XML 문서의 다양한 구조 검색을 위한 효율적인 동적 색인 모델

(An Efficient Dynamic Indexing Model for Various Structure Retrievals of XML Documents)

신 승 호 † 손 충 범 †† 강 형 일 ††† 유 재 수 ††††
(Seung Ho Shin) (Chung Beon Son) (Hyuong Il Kang) (Jae Soo Yoo)

요 약 정보 표현의 기본 단위인 엘리먼트로 구성되는 XML 문서 내에서 동적으로 구조 변경이 이루어진다. 이때 XML 문서의 구조변경은 빠른 검색을 위해 기존의 색인 구조 정보의 변경 없이 효율적으로 처리되어야 한다. 이를 위해 본 논문에서는 XML 문서의 구조 변경 시 기존의 색인 구조에 효율적으로 수용될 수 있는 동적 색인 모델을 제안한다. 제안하는 동적 색인 모델은 다양한 구조 검색을 지원하기 위한 구조 정보 표현 방법과 효율적인 구조 검색을 지원하기 위한 동적 색인 구조로 구성된다. 제안하는 색인 기법이 기존의 동적 색인을 지원하는 기법보다 내용 색인, 구조 색인, 애트리뷰트 색인 측면에서 우수함을 성능 평가를 통해 보인다.

키워드 : XML, 동적 색인모델, 구조 검색

Abstract XML documents consist of elements that are basic units of information. When the structure of XML documents is changed dynamically, we need to update structure information efficiently without changing the information of the index structure for fast retrieval. In this paper, we propose a dynamic indexing model scheme that updates the index structure in real time as the structure of XML documents is changed by insertion and deletion of elements. Our dynamic indexing model consists of a structure information representation method and a dynamic index structure. The structure information representation method supports various types of structure retrievals. Our dynamic index structure processes various structural queries efficiently. We show through various experiments that our method outperforms existing ones in processing various types of queries such as content based queries, structural queries and hybrid queries.

Key words : XML, Dynamic Indexing Model, Structural Query

1. 서 론

XML(eXtensible Markup Language)은 1996년 W3C (World Wide Web Consortium)에서 제안한 것으로서, 웹 상에서 구조화된 문서를 전송 가능하도록 설계된 표준화된 텍스트 형식이다. 이는 인터넷에서 기존에 사용하던 HTML(HyperText Markup Language)의 한계를 극복하고 SGML(Standard Generalized Markup Language)의 복잡함을 해결하는 방안으로써 HTML에 사용자가 새로운 태그를 정의할 수 있는 기능이 추가되었다. 또한, XML은 SGML의 실용적인 기능만을 모은 부분집합이라 할 수 있으며, 인터넷상에서 뿐만 아니라 전자 출판, 의학, 경영, 법률, 판매 자동화, 디지털 도서관, 전자상거래, CSCW(Computer Supported Cooperative Work), EDI (Electronic Data Interchange)등 매우 다양한 분야에서 XML을 활용하고 있다. 이러한 분야에서 대용량의 XML 문서에 대해 효율적인 검색을 지원하는 시스템의 필요성이 대두되었고, 지금까지 많은 연구가 진행되어 오고 있다.

기존의 정보검색 시스템은 문서의 내용을 분석하여

· 이 논문은 2002년도 한국학술진흥재단의 지원에 의하여 연구되었음 (KRF-2002-003-D00281)
† 비 회 원 : 대우정보시스템(주)
shshin@disc.co.kr
†† 비 회 원 : 인하공업전문대학 정보통신과 교수
cbson@inhatc.ac.kr
††† 비 회 원 : 주성대학 멀티미디어정보통신공과 교수
khi69@jsc.ac.kr
†††† 종신회원 : 충북대학교 전기전자컴퓨터공학부 교수
컴퓨터 · 정보통신연구소 소장
yjs@cbucc.chungbuk.ac.kr
논문접수 : 2001년 10월 24일
심사완료 : 2003년 10월 9일

색인어를 추출하고 이를 바탕으로 색인을 구성한 다음 사용자 질의가 주어졌을 경우 질의에 사용된 단어와 색인어의 유사성을 계산하여 관련 문서를 제공한다. 하지만 이러한 기존의 정보검색 시스템은 문서의 구조정보를 표현하는데 적합한 XML 문서에 대한 정보검색에는 효율적이지 못하다. XML 문서를 위한 정보 검색 시스템을 설계하는데 있어서 가장 중요한 것은 첫째, XML 문서의 추상화된 논리적 정보 표현의 기본 단위는 엘리먼트이므로 기존의 문서단위 및 문서 내에 쓰인 내용 검색과 더불어 엘리먼트 단위의 검색 또한 이루어져야 한다. 둘째, 문서의 부분 삭제와 삽입이 발생할 경우 기존의 색인 구조에 커다란 영향을 주지 않으면서 효율적인 구조 정보를 표현 할 수 있는 방법과 이를 수용할 수 있는 동적 색인 방법이 필요하다.

이에 본 논문에서는 새로운 구조 표현 방법과 이 방법 기반의 동적 색인 모델을 제안한다. 제안하는 구조 정보 표현 방법은 특정 엘리먼트에 대한 직접적인 접근이 가능하고 엘리먼트 간의 관계를 구하기 위해 복잡한 연산이 필요하지 않도록 DTD(Document Type Definition)에 나타난 엘리먼트들과 XML 문서의 구조 정보를 사용해서 XML 문서를 효율적으로 관리하고 검색할 수 있다. 또한, 제안하는 동적 색인 모델은 XML 문서의 부분삭제 및 삽입을 처리할 때 전체 문서에 대한 구조 정보를 다시 추출하여 새로운 색인 구조를 생성하는 정적 색인 모델 구조를 지양하고 보다 빠른 검색과 저장 공간의 낭비를 최소화한다. 뿐만 아니라 제안하는 색인 모델은 기존 색인 구조를 수용할 수 있다.

본 논문의 구성은 다음과 같다. 2장에서 XML 문서에 대한 기존 연구들의 인덱스 구조에 관한 연구를 살펴본다. 3장에서 XML 문서 변경 시 효율적인 검색을 위한 구조정보 표현 및 전체적인 시스템 구성을 설명한다. 4장에서는 추출된 구조정보를 이용해 색인을 구성하고 색인된 인덱스 구조에서 문서의 부분 삽입 및 삭제가 발생할 경우 효율적으로 구축된 인덱스 구조에 적용됨을 보인다. 5장에서는 제안한 동적 색인 모델에서 구조 검색과 혼합 검색, 내용 검색 각각에 대하여 질의처리 및 검색 알고리즘을 기술한다. 6장에서는 제안한 색인 모델의 성능 평가를 기술하고, 마지막으로 7장에서는 결론과 향후 연구 방향을 제시한다.

2. 관련 연구

문서의 구조적 특성을 표현할 수 있는 XML 문서에 대한 효율적인 검색을 지원하기 위해서 구조 정보 기반을 토대로 한 색인 구성 방법에 초점을 맞추어 구조 정보 표현 및 색인 구성 연구가 진행되어져 왔다. 이에 대한 관련 연구는 다음과 같다. 먼저 R. Sacks-Davis가 제안한 K-ary 완전 트리 방법은 SGML 문서를 K-ary 완전 트리에 매핑하여 구조 검색을 지원하는 방법이다 [1-3]. 이 방법에서는 K-ary 완전 트리의 특성상 논리적 구조 관계인 부모 노드와 자식 노드의 관계를 연산식을 이용하여 손쉽게 알 수 있다는 장점을 가진다. 그러나 노드의 깊이가 깊어질수록 노드 변화가 커지고 사용하지 않는 노드 번호가 많아지므로 데이타의 양이 커지는 단점이 있다. 그리고 K-ary 완전 트리의 특성상 연산을 통해 구한 자식이나 형제의 노드 번호가 실제 존재하는 노드인지 확인해야 하는 단점뿐만 아니라 문서의 구조 변경이 발생하였을 경우 차수만큼의 노드를 모두 사용하여 할당할 수 있는 노드 번호가 없을 때 처음부터 다시 문서의 구조정보를 추출하여 색인을 구성해야 되는 단점을 지니고 있다.

[1]에서 제안한 엘리먼트 단위 구문 트리 구조는 SGML 파서를 통해 나온 구문 트리 정보를 여러 개의 엘리먼트 단위 레코드로 저장하는 방식이다. 문서 구조를 이루는 각각의 엘리먼트를 빠르게 접근하기 위해 B+ 트리를 사용한다. 그림 1은 SGML 파스 트리를 이용해 추출된 엘리먼트 정보를 엘리먼트 단위 구문 트리 구조로 변환하는 과정을 나타낸다. 엘리먼트 단위 구문 트리 구조의 경우 문서의 부분 삽입 및 삭제가 발생하게되면 기존의 색인 정보 변경없이 효율적으로 색인 구조에 반영할 수 있고, 문서의 특정 부분 구조를 접근할 때 B+ 트리를 통해 빠르게 접근할 수 있는 특징이 있다. 하지만 엘리먼트에 대한 B+트리를 유지해야 하기 때문에 부가 저장 공간이 필요하다는 단점이 있으며, 문서 구조의 순서 정보 검색을 지원하기 위한 인덱스 구조의 반복적인 노드 순회가 필요하다는 단점이 있다.

[4]에서 제안한 모델은 다양한 구조 정보 검색을 지원하기 위한 구조 정보 추출 방법과 이를 기반으로 하는 색인 구조로 구성된다. 이 방법의 구조정보를 표현하기

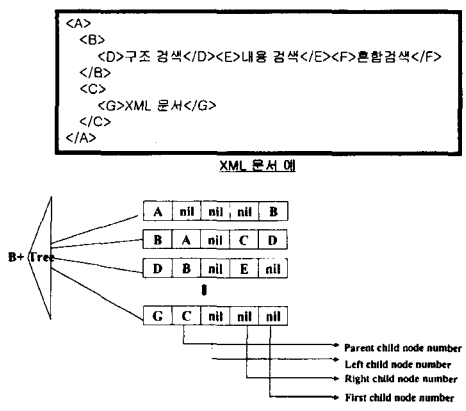

그림 1 엘리먼트 단위 구문 트리

위한 요소는 DTD에 나타나는 각각의 엘리먼트에 대해 유일한 값인 EID, 엘리먼트들 간의 계층정보를 나타내는 ETID, XML 문서에서 형제 엘리먼트의 순서정보인 SORD, 같은 타입의 엘리먼트 형제에 대한 순서정보인 SSORD, 그리고 특정 엘리먼트가 가지는 자식 수인 Ccount로 구성된다. ETID, SORD, SSORD, Ccount 색인 모델을 이용한 검색의 경우 특정 엘리먼트에 대한 직접적인 접근이 가능하며, 다양한 구조적 질의를 효과적으로 처리할 수 있다. 하지만, 문서의 구조 정보 트리가 깊을수록 포스팅 파일의 크기가 커져 검색 성능을 저하시키는 요인이 되며, 문서의 구조 변경이 발생할 경우 구조 정보를 다시 추출해야 되는 단점을 지니고 있다.

마지막으로 경로 조인을 이용한 방법[5]에서는 엘리먼트들에 대해 숫자 부여방법을 제시하여 경로 정보 없이 조상-후손 관계를 빠르게 결정할 수 있고 고정된 길의 경로나 알 수 없는 길이 경로에 적용할 수 있는 장점은 있으나, 단순한 경로를 포함하는 경우에는 두 번 이상의 색인 구조를 접근해야 하는 단점을 가지고 있다.

3. 제안하는 구조정보 표현 방법

XML 문서는 문서의 내용 정보뿐만 아니라 문서의 구조정보를 함께 표현한다. 따라서 문서의 논리적 구조를 효율적으로 표현하기 위한 구조 정보 표현 방법이 필요하다.

본 논문에서는 XML 문서의 논리적 구조 정보인 엘리먼트 및 애트리뷰트의 구조 정보, 그리고 내용 정보인 키워드를 통해 색인을 구성한다. 이러한 XML 문서 내에서 사용되는 구조 정보를 통해 색인을 구성할 경우 엘리먼트들은 문서 내에서 각각 논리적 구조를 형성하고 있기 때문에 각각의 색인 정보는 그림 2와 같이 문서구조와 상호 연관성을 형성하게 된다[1].

이로 인해 엘리먼트의 구조 정보 변경은 각각의 색인 정보의 변경을 유발시킨다. 따라서 XML 문서에 대하여 엘리먼트의 부분 삽입 및 삭제 그리고 내용 변경이 발생해도 색인 정보의 변경 없이, XML 문서 내의 엘리먼트에 대한 구조 정보와 데이타의 위치, 특성 값 등을 잘 표현할 수 있는 인덱스 구조가 설계되어야 한다. 따라서 본 논문에서는 이러한 문제점들을 해결할 수 있는 동적 색인 모델을 제안한다. 본 논문에서는 제안한 동적 색인 모델을 ESSIE(Extended SORD Structure Information Expression)라 표기한다.

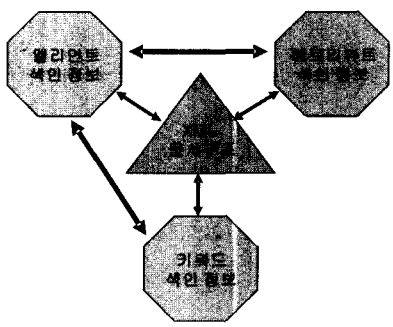

그림 2 XML 색인 정보들의 상호 연관성

3.1 DTD에 따른 엘리먼트 식별 ID 부여

본 논문에서 제안하는 ESSIE 구조 정보 표현을 위해서 XML 문서 내에 사용된 엘리먼트들에 대하여 유일한 Element ID(EID)를 부여해야 한다. 이러한 EID는 DTD에서 정의된 각 엘리먼트에 대하여 유일한 ID를 의미한다. 각 엘리먼트에 대해 ID를 부여하는 방식을 통해 DTD의 논리적 구조를 분석할 때 발생하는 엘리먼트간의 순환 문제와 ANY로 설정된 엘리먼트 처리 문제를 해결할 수 있다. EID는 2바이트를 사용하여 표현하는데 각 바이트는 '0' → '9' → 'A' → 'Z' → 'a' → 'z' 순으로 ASCII 코드의 순서를 따르는 62개의 문자를 사용하고 있다. 이를 통해 EID는 총 3844개의 엘리먼트를 표현할 수 있다. 만약 DTD에 정의된 엘리먼트의 개수가 3844개를 초과하는 경우에는 EID의 2바이트 표현을 3바이트 이상으로 확장해서 수용할 수 있다.

3.2 ESSIE 구조정보 표현 방법

본 논문에서 제안하는 동적 색인 모델(ESSIE)을 위한 구조 정보 표현은 특정 엘리먼트를 구별하면서 엘리먼트들 간의 연관성있는 구조 정보를 표현하기 위해 EID를 사용하여 XML 문서상에 나타나는 엘리먼트들에 대해 ETID(Element Type ID)를 부여한다. 이렇게 부여된 ETID는 XML 문서 내에 나타나는 엘리먼트들 간의 상속 관계를 나타내게 된다. 또한, XML 문서에서는 DTD에 나타난 발생 지시자에 의한 반복적인 엘리먼트의 사용이 가능하다. 이렇게 반복적으로 사용된 엘리먼트들은 ETID만으로는 구별이 불가능하다. 따라서 본 논문에서 이러한 문제를 해결하고, 문서의 동적 환경을 고려한 추가적인 구조 정보 표현을 제안한다. 추가되는 구조 정보는, 문서 내 사용된 엘리먼트들의 발생순서를 나타내는 SORD(SiblingORDer)와 동일한 타입의 형제 엘리먼트들 간의 순서정보를 나타내는 SOLST(Sibling ORDer List with Same Type), 같은 타입의 형제 노드에 대한 개수 정보를 나타내는 Scount이다. 또한, XML 문서 내에서 특정 엘리먼트가 갖는 자식의 정보를 나타내기 위한 COL(Child ORDer List)과 자식들의 범위 검색을 효율적으로 처리하기 위한 엘리먼트가 갖는 자식의 수 Ccount이다. 그림 3은 예를 들어 설명하기 위해 간단한 XML 문서를 보여주고 그림 4에서는 그림 3에 나타난 XML 문서의 구조정보를 본 논문에서

```
<?xml version="1.0" encoding="euc-kr" ?>
<!DOCTYPE paper SYSTEM "paper.dtd">
<paper status="public">
   <head>
            <title> XML 문서의 효율적인 검색을 위한 인덱싱 모델 </title>
            <author gender = "male">
                     <name>신승호</name>
                     <department>정보통신공학과</department>
            </author>
             <author gender = "female">
                     <name>이소라</name>
                     <department>정보통신공학과</department>
            </author>
            <abstract> 본 논문에서는 XML 문서의 정보 검색을 위한 동적 색인모델.. </abstract>
   </head>
   <body>
            <chapter> 1. 색인 모델 서 론
                     <section>관련연구  1
                              <para> 관련 연구 내용 </para>
                              <image src="fig1.gif"> 그림 1. XML 문서 예제</image>
                     </section>
                     <section> 관련 연구 2
                              <para> 관련 연구 사례 </para>
                     </section>
            </chapter>
            <chapter> 2. 색인 모델 본 론
                      <section> 2.1 색인 모델 구현
                              <para> 내용 검색 </para>
                              <para> 구조 검색 </para>
                      </section>
            </chapter>
   </body>
   <reference> 참고문헌 </reference>
</paper>
```

그림 3 XML 문서의 간단한 예

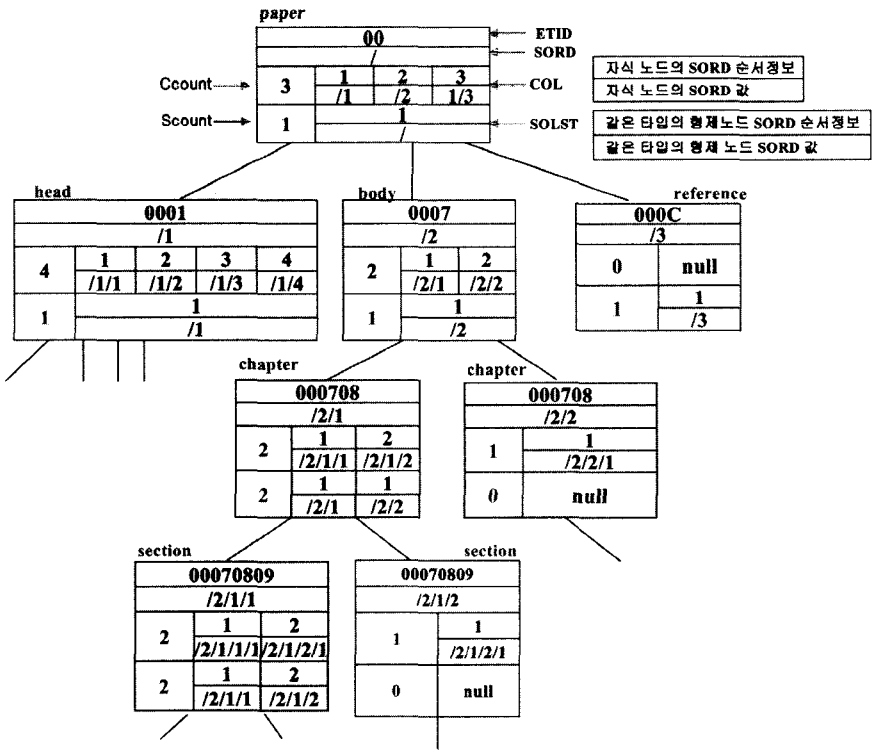

그림 4 XML 문서의 구조정보 표현

제안하는 ESSIE 구조 정보 표현 방법을 사용해서 트리의 형태로 표현한 일 부분의 예를 보여 준다.

예를 들어 head 엘리먼트에 대해서 제안한 구조 표현을 적용하는 경우에 대해 설명하면 다음과 같다. 먼저 head의 ETID는 부모 paper 엘리먼트의 ETID에 자기 EID인 01값을 결합한 값인 0001이 된다. SORD 값은 부모 엘리먼트의 SORD 값인 /에 자기 엘리먼트 순서 정보를 결합한 값인 /1이 된다. COL은 head 엘리먼트의 자식들에 대한 SORD값을 나타낸다. Ccount는 자식의 개수로 4가 된다. SOLST는 같은 엘리먼트 이름을 갖는 엘리먼트들의 순서로 현재 head는 자기 혼자이기 때문에 자기 자신의 SORD값인 /1값을 갖게 된다. 다른 엘리먼트들도 위와 같은 과정을 통해 해당하는 값을 얻을 수 있다.

3.2.1 엘리먼트들 간의 계층 관계

XML 문서 내에 사용된 엘리먼트들 간의 계층정보를 표현하기 위해 ETID를 사용하는데, 이 ETID는 XML 문서에 사용된 DTD를 분석하여 엘리먼트마다 유일하게 부여한 EID를 사용하여 만들게 된다. 즉, 부모의 ETID에 자신의 EID를 붙여서 자신의 ETID를 생성하게 되는데 부모가 없는 루트 엘리먼트인 경우는 자신의 EID를 ETID로 사용한다. 따라서, 문서 내에 사용된 각 엘리먼트에 부여된 ETID는 문서의 논리적 구조면에서 같은 부모를 갖는 동일 엘리먼트 타입에 부여되는 유일한 값이다. 이렇게 생성된 ETID와 해당 엘리먼트 이름과의 사상을 위해 ETID 매핑 테이블이 존재하게 되는데, 이 매핑 테이블을 참조하면 효과적인 구조 검색이 가능해 지고, 엘리먼트간의 상속 관계인 계층정보에서 조상과 자식이 어떤 타입의 엘리먼트인지 쉽게 알 수 있다. 아래의 그림 5는 엘리먼트 이름과 ETID의 매핑 테이블을 나타낸 것이다.

paper	00	body	0007
head	0001	chapter	000708
title	000102	section	00070809
author	000103	para	0007080908A
name	00010304	img	0007080908B
department	00010305	reference	000C
abstract	000106		

그림 5 ETID 매핑 테이블

ETID만을 가지고 XML 문서의 직접적인 접근 없이 엘리먼트들 간의 상속 관계를 나타내는 계층정보를 검색할 수 있다. 하지만, 같은 부모를 갖는 동일한 타입의 엘리먼트라면 ETID가 같아지게 된다. 이를 해결하기 위해서 다음과 같은 순서 정보 표현 방법을 제안한다.

3.2.2 엘리먼트들 간의 순서정보

XML 문서에서 사용되는 엘리먼트의 구조 정보 표현은 문서의 논리적 구조를 기술한 DTD의 표기에 따라 사용되어진다. DTD의 발생지시자의 종류에 따라 동일한 엘리먼트들이 반복적으로 나타날 수 있는데, 반복적으로 나타나는 엘리먼트들 간의 순서정보를 표현하기 위해 문서 내에 사용된 엘리먼트들을 유일하게 구별 할 수 있는 SORD와 동일한 부모를 갖으면서 동일한 타입의 순서 정보를 표현하기 위한 SOLST를 사용한다. SORD의 표현 방법은 UNIX 파일 시스템에서 디렉토리를 표현하는 방법을 사용하며 부모의 순서정보를 상속받는다. 그리고 엘리먼트들이 동일한 부모를 갖으면서 같은 타입의 형제 노드로 표현될 경우, 특정 엘리먼트의 접근만으로 순서 정보 검색을 할 수 있도록 SOLST을 유지한다. 이때 문서의 변경이 발생할 수 있는 동적 환경을 고려하고 저장 공간의 효율성을 위해, 최초 문서의 구조 정보 추출시 가장 먼저 나타나는 엘리먼트에 대해서만 SOLST을 유지하도록 한다. 그리고, 같은 타입의 형제 노드에 대한 범위 검색을 지원하기 위해 Scount(동일 부모를 갖으면서 같은 타입을 갖는 형제 노드의 개수)를 표기한다. 또한, 특정 엘리먼트의 자식에 대한 순서 정보와 자식에 대한 범위 검색을 지원하기 위해 COL과 Ccount(특정 엘리먼트의 자식의 개수)를 표기한다.

4. 동적 환경에 적합한 색인 모델 설계

4.1 시스템 구성도

제안하는 색인 모델은 크게 구조 정보 추출기와 색인 구성기로 구성된다. XML 문서의 구조 정보를 추출하기 위해 먼저 구조 정보 추출기의 EID 생성기를 통해 DTD로부터 엘리먼트의 타입을 추출하고, 엘리먼트의 이름과 엘리먼트 ID를 사상시키는 EID매핑 테이블을 생성한다. 그리고 그림 7에서와 같이, 실질적인 XML문서의 구조 정보를 추출하는 XML 구조 정보 추출기에서 XML 문서를 분석하면서 EID 매핑 테이블을 참조하여 ETID를 생성해 내고, 본 논문에서 제안하는 ESSIE의 각종 구조 정보를 추출한다. 이렇게 추출된 구조정보를 이용하여 색인 생성기는 효율적인 구조 정보 및 내용 검색을 위해 각각의 색인을 구성한다. 그림 6은 제안하는 색인 모델에 대한 전체적인 시스템 구조를 나타낸다.

XML 구조정보 추출기에 의해 추출되는 정보는 그림 6의 구조정보 추출 결과에서 보듯이 여러 필드를 갖는 텍스트 파일이다. 각 필드들은 각각의 의미를 갖는데 Tag name은 엘리먼트의 이름 혹은 애트리뷰트의 이름이 될 수 있고, DID(Document ID)는 문서를 구별하기 위해 부여된 고유 문서 번호이다. ETID, SORD, SOLST, Scount, COL, Ccount는 이미 앞에서 설명한 바와 같이, 본 논문에서 제안하는 동적 색인 모델을 구

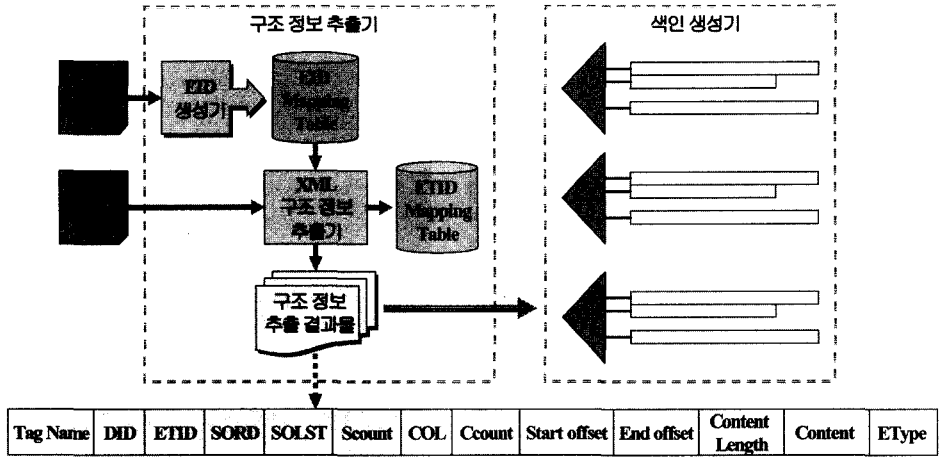

그림 6 제안하는 동적 색인 모델 구성도

성하기 위한 구조 정보이며, EType은 추출된 결과물이 엘리먼트인지 애트리뷰트인지를 나타내기 위한 구조 정보이다. Start offset와 End offset은 각각 XML 문서 파일에서 해당 엘리먼트의 시작위치와 끝위치를 나타낸다. Content는 해당 엘리먼트에 속하는 실제 내용이거나 애트리뷰트의 값이며, Content length는 Content 필드에 있는 값의 전체 길이를 나타낸다.

4.2 제안하는 동적 색인 모델

본 절에서는 XML 문서의 다양한 구조 정보 질의를 효율적으로 처리할 수 있으면서 문서의 부분 삭제 및 삽입, 내용 변경이 발생할 경우 이를 효율적으로 처리할 수 있는, 동적 환경을 고려한 구조 정보 표현 및 색인 구조를 설계한다. 색인 구성은 3장에서 추출한 ESSIE 구조 정보를 이용하여 구성하며, 색인은 내용 색인, 구조 색인, 애트리뷰트 색인 3가지로 구성되고 기존의 역파일 방식을 사용한다.

4.2.1 ESSIE 구조 정보 변경 연산

제안하는 ESSIE 구조 정보 표현은 다양한 XML문서의 구조 정보 검색 및 내용 검색을 지원할 수 있고, 문서의 변경이 발생할 수 있는 상황을 고려한 구조 정보 표현 방법이다. 따라서, ESSIE 구조 정보로 표현된 구조 정보가 변경될 경우 어떤 절차를 거쳐 이를 ESSIE 구조 정보 표현 방법에서 수용할 수 있는지를 XML 문서 내에 사용된 엘리먼트의 삽입 및 삭제의 경우를 그림 4의 구조 정보 표현 예를 사용하여 처리 과정을 보인다.

1) Element 삽입

최초 XML 문서의 구조 정보 추출시 SOLST는 문서에 사용된 DTD의 발생지시자(?, +, *)를 갖는 엘리먼트에서 동일 부모를 가지면서 같은 타입의 형제 노드들 중 제일 먼저 나타나는 노드에만 생성된다. 또한, Scount의 값은 동일한 타입의 형제노드 개수에 따라 달라질 수 있지만, 0은 자기 자신의 형제노드에 대한 정보를 다른 노드에서 유지하고 있음을 뜻하고, 1은 형제노드가 자기 자신 하나이면서 자신이 SOLST를 유지하고 있음을 나타낸다.

ESSIE의 구조 정보 표현에서 일단 문서의 구조 변경이 발생하게 되면 기존의 ETID, SORD, SSORD, Ccount 구조 정보 표현 방법에서의 SORD 순서 정보는 무의미해진다. 따라서 본 논문에서 제안하는 ESSIE 구조 정보 표현 방법에서 SORD의 의미를 부여하기 위해, 최초 문서의 구조 정보 추출시 엘리먼트에 부여된 SORD의 논리적 순서 정보가 위배되지 않도록 SORD의 확장 방법을 사용한다. SORD의 확장은 표기 방법에 따라 여러 가지가 있을 수 있으나, 본 논문에서는 float (1.0e-45)타입의 45 자리 수까지 확장하여 표기한다. 그림 7은 그림 4의 section 엘리먼트가 삽입 위치에 따라 삽입되는 과정을 보여 준다. 그림 8은 엘리먼트의 3가지 삽입 위치에 따라 처리되는 과정을 나타낸다.

2) Element 삭제

문서 내에 사용된 엘리먼트의 삭제가 발생할 경우에 제안한 ESSIE 구조 정보 표현의 특성을 고려한 다음 3가지의 경우로 나누어 처리하여야 한다. 첫 번째는 그림 9)에서 보는 바와 같이 형제 노드가 없는 특정 노드의 삭제(Delete A)이고, 두 번째는 형제 노드가 있으면서 자기 자신이 SOLST를 유지하지 않는 경우의 삭제(Delete B), 그리고 마지막으로 형제 노드를 가지고 있으면서 자기 자신이 SOLST를 유지하는 경우의 삭제이

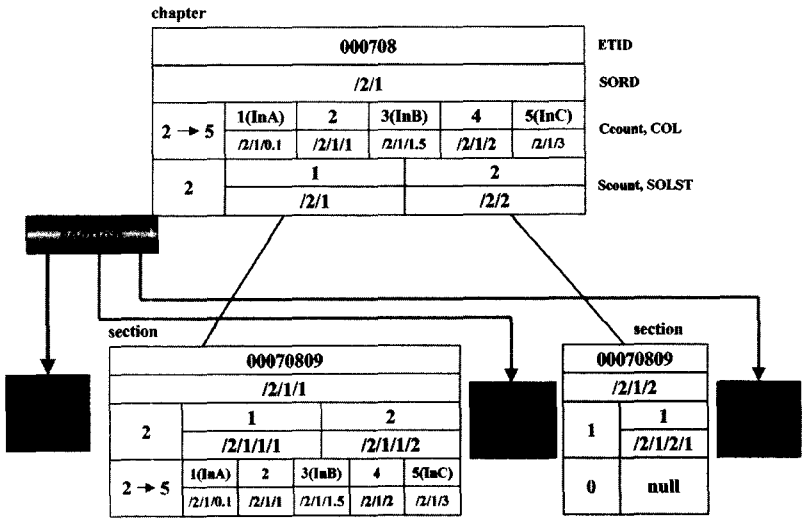

그림 7 XML 문서 내 엘리먼트의 삽입

A) 구축된 처음 노드 앞으로 삽입될 경우
1. 처음 노드에 0.1을 곱하여 SORD 값을 취함
2. 부모 노드의 COL에 추가 및 정렬, Ccount 증가
3. SOLST에 노드 추가 및 정렬, Scount 증가

B) 구축된 노드 사이로 삽입될 경우
1. 양쪽 노드의 마지막 SORD 값을 더해 나누기 2한 값을 SORD 값으로 취함
2. 부모 노드의 COL에 추가 및 정렬, Ccount 증가
3. Scount 값이 NULL이 아닌 형제 노드를 찾아 SOLST에 노드 추가 및 정렬, Scount 증가

C) 구축된 마지막 노드 뒤로 삽입될 경우
1. 마지막 노드와 차이를 1을 두어 SORD 값을 취함
2. 부모 노드의 COL에 추가 및 정렬, Ccount 증가
3. Scount 값이 NULL이 아닌 형제 노드를 찾아 SOLST에 노드 추가 및 정렬, Scount 증가

그림 8 3가지 경우의 엘리먼트 삽입

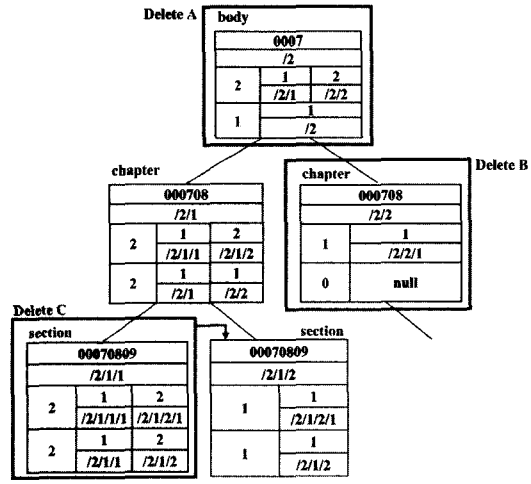

그림 9 XML 문서 내 엘리먼트의 삭제

다. 각각의 엘리먼트 삭제에 대한 구조 정보 변경 단계를 그림 3의 XML 문서를 그림 4의 ESSIE 표현 방법을 통해 표기한 XML 문서의 일부분을 예로 들어 설명하도록 한다. 그림 10은 엘리먼트의 3가지 삭제 위치에 따라 처리되는 과정을 나타낸다.

4.3 동적 색인 구성

4.3.1 내용 색인

내용 색인의 목적은 내용 검색에 대한 질의의 결과에 해당하는 DID와 SORD의 반환을 최종 목적으로 하며, 색인 구성시 불필요한 색인어를 줄이고, 검색 성능의 효율을 높이고자 PCDATA를 갖거나 CDATA를 갖는 노드에 대해서만 색인어를 추출하여 내용 색인을 구성한다. 그리고, 내용 색인의 구성은 XML 문서로부터 추출

A) 형제 노드가 없는 특정 노드의 삭제
1. 삭제되는 노드의 모든 하위 노드를 삭제함
2. 부모 노드의 COL에 제거 및 정렬, Ccount 감소

B) 형제노드가 있으면서 자신이 SOLST을 유지하지 않는 노드의 삭제
1. 삭제되는 노드의 모든 하위 노드를 삭제함
2. 부모 노드의 COL에서 제거 및 정렬, Ccount 감소
3. 형제 노드 중 SOLST를 유지하는 노드에서 자신의 SORD를 삭제, Scount 감소

C) 형제 노드를 가지고 있으면서 자신이 SOLST를 유지하는 경우
1. 삭제되는 노드의 모든 하위 노드를 삭제함
2. 부모 노드의 COL에서 제거 및 정렬, Ccount 감소
3. 자신의 노드를 이전할 적절한 형제 노드를 찾아 SOLST를 이전하고 삭제됨, Scount 감소

그림 10 3가지 경우의 엘리먼트 삭제

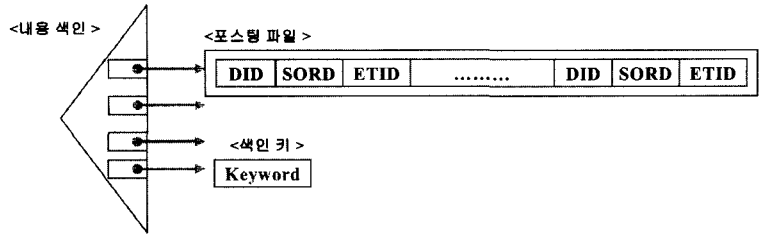

그림 11 내용 색인의 구조

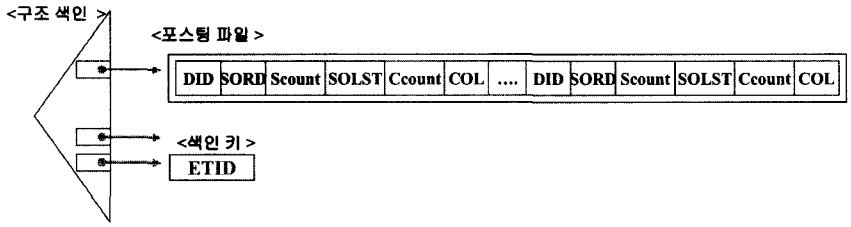

그림 12 구조 색인의 구조

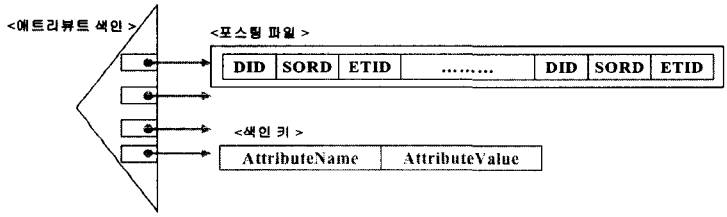

그림 13 애트리뷰트 색인의 구조

된 색인어를 통해 색인된 색인 파일과 색인어가 출현한 문서와 엘리먼트의 정보를 나타내는 포스팅 파일로 구성된다. 포스팅 파일은 DID, ETID, SORD로 구성된다. DID는 문서 단위 검색의 경우 반환 값으로 사용되며, SORD는 엘리먼트 단위의 검색, 그리고 ETID는 내용+구조 검색의 경우 사용된다. 그림 11은 내용 색인의 구조를 나타낸다.

4.3.2 구조 색인

구조 색인은 구조 검색을 지원하는 색인으로서 문서의 논리적인 구조 정보를 손실 없이 표현해야 하며, 이를 통해 엘리먼트 간의 계층관계 검색, 엘리먼트 간의 순서 관계인 형제 및 자식 검색뿐만 아니라, 형제 노드 또는 자식 노드간의 범위 검색을 지원한다. 그림 12는 구조 색인의 구조를 나타낸다.

4.3.3 애트리뷰트 색인

애트리뷰트 색인은 애트리뷰트 검색을 지원하며 색인으로서 추출된 구조 정보 중 애트리뷰트 이름과 값을 색인 파일로 구성하고 이와 관련된 DID, ETID, SORD들은 포스팅 파일을 구성한다. 다음 그림 13은 애트리뷰트 색인의 구조를 나타낸 것이다.

5. 색인 구조에 따른 질의처리 알고리즘

제안하는 색인 구조(ESSIE)에서는 각각의 색인 구조별로 효율적인 검색을 지원하기 위해 질의처리 알고리즘을 분류해 볼 수 있다. 첫째, 내용 색인을 이용한 검색의 경우, 구조 검색 및 키워드 검색을 할 수 있다. 내용 색인 앞에서 설명한 바와 같이 색인어를 통한 문서 및 엘리먼트 검색이 가능하고 ETID를 이용하여 색인어가 속한 엘리먼트의 조상이나 후손을 검색하는 일 부분의 구조 검색을 지원하지만, 색인어와 함께 순서 정보를 요구하는 검색은 지원할 수 없다. 이러한 경우는 구조 검색과 혼용하여 질의를 처리하여야 한다. 둘째, 구조 색인을 이용한 검색의 경우 키워드 검색을 제외한 어떠한 구조 검색도 가능하다. 셋째, 애트리뷰트 색인을 이용한 검색의 경우 애트리뷰트 이름과 값을 이용한 질의처리가 가능하며, 내용 색인과 마찬가지로 조상 및 후손

의 구조 정보 검색을 지원할 수 있으나 순서 정보를 요구하는 질의에 대해서는 구조 검색과 혼용하여 질의를 처리하여야 한다.

5.1 내용 검색 알고리즘

내용 검색의 경우 단순한 색인어를 키워드로 한 엘리먼트 및 문서 검색을 지원하다. 그리고, 내용 색인을 이용하여 간단한 구조 검색을 함께 지원할 수 있다. 하지만, 구조 검색 질의중 순서 정보를 포함하고 있으면 지원할 수 없다. 그림 14는 색인어 질의 처리 및 간단한 구조 정보 검색 알고리즘을 보여준다. 우선 질의문을 분석하여 질의문에 사용된 엘리먼트의 ETID를 ETID 매핑 테이블로부터 구한다. 그리고 내용 색인을 접근하여 해당 색인어에 대한 (DID, SORD, ETID)의 집합을 구한다. 이렇게 구한 집합 중 ETID가 질의문에 사용된 엘리먼트의 ETID인 것만을 추출한다. 추출된 집합 중 SORD 및 ETID의 적절한 조작으로 최종 질의에 대한 결과를 산출한다.

5.2 구조 검색 알고리즘

구조 검색이란 XML 문서 내에 사용된 특정 엘리먼트의 부모, 자식, 조상, 후손 노드를 찾는 것으로 정의할 수 있다. 그림 15는 구조 검색의 한 예로 특정 엘리먼트의 부모 엘리먼트를 찾기 위한 검색 알고리즘을 보여준다. 본 논문에서 제안한 색인 구조에서 부모를 검색하기 위해서는, 일단 해당 엘리먼트의 ETID를 ETID 매핑테이블로부터 구한다. 그리고, ETID 끝 두 자리를 잘라낸 다음 부모 엘리먼트의 ETID를 구해서, 구조 색인을 이용해 해당 엘리먼트의 (DID, SORD) 집합을 얻는다. 만약 질의에 사용된 엘리먼트의 ETID가 ETID 매핑테이블에 1개 이상 존재할 경우 그 수만큼 검색을 반복하여 수행한다.

5.3 구조 + 내용 검색

구조+내용색인을 혼합한 검색의 질의 요구는 색인어를 포함하면서 문서 내에 사용된 엘리먼트의 순서 정보를 요구하는 경우이다. 이러한 경우 질의의 유형에 따라

```
query_keyword = 질의 문을 분석, 검색 하고자 하는 색인어를 구함
content_etid = ETID 매핑 테이블을 이용하여 질의에 사용된 엘리먼트의 ETID를 구함
contentResult_set = query_keyword 기반으로 내용색인을 접근하여 구한
                    (DID,SORD,ETID) 집합
for(content_etid 수만큼)
{
   for(contentResult_set 수만큼)
   {
      if (contentResult_set[contentResult_count].etid != content_etid[content_count].etid){
         contentResult_count++
         continue
      }
      lastContentResult_set[contentResult_count].sord = contentResult_set[contentResult_count].sord
      contentResult_count++
   }
   content_count++
}
```

그림 14 내용 검색 알고리즘

```
count_etid = ETID 매핑 테이블로 질의에 사용된 엘리먼트의 ETID를 구함
for(count_etid 수만큼)
{
   parent_etid = 매핑 테이블로 부터 구한 ETID에서 끝자리 두 자리를 잘라 부모 ETID를 구함
   result_set = 구조 색인에서 parent_etid[i]를 색인키로 검색하여 (did,sord)집합을 구함
   for ( result_set의 검색 결과 수 만큼)
       {
          parent_set[result_count].sord = result_set[result_count].sord
          result_count++
       }
   count_etid++
}
```

그림 15 구조 검색 알고리즘 (부모)

두 가지로 검색 알고리즘이 분류될 수 있는데, 첫째는 내용+구조에서 구조 검색이 자식의 순서정보를 요구하는 경우이고 둘째는 첫째와 같은 경우이지만 구조 검색의 부분이 형제 노드에 대한 순서 정보를 요구하는 것이다. 앞에서 구조 검색만을 이용하는 경우 이에 대한 알고리즘을 기술하였으므로, 이번 절에서는 구조+내용을 혼합하여 질의를 처리하는 경우 중, 구조 정보의 질의 부분이 같은 타입의 형제 노드에 대한 순서 정보를 요구하는 검색 알고리즘을 기술하도록 하겠다. 그림 16은 혼합 검색 중 구조+내용 검색에 대한 검색 알고리즘이다. 우선 질의를 분석하여 순서 정보를 얻는다. 그리고, 해당 엘리먼트의 ETID를 ETID 매핑 테이블로부터 가져온다. 또한, 가져온 엘리먼트의 ETID를 색인키로 구조 색인을 접근하여 얻은 결과의 집합 중, 질의에서 요구하는 순서 정보에 맞는 (DID, SORD) 집합만을 추출한다. 그런 다음 내용 검색에 사용된 "색인어"를 가지고 내용 색인을 접근하여 얻은 집합 중에서ETID가 질의 조건에 맞는 엘리먼트의 ETID이면서, 구조 색인을 이용해 얻은 SORD와 일치하는 (DID, SORD)의 집합만을 추출한다.

6. 성능 평가

제안한 구조 정보 표현을 이용하여 색인을 구성할 경우 성능 평가를 위해 기존의 IM-K(K-ary 완전 트리 구조 기반의 색인 모델), IM-E(엘리먼트 단위 구문 트리 색인 구성), ESSC(ETID, SORD, SSORD, Ccount)

```
element_order = 질의 문을 분석, 검색 하고자 하는 엘리먼트의 순서 정보를 구함
element_etid = ETID 매핑 테이블을 이용하여 질의에 사용된 엘리먼트의 ETID를 구함
contentResult_set = 질의문에 사용된 키워드를 이용하여 얻은 내용 검색 결과(DID, SORD, ETID)
for(element_etid 수만큼)
{
   result_set = 구조 색인에서 element_etid[i]를 색인키로 검색하여,
                (DID, Scount, STSOL)집합을 구함
   for ( result_set의 검색 결과 수 만큼)
   {
        if ( result_set[result_count].Scount < element_order ){
            result_count++
            continue
        }
        element_set[result_count].sord = result_set[result_count].STSOL.list[element_order]
        result_count++
   }
   element_count++
}
for ( contentResult_set 수 만큼)
{
    if (contentResult_set[contentResult_count].etid == element_etid[element_count].etid ){
        for(element_set 수만큼)
        {
            if( contentResult_set.sord != element_set[element_count].sord){
                element_count++
                continue
            }
            lastResult_set[element_count].sord = element_set[element_count].sord
            element_count++
        }
   }
   contentResult_count++
}
```

그림 16 혼합 검색 알고리즘 (구조+내용)

색인 기법, XISS(경로 조인을 이용한 색인구조), 그리고 제안하는 ESSIE 색인 모델과의 비교를 통해 평가한다. XISS 방법은 키워드 기반에 대한 검색을 고려하고 있지 않아 경로표현에 관한 검색만을 고려하여 성능 평가한 결과를 보인다.

성능평가 환경은 CPU 400MHz, 메모리 1G의 SUN 울트라 2이며, 자바로 구현하였고, 사용한 DBMS는 오라클 8.0.1이다. 성능을 비교함에 있어 동적 환경을 고려한 문서의 삽입 및 내용, 구조, 혼합 검색의 성능 평가를 수행한다. 성능 평가를 위한 실험 데이타는 논문지 형식에 맞게 설계된 DTD에 의해 작성된 문서 1000개를 이용하였다. 다음의 그림 17은 실험 데이타를 보여준다. 성능 평가를 위해 사용된 실험데이타는[1]를 기반으로 하고 있다. 실험에 사용된 질의는 표 1에 나타나 있다.

본 논문에서 수행한 성능 평가의 전체적인 비교 결과를 표 2와 3에서 나타낸다. 표 2는 동적 환경을 고려하지 않은 경우의 평균 성능 평가 결과이고, 표 3은 동적 환경을 고려한 경우의 평균 성능 평가를 나타낸 것이다. 표에서 알 수 있듯이 전반적으로 제안한 방법이 기존의 방법보다 우수한 성능을 나타나고 XML 문서의 구조가 많이 변경되는 경우에 좋은 결과를 보인다. 그러나 제안한 방법이 기존의 방법보다 내용 및 구조 검색 그리고 동적 환경을 고려한 문서의 삽입 및 삭제를 효율적으로 처리하기 위하여 보다 더 많은 정보를 중첩하여 저장함으로써 추가적인 저장 공간을 필요함으로 인해 동적 환경을 고려하지 않은 상황에서는 저장 공간의 비효율성을 갖는 단점을 가진다. XISS 색인 구조는 별도의 경로 정보 없이 조상-후손 관계를 빠르게 결정할 수 있고 고정된 길이의 경로나 길이를 알 수 없는 경로의 검색

전체 문서 개수	1000
평균 문서의 크기	50KB
문서당 평균 엘리먼트 개수	1,150
문서당 평균 애트리뷰트의 개수	280
엘리먼트의 평균 깊이(Depth)	6
엘리먼트의 평균 자식 개수	5
K-ary기반 색인에서 평균 차수	56
전체 엘리먼트 개수	978,000
전체 키워드 개수	1,928,310

그림 17 실험 데이타를 위한 구성

표 1 실험에 사용된 질의

내용 검색	키워드
구조 검색	parent::title child::title ancestor::title descendant::title
혼합 검색	parent::title[.="검색"] child::title[.="XML"] ancestor::title[.="색인"] descendant::title[.="구조"]
경로표현 검색	part/*/section /chapter/*/figure[@cap="XML"]

표 2 동적 환경을 고려하지 않은 성능평가 결과

		K-ary 색인모델	IM-E 색인모델	ESSC 색인모델	XISS 색인모델	제안하는 색인모델
평균검색시간	내용 검색	3.2초	3.2초	3.1초	-	3.1초
	구조 검색	6초	5.5초	4.8초	-	5.1초
	내용+구조 검색	9.2초	8.7초	7.9초	-	8.2초
	경로 표현 검색	8초	7.5초	6.8초	3.63초	7.1초
	문서 삽입	10초	6.4초	6초	-	6.5초

표 3 동적 환경을 고려한 성능평가 결과

		K-ary 색인모델	IM-E 색인모델	ESSC 색인모델	제안하는 색인모델
평균검색시간	내용 검색	3.2초	0.92초	3초	0.76초
	구조 검색	4.5초	1.8초	3.5초	1.3초
	내용+구조 검색	8초	2.5초	5초	2초
	문서 삽입	10초	3.9초	7.6초	3.4초

에 적합한 색인 구조이지만, 키워드 기반의 검색을 지원하지 않아 이 기능을 추가해야 한다.

7. 결 론

본 논문에서는 XML 문서의 다양한 구조 검색을 위한 효율적인 동적 색인 모델을 제안하였다. 제안한 색인 모델은 다양한 사용자 질의와 문서의 변경이 발생할 경우 색인 정보의 변경 없이 변경된 부분의 구조정보를 효율적으로 처리한다. 또한, K-ary 완전 트리를 이용한 방법과 엘리먼트 구문 트리를 이용한 방법, ETID, SORD, SSORD, Ccount 색인모델, 경로 조인을 이용하는 색인모델과의 성능 평가를 실시하였다. 성능 평가 결과 문서의 변경이 발생하지 않은 경우에는 구조 검색 및 내용검색에서 그다지 성능의 차이를 보이지 않았으나, 문서의 변경을 고려한 동적 환경에서의 구조 검색 및 내용 검색에서는 상당히 우수한 성능 평가를 보였다.

하지만 본 논문은 ETID 생성 시 부엘리먼트의 깊이가 깊어지는 경우 ETID가 길어지는 문제가 있다. 향후 이와 같은 문제가 발생하지 않도록 제안하는 색인 모델을 보완할 예정이다.

참 고 문 헌

[1] 한성근, 송정한, 장재우, 김현기, 강현규 "동적 환경에 적합한 SGML인덱스 관리자의 설계 및 구현", 한국정보처리학회, 제6권 10호, pp. 12~24, 1999.

[2] R. Sacks-Davis, T. Arnold-Moore, and J. Zobel, "Database systems for structured documents," Proc. The International Symposium on Advanced Database Technologies and Their Integration (ADTI '94), Nara, Japan, pp. 277~283, 1994.

[3] Sung-Geun Han, Jeong-Han Son, Jae-Woo Chang and Zong-Chel Zhoo, "Design and Implementation of a Structured Information Retrieval System for SGML documents," Database Systems for Advanced Applications, pp. 81~88, 1999.

[4] 박종관, 강형일, 손충범, 유재수 "XML 문서에 대한 효율적인 구조 기반 검색을 위한 색인 모델", "한국정보과학회, 2000 추계 학술발표논문집, pp. 18~20, 2000.

[5] Quanzhong Li, Bongki Moon, "Indexing and Querying XML Data for Regular Path Expression," VLDB, pp. 361~370, 2001.

[6] 민영수, 강승헌, 강형일, 유재수, 이하욱, 최한석, "XML 문서를 위한 구조정보 추출기의 설계 및 구현", 한국정보과학회 '99 가을 학술발표논문집(Ⅰ), 한국정보과학회, pp, 81~83, 1999.

[7] 연제원, 조정수, 이강찬, 이규철, "XML 문서 구조검색을 위한 저장 시스템 설계", 한국정보과학회 학술 발표 논문집(B), 제26권 1호, pp. 3~5, 1999.

[8] Alin Deutsch, Mary Fernandez, and Dan Suciu, "Storing Semistructured Data with STORED," SIGMOD, 1999.

[9] Brian Lowe, Justin Zobel, Ron Sacks-Davis "A Formal Model for Databases of Structured Text," Proceedings of the Fourth International Conference on Database Systems for Advanced Applications (DASFAA '95), pp. 449~456, 1995.

[10] Chow, J.H., Cheng, J., Chang, D., Xu, J., "Index Design for Structured Documents Based on Abstraction," Proceedings of the 6th International Conference on Database Systems for Advanced Applications, pp. 89~96, 1999.

[11] Ricardo Baeza-Yates and Gonzalo Navarro, "Integrating Contents and Structure in Text Retrieval," FONDECYT, 1995.

[12] Shin, D.W., Jang, H.C., Jin, H.L., "BUS: An Effective Indexing and Retrieval Scheme in Structured Documents," Proc. Digital Libraries 98, 1998.

[13] S.H. Myaeng, D.H. Jang, M.S. Kin, and Z.C. Zhoo, "A Flexible Model for Retrieval of SGML Documents," SIGIR'98 ACN, pp. 138~145, 1998.

[14] Tuong Dao, Ron Sacks-Davis, James A. Thom, "An Indexing Scheme for Structured Documents and its Implementation," Proceedings of the Fifth International Conference on Database Systems for Advanced Applications (DASFAA '97), pp. 125~134, 1997.

[15] Brain Cooper, Neal Sample, Michael J. Franklin, Cisli R. Hjaltason and Moshe Shadmon, "A Fast Index for Semistructured Data," VLDB, 2001.

[16] Chun Zhang, Jeffrey F. Naughton, David J. DeWitt, Qiong Luo and Guy M. Lohman, "On SUpporting Containment Queries in Relational Database Management Systems," SIGMOD, 2001.

신 승 호

2000년 충주대학교 컴퓨터공학과(공학사). 2002년 충북대학교 정보통신공학과(공학석사). 2003년~현재 대우정보시스템(주). 관심분야는 데이타베이스 시스템, 모바일시스템, XML, 멀티미디어 정보 검색 등

손 충 범

1997년 충북대학교 정보통신공학과(공학사). 1999년 충북대학교 정보통신공학과(공학석사). 2002년 충북대학교 정보통신공학과(공학박사). 2003년~현재 인하공업전문대학 정보통신과 전임강사. 관심분야는 데이타베이스 시스템, XML, 멀티미디어 정보 검색 등

강 형 일

1996년 목포대학교 전산통계학과(이학사). 1998년 목포대학교 전산통계학과(이학석사). 2001년 충북대학교 정보통신공학과(공학박사). 2001년~현재 주성대학 멀티미디어정보통신공과 조교수. 관심분야는 데이타베이스 시스템, XML, 정보검색 등

유 재 수

1989년 전북대학교 공과대학 컴퓨터공학과(학사). 1991년 한국과학기술원 전산학과(공학석사). 1995년 한국과학기술원 전산학과(공학박사). 1995년~1996년 목포대학교 전산통계학과 전임강사. 1996년~현재 충북대학교 전기전자컴퓨터공학부 부교수 및 컴퓨터·정보통신연구소 소장. 관심분야는 데이타베이스 시스템, 정보검색, 멀티미디어 데이타베이스, 분산 객체 컴퓨팅