EOF)
              if (pi in d)
                    d(pi) = 1
              else
                    d(pi) = 0
              endif
end
```

(알고리즘 2) 레이블 경로 인덱싱

비트맵 인덱스 기법을 (그림 4.3)의 스키마 그래프의 레이블 경로와 비교하여 각 XML 문서에서의 레이블 경로 존재 여부에 따라 0/1의 값을 갖는다. 이를 구하는 알고리즘은 (알고리즘 2)와 같다.

4.3.2 레이블 경로 빈도 수를 이용한 스키마 추출

모든 XML 문서에 대한 비트맵 인덱스에 대해 Bitwise-OR 연산을 수행하면 모든 레이블 경로를 포함되는 스키마 그래프가 생성되고, 이 그래프는 데이터 그래프에서 생성되는 스키마 그래프와 동일하다((알고리즘 3)).

```
bitwiseORLabelPath()
 begin
    for each i in path P
        new_path(i) = bitwise_OR(ΣdjPi)
end
```

(알고리즘 3) 레이블 경로에 대한 bitwise-OR 연산

각 레이블 경로의 발생 빈도 수를 이용하여 여러 단계의 스키마 그래프를 생성한다. 모든 XML 문서에서 각 레이블 경로에 대한 비트맵을 bitwise-OR 연산을 수행하면 모든 레이블 경로가 표현되는 최대 범위의 스키마 그래프를 구할 수 있고, bitwise-AND 연산을 수행하면 모든 XML 문서에서 공통적으로만 표현되는 레이블 경로로만 구성된 최소 범위의 스키마 그래프가 생성되어 질의 범위를 축소할 수 있다. 또한 레이블 경로 빈도 수를 조절하여 스키마 그래프를 생성함으로써 그 질의 범위를 유동적으로 할 수 있다. 여러 단계의 스키마를 추출하기 위해 레이블 경로의 발생 빈도 수를 (알고리즘 4)에 의해 계산한다.

```
//di : XML 문서
countLabelPath()
  begin
     for each i in path P
         Freq(i) = count(Σdj)
 end
```

(알고리즘 4) 레이블 경로 빈도수 계산

(알고리즘 4)를 바탕으로 계산되어진 레이블 경로의 빈도 수를 어떤 임계치를 부여하여 여러 단계의 스키마를 추출할 수가 있다. 주어진 임계치보다 빈도수가 적을 레이블 경로는 0의 값을, 빈도수가 많은 레이블 경로는 1의 값을 갖는다. 1의 값을 갖는 레이블 경로로만 이루어진 스키마 그래프를 생성하는 것이다. 이렇게 어떤 임계치를 기준으로 스키마를 조정할 수 있다면 사용자 질의에 대해 질의 수행 범위를 조정하여 보다 효율적인 질의 처리가 가능해질 것이다. 만약 임계치가 1이라면 bitwise-OR 연산 결과와 마찬가지로 모든 레이블 경로가 표현되는 스키마 그래프가 나올 것이고, 임계치가 3이라면 발생 빈도수가 3이상인 레이블 경로로만 이루어진 스키마 그래프가 만들어질 것이다.

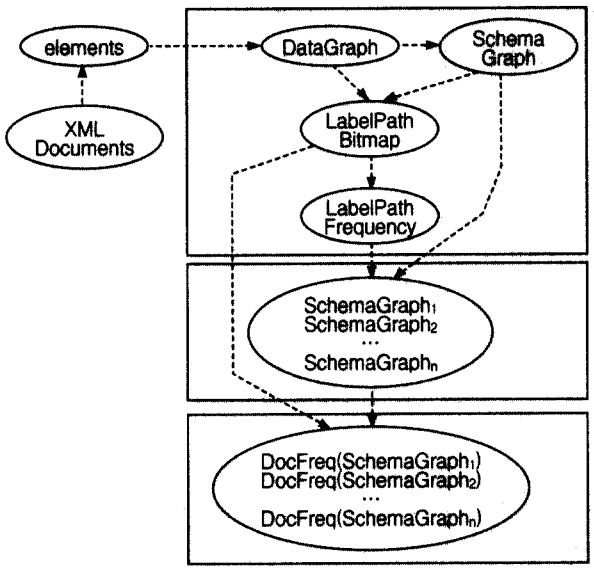
5. 실 험

5.1 실험 환경

본 논문에서 제시한 레이블 경로의 발생 빈도 수에 따른 스키마 추출 방법의 실험 환경은 다음과 같다. 운영 체제는 Windows2000이고, 데이터베이스는 Oracle9i 를 사용하였다. 구현 언어는 JDK 1.3.1과 JSP를 사용하였고, 오라클과의 연동을 위하여 Oracle JDBC thin driver 를 사용하였다. 웹서버는 IIS 5.0, JSP 엔진으로는 resin 2.0.1을 사용하고, XML 파서로는 Oracle Parser (버전 2.0.1.0)를 사용하였다.

5.2 실험 모델

본 논문에서 사용된 XML 문서는 영화에 관한 정보가 있는 http : //www.imdb.com를 참조로 하였다[16]. Top250 films 중에서 100위까지의 데이터 중에서 멀티미디어 데이터를 제외한 텍스트 데이터를 중심으로 XML 문서를 작성하여 실험하였다. 그리고 본 논문에서 제안하는 스키마를 추출하는 과정은 (그림 5.1)과 같다.



(그림 5.1) 스키마 추출과정

5.3 실험 결과 : 레이블 경로의 발생 빈도 수

레이블 경로의 발생 빈도 수에 대한 실험 결과는 <표 5.1>과 같다. <표 5.1>의 결과를 보면 director_firmstname과 director_lastname이 가장 최소 발생 빈도수임을 알 수 있고, cast_name, cast_role, color, country, genre, keywords, title, year는 100개로 모든 실험 문서에 나타나는 레이블 경로임을 알 수 있다.

〈표 5.1〉 레이블 경로 발생 빈도수

Label Path	발생 빈도수
cast_awards_award	56
cast_awards_category	56
cast_name	100
cast_role	100
cast_spouse_name	89
cast_spouse_occupation	89
color	100
country	100
director	56
director_firstname	49
director_lastname	49
genre	100
keywords	100
language	98
title	100
writer	78
writer_firstname	55
writer_lastname	55
year	100

5.4 실험 결과 : 임계치에 따라 생성되는 스키마

임계치를 0.1부터 1.0까지 0.1씩 증가시키면서 생성되는 스키마의 엘리먼트의 개수와 생성되는 스키마의 엘리먼트를 포함하는 XML 문서의 개수를 추출해 보았다. 그 결과는 <표 5.2>와 같다.

〈표 5.2〉 임계치에 따라 생성되는 스키마

임 계 치	엘리먼트 개수	XML 문서 개수
0.1	19	2
0.2	19	2
0.3	19	2
0.4	19	2
0.5	17	10
0.6	12	68
0.7	12	68
0.8	11	87
0.9	9	98
1.0	8	100

5.5 실험 결과 분석

실험 결과를 보면 임계치가 작을수록 추출되는 스키마은 많은 엘리먼트 정보를 포함하여 범위는 커지지만 이 새로운 스키마를 만족하는 XML 문서의 개수는 적어진다. 예를 들어, 임계치가 1.0의 경우는 모든 실험 XML 문서에 공통적으로 나타나는 엘리먼트만을 포함하여 8개의 엘리먼트를 표현하지만 그 범위는 100개로 최대가 된다. 하지만 임계치가 0.6인 경우는 12개의 엘리먼트를 포함하는 스키마를 생성하지만 이를 만족하는 XML 문서의 개수는 68개가 된다. 사용자 질의에 대해 임계치에 따란 다르게 생성되는 스키마에 적용한다면 질의 처리전에 그 범위를 확장, 축소하여 질의 처리에 효율성을 가져올 수 있음을 알 수 있다.

예를 들어 시나리오 작가(writer)에 대한 질의를 수행하기 위해서는 임계치가 1.0인 경우에 스키마에 대해서는 적용할 수가 없다. 임계치가 1.0인 스키마는 XML 문서의 범위가 100개로 최대이지만 writer에 대한 정보를 포함하지 않기 때문이다. 이런 경우는 임계치가 0.7 이하인 경우의 스키마가 적용 가능한 스키마가 될 것이다. writer에 대한 정보를 포함하는 스키마는 0.1부터 0.7까지의 스키마에 모두 해당되지만 그 중에서 가장 많은 XML 문서를 포함하는 0.6이나 0.7인 경우의 스키마가 질의에 가장 적합한 스키마라고 할 수 있다.

6. 결 론

XML이 인터넷상에서 데이터를 표현하고 교환하는 새로운 표준으로 등장하고 있다. XML은 미리 정의된 스키마가 없고, 문서 자체에 데이터와 데이터 구조를 갖고 있기 때문에 기존의 SQL이나 OQL을 바로 적용하기가 어렵다. 따라서 이러한 XML에 대해 새로운 질의어와 질의 처리를 위한 스키마 추출에 대한 많은 연구가 이루어지고 있다.

본 논문에서는 XML 문서의 엘리먼트 정보를 비트맵 인덱싱을 이용하여 표현하고 그 발생빈도수에 따라 스키마를 추출하는 방법을 제안하였다. 비트맵 인덱싱을 이용한 스키마 추출방법은 일단 같은 DTD를 갖는 XML 문서들에 대해 스키마 그래프를 생성하여 XML문서에 있는 모든 엘리먼트의 정보가 단 한번만 표현될 수 있도록 한다. 그리고 스키마 그래프를 바탕으로 XML 문서를 비트 벡터로 표현하고, XML 문서에서 레이블 경로의 발생 빈도 수를 계산하여 어떤 임계치에 따라 여러 단계의 스키마 추출을 가능하게 함으로써 사용자 질의에 대해 보다 효율적으로 처리할 수 있도록 하였다.

본 논문에서 제안하는 방법을 영화 관련 데이터를 바탕으로 XML문서를 생성하여 실험하였다. 실험 결과 임계치가 낮을수록 생성되는 스키마에는 많은 엘리먼트 정보를 포함하지만 이를 만족하는 XML 문서의 개수는 적고, 임계치가 높을수록 생성되는 스키마에 포함되는 엘리먼트 정보는 적지만 많은 XML 문서에 적용가능함을 알 수 있었다. 그리고 XML 문서를 구성하는 엘리먼트 정보와 그 발생 빈도를 파악함으로써 XML 문서의 전체적인 구조뿐만 아니라 빈도수에 따라 구조의 범위의 축소, 확대를 가능하게 함으로써 질의 범위를 조정할 수 있음을 알 수 있다. 또한 사용자 질의 패턴을 파악할 수 있게 하여 사용자마다 다른 스키마를 제공함으로써 사용자에게 적합한 스키마를 제공할 수 있다.

향후 연구과제로는 메타 데이터를 이용한 스키마 추출방법을 고려하여 데이터의 형태뿐만 아니라 의미적으로도 분석이 가능해 보다 효율적인 스키마를 추출하는 방법에 대한 연구가 필요하다.

참 고 문 헌

[1] Jon Bosak, "XML, Java, and the Future of the Web," http : //webreview.com/wr/pub/97/12/19/xml/index.html.

[2] Roy Goldman, Jennifer Widom, "DataGuides : Enabling Query Formulation and Optimization in Semistructured Databases," In Proceedings of VLDB, 1997.

[3] Jiawei Han, Jian Pei, Yiwen Yin, "Mining Frequent Patterns without Candidate Generation," Proceedings of the 2000 ACM SIGMOD on Management of data, pp.1-12, 2000.

[4] Theodore Johnson, "Performance Measurements of Compressed Bitmap Indices," VLDB, pp.278-289, 1999.

[5] Alon Levy, "More on Data Management for XML," University of Washington, May 9th, http : //www.cs.washington.edu/homes/alon/widom-response.html, 1999.

[6] J. McHugh, S. Abiteboul, R. Goldman, D. Quass, J. Widom, "Lore : A Database Management System for Semistructured Data," SIGMOD Record, 26(3), pp.54-66, September, 1997.

[7] Patrick O'Neil, "Improved Query Performance with Variant Indexes," Proceedings of ACM SIGMOD, pp.38-49, 1997.

[8] Jayavel Shanmugasundaran, Kristin Tufte, Gang He, Chun Zhang, David DeWit, Jeffrey Naughton, "Relational Databases for Querying XML Documents : Limitations and Opportunities," Proceedings of the 25th VLDB Conference, 1999.

[9] Dan Suciu, "Semistructured Data and XML," In Proceedings of International Conference on Foundation of Data Organization, 1998.

[10] M. C. Wu, A. P. Buchmann, "Encoded Bitmap Indexing for Data Warehouses," Proc. ICDE '98, pp.220-230.

[11] Jennifer Widom, "Data Management for XML," Working Document, initial draft appeared April 1999, Also IEEE Data Engineering Bulletin, Special Issue on XML, 22(3) : 44-52, September, 1999.

[12] Ke Wang, Huiqing Liu, "Schema Discovery from Semistructured Data," International Conference on Knowledge Discovery and Data Mining, pp.271-274, August, 1997.

[13] Ke Wang, Huiqing Liu, "Discovering Typical Structures of Documents : A Road Map Approach," The ACM SIGR conference on Research and Development in Information Retrieval, pp.146-154, August, 1998.

[14] Ming-Chuan Wu,, "Query optimization for selections using bitmaps," Proceedings of the 1999 ACM SIGMOD international conference on Management of data, pp.227-238.

[15] J. Yoon, S. Kim, "Schema Extraction for Multimedia XML Document Retrieval," in Proc. of International Database Symposium on Mobile, XML and Post-Relational Databases, Hong Kong, June, 2000. Also to appear in Journal of Applied Systems Studies, Cambridge International Science Publishing, Cambridge, UK, 2001.

[16] http : //us.imdb.com/top_250_films.

김 성 림

email : srkim@dongduk.ac.kr

1994년 숙명여자대학교 전산학과 졸업 (이학사)

1997년 숙명여자대학교 대학원 전산학과 졸업(이학 석사)

2002년 숙명여자대학교 대학원 컴퓨터학과 졸업(이학 박사)

2001년~현재 동덕여자대학교 정보학부 컴퓨터학전공 강의전임교수

관심분야 : XML, 멀티미디어 데이터베이스, 질의 처리

윤 용 익

email : yiyoon@sookmyung.ac.kr

1983년 동국대학교 통계학과 졸업(이학사)

1985년 한국과학기술원 전산학과 졸업 (공학석사)

1994년 한국과학기술원 전산학과 졸업 (공학박사)

1997년~현재 숙명여자대학교 정보학부 멀티미디어학과 교수

관심분야 : 정보통신, 멀티미디어 통신, 분산 시스템, 실시간 처리 시스템, 분산 미들웨어 시스템, 분산 데이터베이스 시스템, 실시간 OS/ DBMS