XML 문서의 효율적인 구조 검색을 위한 색인 모델

박 종 관[†] · 손 충 범[††] · 강 형 일[†††] · 유 재 수[††††] · 이 병 엽[†††††]

요 약

본 논문에서는 XML 문서의 효율적인 구조 검색을 위한 색인 모델을 제안한다. 제안한 색인 모델은 문서 계층상의 모든 레벨에서 내용 기반 질의와 구조 및 애트리뷰트 질의와 같은 다양한 질의를 지원하기 위한 구조정보와 이를 이용한 색인 구조로 구성된다. 구조검색을 지원하기 위해 새로운 구조정보 표현 방법을 제안한다. 제안한 구조정보 표현 방법에 따라 표현된 구조정보를 이용해 효율적인 검색을 위한 내용 색인, 구조 색인, 애트리뷰트 색인을 구성한다. 또한 내용과 구조가 혼합된 질의의 처리과정을 설명하고, 제안하는 색인 모델의 성능평가를 통하여 기존의 방법보다 성능이 우수함을 보인다.

An Indexing Model for Efficient Structure Retrieval of XML Documents

chong kwan Park[†] · Chung Beon Son[††] · Hyung Il Kang[†††] ·
Jae Soo Yoo[††††] · Byoung Yup Lee[†††††]

ABSTRACT

In this paper, we propose an indexing model for efficient structure retrieval of XML documents. The proposed indexing model consists of structured information that supports a wide range of queries such as content-based queries and structure-attribute queries at all levels of the document hierarchy and index organizations that are constructed based on the information. To support structured retrieval, a new representation method for structured information is presented. Using this structured information, we design content index, structure index, and attribute index for efficient retrieval. Also, we explain processing procedures for mixed queries and evaluate the performance of proposed indexing model. It is shown that the proposed indexing model achieves better retrieval performance than the existing method.

키워드 : XML, 구조 검색(Structure Retrieval), 색인 모델(Indexing Model), 구조 정보 표현(Structure Information Representation)

1. 서 론

최근 인터넷 사용과 정보의 양이 급증하면서 인터넷상의 정보를 보다 효과적으로 사용하고자 하는 연구가 활발히 진행되고 있다. 한편 웹에서 가장 많이 사용되고 있는 HTML은 문서의 구조보다는 표현에 중점을 두고 있어 특정 응용분야의 구조를 표현하는 문서로는 기능이 부족하다는 단점이 있다[2]. 이에 웹 발전의 주도적인 역할을 하고 있는 W3C(World Wide Web Consortium)에서는 차세대 웹 문서의 표준으로 XML(eXtensible Markup Language)이라는 전자문서 메타 언어를 1996년에 제안하였으며 현재까지 그 기능이 계속 확장되고 있는 상태이다[5].

XML은 이기종간의 시스템에서 작성된 문서의 상호 교환과 다양한 형식의 문서들을 일관성 있게 구조화하기 위하여 고안된 SGML을 간략화 시키고 HTML보다 사용자의 다양한 요구를 충분히 수용할 수 있어 웹 문서뿐만 아니라 전자도서관, CSCW(Computer Supported Cooperative Work), EDI(Electronic Data Interchange), CALS(Comme-rce At the Light Speed), e-business 등 다양한 분야에서 XML을 활용하고자 폭 넓은 연구를 하고 있으며 구체화되는 XML 응용이 부쩍 많아지고 있다.

DTD(Document Type Definition)의 논리적 구조를 따르는 XML 문서의 구조정보는 XML 문서의 관리나 구조 검색을 효율적으로 수행하는데 이용된다[1]. 이러한 논리적 구조를 수용하려는 기존의 구조정보 표현을 위한 연구들은 XML 문서를 가지고 구조정보를 표현하려고 했다. 일부 XML 문서 위주의 구조정보 표현 방법은 문서마다 특정 엘리먼트에 부여된 ID가 다를 수 있다[7, 11]. 또한, 특정 엘리먼트에 대한 직접적인 접근이 불가능하거나 조상, 자손, 형제의 관계에 있는 엘리먼트를 접근하기 위해 복잡하거나 반복적인 연산을 수행해야 했다[2, 3, 11-13].

※ 본 연구는 한국과학재단 목적기초연구(과제번호 : 1999-1-303-007-3) 지원으로 수행되었음.
† 준 회 원 : 대우정보시스템(주)
†† 준 회 원 : 충북대학교 대학원 정보통신공학과
††† 준 회 원 : 주성대학 멀티미디어정보통신공학부 교수
†††† 종신회원 : 충북대학교 정보통신공학과 및 컴퓨터정보통신연구소 교수
††††† 정 회 원 : 대우정보시스템 e-솔루션사업팀 차장
논문접수 : 2000년 11월 13일, 심사완료 : 2001년 9월 11일

이에 본 논문에서는 특정 엘리먼트에 대한 직접적인 접근이 가능하고 엘리먼트 간의 관계를 구하기 위해 복잡한 연산이 필요 없도록 DTD에 나타난 엘리먼트들과 XML 문서의 구조정보를 사용해서 XML 문서를 효율적으로 관리하고 검색할 수 있는 구조정보를 제안하고, 이 정보들을 이용해 효율적인 검색을 위한 색인 구조를 구성하는 색인 모델을 제안한다. 또한 제안한 색인 모델에서 혼합 질의에 대해 처리 과정을 보이고 검색 측면에서 성능 평가를 수행한다.

본 논문의 구성은 다음과 같다. 2장에서 XML 문서에 대해 질의 유형을 알아보고, 구조정보를 표현하고 검색하는 방법에 대한 기존 연구들을 살펴본다. 3장에서 효율적인 검색을 위한 구조정보 표현을 제안하고 전체적인 시스템 구성을 설명한다. 4장에서는 추출된 구조정보를 이용해 색인을 구성하고 혼합 질의 처리의 예를 보이며 5장에서는 제안한 색인 모델의 성능 평가를 기술한다. 마지막으로 7장에서는 결론과 향후 연구 방향을 제시한다.

2. 관련 연구

XML 문서를 구성하는 기본 단위는 엘리먼트이다. 따라서 기존의 정보 검색 시스템에서 사용한 문서 단위를 위주로 한 검색 이외에 임의 깊이에서 나타나는 엘리먼트 단위의 검색이 가능해야 한다. 또한 엘리먼트 간의 논리적인 포함관계, 형제 관계 및 엘리먼트의 애트리뷰트 값에 대한 질의도 지원되어야 한다. 이와 같은 특성을 갖는 XML 문서에 대한 검색은 크게 내용 검색, 구조 검색 그리고 내용과 구조가 혼합된 혼합 검색으로 나눌 수 있으며 부가적으로 애트리뷰트 검색이 있다.

내용 검색은 사용자에 의해 주어지는 키워드와 관련된 문서나 엘리먼트를 검색하는 것으로 단순 검색, 불리언 검색, 가중치 검색, 와일드 카드 검색, 제한 검색 등으로 지원될 수 있다. ["구조검색"을 포함하는 문서를 검색하시오]와 같은 질의를 예로 들 수 있다. 구조 검색은 문서의 논리적 구조에 대한 검색으로 부모, 조상, 자식, 자손, 형제에 대한 검색이 될 수 있다. 이런 구조 검색은 [<author>가 3명인 논문을 검색해 주시오], [<EnglishAbstract>이 있는 논문을 검색해 주시오]와 같은 질의가 될 수 있다. 혼합 검색은 내용과 구조 정보를 이용한 검색, 내용과 애트리뷰트를 이용한 검색, 애트리뷰트와 구조 정보를 이용한 검색 등으로 나눌 수 있다. [<title>에 "XML"을 포함하는 논문을 검색해 주시오], [2장에 "구조검색"을 포함하는 논문을 검색해 주시오], [첫 번째 <author>이 "홍길동"인 논문을 검색해 주시오]와 같은 질의들이 이에 해당한다. 애트리뷰트 검색은 엘리먼트에 나타날 수 있는 애트리뷰트에 대한 검색으로 애트리뷰트 이름과 값을 주고 그에 해당하는 문서나 엘리먼트를 찾는 질의이다. [<author>의 애트리뷰트 sex가 "female"인 문서를 검색해 주시오]와 같은 질의를 예로 들 수 있다. 이와 같은 XML 문서의 특성에 기반한 질의를 지원하기 위한 색인 구조에 대한 기존의 연구는 다음과 같다.

호주 RMIT에서 제시한 SCL(Simple Concordance List) 모델은 정의된 문서 계층과 정의된 마크업 스키마로부터 독립을 제공한다. 이 모델은 텍스트 간격들을 다루며 문서 구조에 대한 질의를 지원하기 위해 계층적인 관계보다 오히려 포함 관계를 사용한다. 이 모델은 SC-list라 불리는 데이터타입에서 중첩된 정보를 허용하기 때문에 리스트의 리스트와 같은 순환 구조를 다룰 수 있다[12, 13]. 그러나 SCL 구조는 색인어를 포함하는 엘리먼트에 대해서는 알 수 있으나 엘리먼트들에 대해 트리내의 깊이를 표현할 수 없다는 단점을 가진다. 즉, 특정 엘리먼트의 조상, 형제를 알수 없으며 포함하고 있는 엘리먼트가 몇 번째 자손에 해당하는지 알 수 없다.

K-ary 완전 트리 방법은 SGML 문서를 K-ary 완전 트리에 매핑하여 구조 검색을 지원하는 방법이다. K-ary 완전 트리의 특성상 i번째 노드의 부모를 구하기 위해 연산식 $Parent(i) = (i-2)/K+1$을 이용하고 i번째 노드의 j번째 자식을 구하기 위해 연산식 $Child(i, j) = K(i-1)+j+1$을 이용하여 문서의 엘리먼트들 간의 부모와 자식의 관계를 손쉽게 알 수 있는 장점을 가진다[7, 11]. K-ary 완전 트리 방식은 계산에 의해서 논리적 포함 관계인 엘리먼트를 빠르게 찾을 수 있는 장점이 있으나, 노드의 깊이가 깊어질수록 노드 변화가 커지고 사용하지 않는 노드 번호가 많아지므로 데이터의 양이 커지는 단점이 있다. 이로 인해 조상, 자손, 형제 등의 관계를 알기 위해 어렵거나 복잡한 연산을 요구하게 된다. 또한 K-ary 완전 트리의 특성상 연산을 통해 구한 자식이나 형제의 노드 번호가 실제 존재하는 노드인지 확인해야 하는 단점을 가지고 있다.

추상화에 기반한 방법은 구조화된 문서의 문법적 구조를 부모와 자식의 관계를 나타낼 수 있는 트리로 표현하여 문서 구조를 나타내고 가능한 문서를 읽지 않고 색인 내에서 경로 표현을 구하기 위해 문서 구조의 특정 형태를 취하는 방법이다. 즉, 실제 문서들의 구조를 추상화한 색인 구조인 추상화에 기반한 구조 검색 색인을 설계하는 것이다. 또한 추상화는 순수 내용과 애트리뷰트에 적용될 수 있다. 추상화는 여러 가지 방법이 있고 활용하려는 응용에 맞게 결정해야 한다[4]. 추상화 색인의 경로 표현을 통해 문서에서 부모, 자식, 자손들을 알 수 있고 색인의 크기를 줄일 수 있으나 완전한 구조 검색 즉, 추상화되지 않은 문서 구조의 검색을 위해서는 문서 전체를 읽어야 하는 오버헤드가 있다. 또한 각각의 응용에 맞게 추상화 방법을 결정하여야 한다. 이런 추상화의 차수에 따라 지원할 수 있는 구조 검색의 정도가 다르고 형제 엘리먼트들의 순서를 알 수 없다.

엘리먼트 단위 구문 트리 구조 방법[15]은 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. 엘리먼트 단위 구문 트리 구조의 경우 문서의 부분 삽입 및 삭제가

발생하게 되면 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 문서 구조의 순서 정보 검색을 지원하기 위한 인덱스 구조의 반복적인 노드 순회가 필요하다는 단점이 있다.

3. 구조정보 표현 및 시스템 구성

본 논문에서 제안하는 구조정보는 엘리먼트 이름을 식별할 수 있는 ID, 부모와 자식 엘리먼트간의 계층정보, 동일한 부모 엘리먼트를 갖는 자식 엘리먼트(이하 형제 엘리먼트)들의 순서정보, 그리고 동일한 부모 엘리먼트를 갖는 자식들 중 동일한 타입의 엘리먼트들에 대한 순서정보로 표현된다. 이들은 부모 엘리먼트의 정보를 유지한다. 그러므로 기준 엘리먼트로부터 특정 엘리먼트에 대한 계층정보와 순서정보를 간단한 문자열 조작만으로 쉽게 구할 수 있다. 이렇게 구한 순서정보는 각 엘리먼트에 유일하게 할당된 값이기 때문에 직접 특정 엘리먼트를 접근할 수 있다.

3.1 엘리먼트 식별 ID

엘리먼트들 간의 계층 정보를 표현하기 위해서는 엘리먼트 이름에 대해 식별할 수 있는 EID(Element ID)를 부여해야 한다. 이러한 EID는 DTD에서 정의된 각 엘리먼트에 대하여 유일한 ID를 의미한다. 각 엘리먼트에 대해 ID를 부여하는 방식을 통해서 DTD의 논리적 구조를 분석할 때 발생하는 엘리먼트간의 순환 문제와 ANY로 설정된 엘리먼트 처리 문제를 제거할 수 있다.

EID는 2바이트를 사용하여 표현하는데 각 바이트는 '0' → '9' → 'A' → 'Z' → 'a' → 'z' 순으로 된 62개의 문자를 사용하며 ASCII 코드의 순서를 따르고 있다. 이를 통해 EID는 총 3844개의 엘리먼트를 표현할 수 있다. (그림 1)은 DTD에서 정의된 각 엘리먼트에 대해 EID를 부여하고 이 EID를 유지하는 EID 맵핑 테이블을 보여 주고 있다.

3.2 구조정보 표현 방법

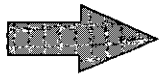
DTD의 논리적 구조를 따르는 XML 문서의 구조정보는 엘리먼트를 기반으로 한다. 이러한 XML 문서상의 엘리먼트를 유일하게 구별하면서 엘리먼트들 간의 계층정보를 표현하기 위해서 일반적으로 ID를 부여한다. 본 논문에서는 특정 엘리먼트를 구별하면서 엘리먼트들 간의 계층정보를 표현하기 위해 EID를 사용하여 문서상의 엘리먼트들에 ETID (Element Type ID)를 부여한다. 이렇게 부여된 ETID는 문서의 논리적 구조를 나타내게 된다. 또한 XML 문서에서는 DTD에 나타난 발생지시자에 의한 반복적인 엘리먼트의 사용이 가능하다. 이렇게 반복적으로 사용된 엘리먼트들은 ETID만으로는 구별이 불가능하다. 따라서 본 논문에서는 다양한 구조검색을 지원하기 위해서 두 종류의 순서정보를 제안한다. 형제 엘리먼트들의 발생순서를 나타내는 SORD (Sibling ORDer)와 동일한 타입의 형제 엘리먼트들 간의 순서정보를 나타내는 SSORD(Same Sibling ORDer)가 그것이다. 또한 특정 엘리먼트의 자식 검색을 위해 엘리먼트가 갖는 자식의 수를 Ccount를 사용하여 유지한다.

(그림 2)에서는 간단한 XML 문서를 보여주고 (그림 3)에서는 (그림 2)에 나타난 XML 문서의 구조정보를 ETID, SORD, SSORD, Ccount를 사용해서 트리의 형태로 표현한 예를 보여 준다.

```
<?xml version = "1.0" encoding = "euc-kr" ?>
<!DOCTYPE paper SYSTEM "paper.dtd">
<paper status = "public">
  <head>
        <title> XML 문서의 효율적인 검색을 위한 인덱
              싱 모델 </title>
  <author>
              <name>박종관</name>
        <department>정보통신공학과</department>
  </author>
  <abstract> 본 논문에서는 XML 문서의 정보 검색을 위한 질
        의들을 분석… </abstract>
  </head>
```

DTD

```
<!ELEMENT paper        (head, body, reference+)>
<!ATTLIST paper status (public | confidential) "public">
<!ELEMENT head         (title, author+, abstract)>
<!ELEMENT title        (#PCDATA)>
<!ELEMENT author       (name, department)>
<!ELEMENT name         (#PCDATA)>
<!ELEMENT department   (#PCDATA)>
<!ELEMENT abstract     (#PCDATA)>
<!ELEMENT body         (chapter+)>
<!ELEMENT chapter      (#PCDATA | section+)>
<!ELEMENT section      (#PCDATA | para | image)*>
<!ELEMENT para         (#PCDATA)>
<!ELEMENT img          (#PCDATA)>
<!ATTLIST img src CDATA #REQUIRED>
<!ELEMENT reference    (#PCDATA)>
```



EID mapping table

Element Type	EID
paper	00
head	01
title	02
author	03
name	04
department	05
abstract	06
body	07
chapter	08
section	09
para	0A
img	0B
reference	0C

(그림 1) EID 부여 예

```
<body>
        <chapter> 1. 서 론
        <section>
        <para> 서론 내용 </para>
        </section>
        </chapter>
        <chapter> 2. 색인 모델
                본 논문에서 제안하는 구조정보는 엘리
                먼트 이름을 식별할 수 있는 ID·
        <section> 2.1 구조정보
          <para> 구조 내용 </para>
          <image src = "fig1.gif"> (그림 1). XML 문서
                                예제</image>
        </section>
        <section> 2.2 성능평가
          <para> 내용 검색 </para>
            <para> 구조 검색 </para>
          </section>
        </chapter>
        <chapter> 3. 결  론 </chapter>
<body>
<reference> 참고문헌 </reference>
</paper>
```

(그림 2) 간단한 XML 문서

3.2.1 계층 정보

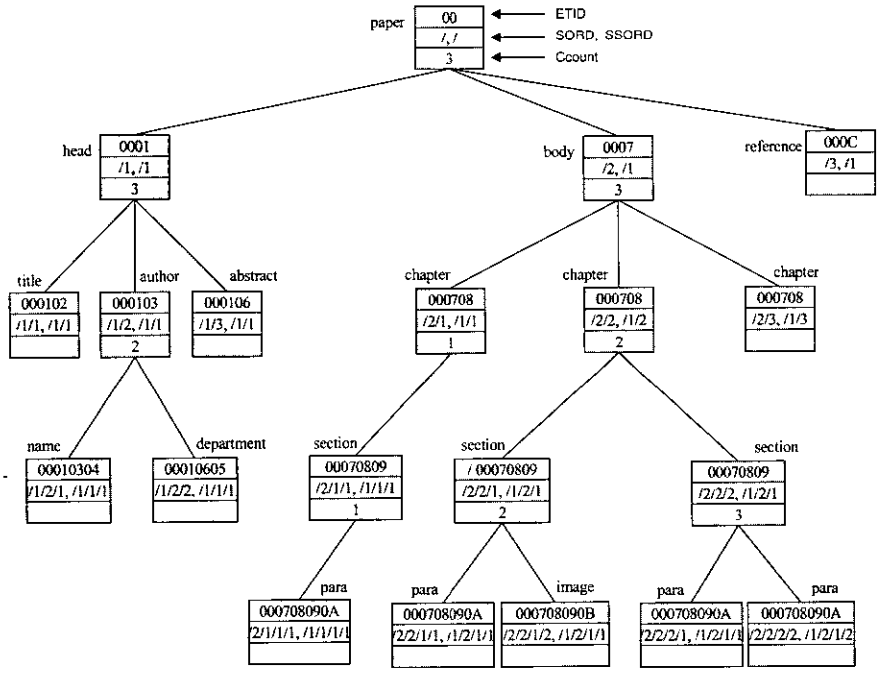
엘리먼트들 간의 계층정보를 표현하기 위해 ETID를 사용하는데 이 ETID는 앞에서 생성된 EID를 이용하여 만들게 된다. 즉, 부모의 ETID에 자신의 EID를 붙여서 자신의 ETID를 생성하게 되는데 부모가 없는 루트 엘리먼트인 경우는 자신의 EID를 ETID로 사용한다. 이 ETID는 문서의 논리적 구조상의 각 엘리먼트 타입에 부여되는 유일한 값이다.

예를 들어 (그림 2)의 구조정보 표현에서 보면 루트 엘리먼트인 <paper>의 ETID는 "00"이고, <body>의 ETID는 부모 노드의 ETID인 "00"에 자신의 EID인 "07"을 붙여 "0007"이 된다. 이렇게 생성된 ETID와 해당 엘리먼트 이름과의 사상을 위해 ETID 매핑 테이블이 존재하게 되는데 이 매핑 테이블을 참조하면 효과적인 구조 검색이 가능해진다. 즉, 엘리먼트간의 조상, 자식이 어떤 형의 엘리먼트인지 쉽게 알 수 있다. 다음의 (그림 3)는 엘리먼트 이름과 ETID의 매핑 테이블을 보여 준다.

엘리먼트들 간의 계층정보를 알아내는 한 예로 (그림 4)의 매핑 테이블에서 <section> 엘리먼트의 부모 엘리먼트는 우선 <section> 엘리먼트의 ETID "00070809"에서 뒤의 두바이트를 잘라내어 부모 엘리먼트의 ETID인 "000708"을 구하고 이 ETID에서 뒤 두바이트 "08"에 해당하는 엘리먼트를 EID 매핑 테이블에서 찾으면 section이라는 것을 알 수 있다. 결과적으로 XML 문서의 접근이 필요 없이 단지 ETID만을 가지고 엘리먼트들 간의 계층정보를 검색할 수 있다.

ElementType	ETID	Element Type	ETID
Paper	00	body	0007
head	0001	chapter	000708
title	000102	section	00070809
author	000103	para	000708090A
name	00010304	img	000708090B
department	00010305	reference	000C
abstract	000106	54	0

(그림 4) ETID Mapping Table



(그림 3) XML 문서의 구조정보 표현

3.2.2 순서정보

XML 문서에서는 동일한 엘리먼트들이 반복적으로 나타날 수 있는데 반복적인 엘리먼트들과 엘리먼트들 간의 순서정보를 표현하기 위해 SORD와 SSORD를 사용한다. 이들의 표현 방법은 UNIX 파일 시스템에서 디렉토리를 표현하는 방법을 사용하며 부모의 순서정보를 상속받는다. 예를 들어, 부모의 SORD가 "/2"(루트 엘리먼트의 2번째 자식)이고 자신의 형제들 중 세 번째로 나타난 엘리먼트라면 이 엘리먼트의 SORD는 "/2/3"이 된다. SSORD 또한 같은 원리로 부여된다. 또한 Ccount는 특정 엘리먼트의 자식 수를 나타내는데, 특정 엘리먼트의 자식을 검색하는데 유용하게 사용되고 자식의 범위를 알 수 있다. 예를 들어, SORD가 "/2/3"인 엘리먼트의 2번째 자식을 검색할 경우, 우선 Ccount의 값이 2보다 큰지 확인하고 크다면 SORD "/2/3"에 2를 붙여 자식의 SORD인 "/2/3/2"를 직접 구할 수 있다.

SORD는 XML 문서에서 나타나는 엘리먼트들을 유일하게 구분할 수 있는 값으로 특정 엘리먼트를 검색하는데 유용하게 이용된다. 또한 형제 엘리먼트들의 순서를 나타내기 때문에 "몇 번째 자식 엘리먼트"라는 질의에 사용될 수 있다. SSORD는 동일한 타입의 형제 엘리먼트들의 순서를 나타내는 것으로 "자식 엘리먼트들 중 몇 번째 특정 엘리먼트"라는 질의 처리에 효과적이다. 따라서 문서의 특정 엘리먼트는 ETID, SORD, SSORD를 이용하면 쉽게 검색할 수 있다.

3.3 시스템 구성

제안하는 색인 모델은 크게 구조정보 생성기와 색인 생성기로 구성된다. 구조정보 생성기는 EID 생성기와 XML 구조정보 추출기로 구성되는데, EID 생성기는 DTD로부터 각각의 엘리먼트들에 대해 엘리먼트 타입을 추출하고 엘리먼트 이름과 EID를 사상시켜주는 매핑 테이블을 구성한다. 그리고 XML 구조정보 추출기는 XML 문서를 분석하면서 EID 매핑 테이블을 참조하여 ETID를 생성해 내고 SORD와 SSORD를 생성하고 Ccount를 계산하여 문서의 구조정보를 추출한다. 이렇게 추출된 구조정보를 이용하여 색인 생성기는 효율적인 검색을 위해 각각의 색인을 구성한다. (그림 5)는 제안하는 색인 모델에 대한 전체적인 시스템 구성도를 나타낸다.

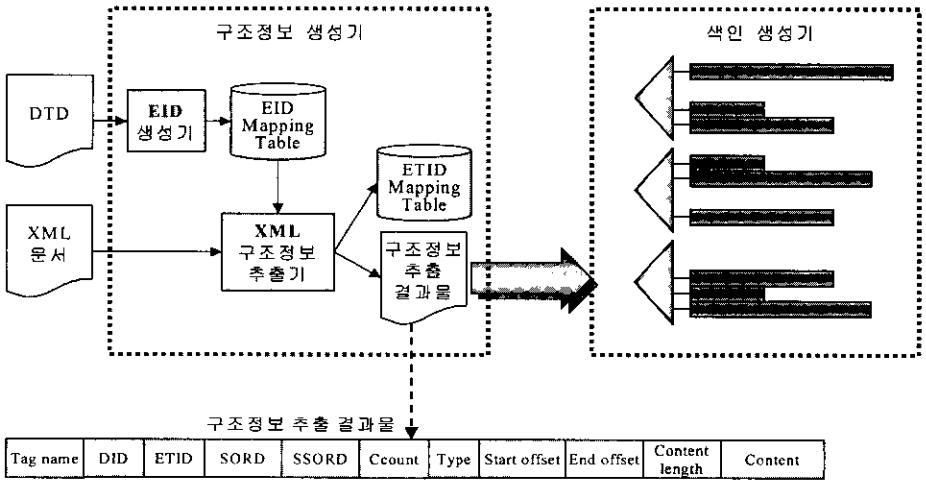
구조정보 추출기에 의해 추출되는 정보는 위의 그림에 나타난 바와 같이 여러 필드들을 갖는 텍스트 파일이다. 각 필드들은 각각의 의미를 갖는데 Tag name은 엘리먼트의 이름 혹은 애트리뷰트의 이름이며 DID는 문서를 구분하기 위해 부여된 고유번호이다. ETID, SORD, SSORD, Ccount는 이미 설명한 바와 같고 Type은 추출된 결과물이 엘리먼트인지 애트리뷰트인지를 나타낸다. Start offset와 End offset은 각각 XML 문서 화일에서 해당 엘리먼트의 시작 위치와 끝위치를 나타낸다. Content는 해당 엘리먼트에 속하는 실제 내용이거나 애트리뷰트의 값이며 Content length는 Content 필드에 있는 값의 길이를 나타낸다.

4. 색인 구성 및 질의 처리

본 장에서는 XML 문서의 특성을 고려한 다양한 질의를 효율적으로 처리할 수 있는 색인 구조를 설계한다. 이 색인은 3장에서 추출한 구조 정보를 이용하여 생성하는데 내용 색인, 구조 색인, 애트리뷰트 색인의 3개로 구성되며 기존의 역화일 방식의 B+ 트리로 구성한다.

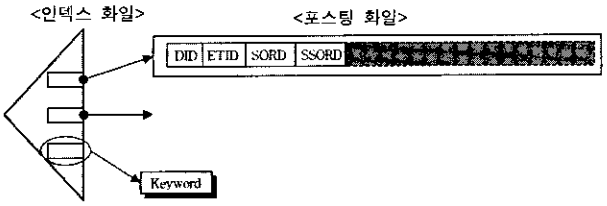
4.1 내용 색인

내용 색인은 내용 검색을 지원하는 색인로서 XML 문서로부터 추출된 색인어로 구성된 색인 화일과 색인어가 출현한 문서와 엘리먼트의 정보를 나타내는 포스팅 화일로



(그림 5) 색인 모델 시스템 구성도

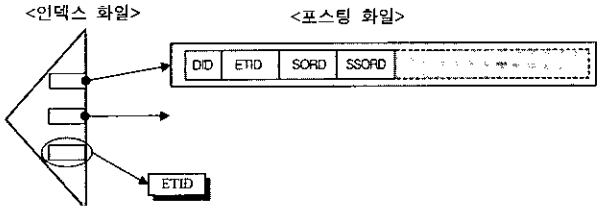
구성된다. 포스팅 화일은 DID, ETID, SORD, SSORD로 구성된다. DID는 문서 단위 검색의 경우 사용되며, ETID, SORD, SSORD는 엘리먼트 단위의 검색을 지원하기 위해 사용된다. (그림 6)는 내용 색인의 구조를 나타낸다.



(그림 6) 내용 색인의 구조

4.2 구조 색인

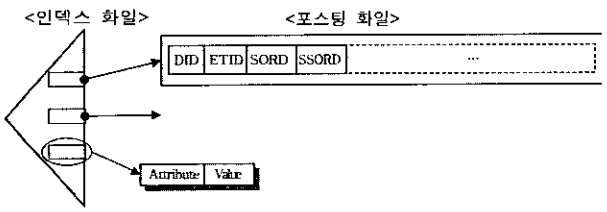
구조 색인은 구조 검색을 지원하는 색인으로서 문서의 논리적인 구조 정보를 손실없이 표현한다. 이를 통해 엘리먼트 간의 계층관계 검색, 엘리먼트 간의 순서 관계인 형제 검색을 지원한다. (그림 7)은 구조 색인의 구조를 나타낸다.



(그림 7) 구조 색인의 구조

4.3 애트리뷰트 색인

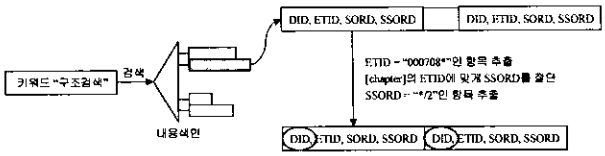
애트리뷰트 색인은 애트리뷰트 검색을 지원하는 색인으로서 추출된 구조 정보 중 애트리뷰트 이름과 값을 색인으로 구성하고 이와 관련된 DID, ETID, SORD, SSORD들을 포스팅 화일로 구성한다. 다음 (그림 8)은 애트리뷰트 색인의 구조를 나타낸 것이다.



(그림 8) 애트리뷰트 색인의 구조

4.4 질의 처리

내용 검색, 구조 검색, 애트리뷰트 검색은 각각의 색인 즉, 내용 색인, 구조 색인, 애트리뷰트 색인을 통하여 직접 지원이 되며, 혼합 검색의 경우는 이들 색인의 결합으로 지원 가능하다. 한 예로 [2장에 "구조검색"를 포함하는 논문을 검색해 주시오]라는 질의에 대해 (그림 9)는 처리 과정을 보여준다.



Element Type	ETID	Element Type	ETID
Paper	00	body	0007
head	0001	chapter	000708
title	000102	section	00070809
author	000103	para	000708090A
name	00010304	img	000708090B
department	00010305	reference	000C
abstract	000106	54	0

ETID 매핑테이블

(그림 9) 질의 처리 과정

우선 ETID 매핑 테이블에서 <chapter>에 해당하는 ETID "000708"을 가져오고, 키워드 "구조검색"을 가지고 내용 색인을 통해 "구조검색"에 대한 정보 즉, DID, ETID, SORD, SSORD를 가져온다. 가져온 정보들중 ETID가 <chapter>의 ETID인 "000708"로 시작하는 항목들을 추출하는데 그 이유는 내용 색인은 단말 노드 즉, 텍스트 데이터를 가지는 노드에 대해서만 색인이 되고, 질의 결과 엘리먼트는 단말 노드뿐만 아니라 중간 노드일 수도 있기 때문이다. 그런 다음 <chapter>의 ETID에 맞게 SSORD를 절단하고 절단된 SSORD가 "/2"로 끝나는 항목들만 추출하고 질의 결과가 문서이기 때문에 DID를 반환하게 된다. 만약 질의 결과가 엘리먼트일 경우는 DID와 SORD가 질의에 대한 반환값이 된다.

5. 성능 평가

5.1 분석적 비용 모델

본 절에서는 XML 문서에 대해 효율적인 검색을 위해 제안한 색인 모델의 내용 검색 질의, 구조 검색 질의, 혼합 검색 질의에 대한 검색 성능을 기존의 K-ary 완전 트리 구

〈표 1〉 실험 데이터

의 미	값
전체 문서 수	1,000
평균 문서의 크기	47KB
문서당 평균 엘리먼트 수	1,402
문서당 평균 애트리뷰트 수	241
엘리먼트의 평균 깊이(Depth)	6
엘리먼트의 평균 자식 수	5
K-ary기반 색인에서 평균 K값	56
전체 키워드 수	43,194
키워드의 평균 길이	6.8
엘리먼트 개수	1,402,000

조 기반의 방법과 분석적 비용모델을 기반으로 성능 평가를 실시하였다.

성능 평가를 위한 실험 데이타는 논문지의 형식에 맞게 설계된 DTD에 의해 작성된 문서 1000개를 이용하였다. 다음의 <표 1>는 실험 데이터에 대한 분석 결과이다.

본 실험에서는 검색 성능을 평가하기 위해 각각의 검색을 수행하였을 때 디스크를 접근하는 회수를 측정한다. 디스크 접근 회수를 측정하는 방법으로 각 검색 방법에 대한 대표 질의어를 정하고, 이를 수행하였을 때 검색 결과가 각각 500, 1000, 2000, 5000, 10000, 50000, 100000개인 경우에 대해 각 색인의 포스팅 파일 접근 회수를 조사하였다.

다음의 <표 2>은 본 실험에서 사용되는 매개변수들이다.

〈표 2〉 성능 평가 매개변수

매개변수	의 미	값
a	각 색인 파일 접근 회수	1
s	디스크 블록 크기(Byte)	4096
s_{k1}	내용 색인에서 포스팅 엔트리 1개 크기(K-ary)	18
s_{k2}	내용 색인에서 포스팅 엔트리 1개 크기(제안 모델)	24
s_{e1}	엘리먼트 색인에서 포스팅 엔트리 1개 크기(K-ary)	8
s_{a1}	에트리뷰트 색인에서 포스팅 엔트리 1개 크기(K-ary)	30
s_{a2}	애트리뷰트 색인에서 포스팅 엔트리 1개 크기(제안 모델)	24
s_s	구조 색인에서 포스팅 엔트리 1개 크기(K-ary)	10
s_t	구조 색인에서 포스팅 엔트리 1개 크기(제안 모델)	13
a_t	구조 색인 접근 회수(K-ary)	r_c or 1000
t	디스크 블록 1회 접근 시간	0.019
r_c	검색 결과 수	500~100000

다음은 각 검색에 대한 디스크 접근 회수를 계산하는 수식이다. 아래 수식에서 K-ary기반 색인의 경우 부모, 형제, 자식의 Uid를 계산하기 위한 K값을 구하기 위해 $a_t*(a+1)$ 만큼 구조 색인을 접근하고 있다. 또한 이미 구한 형제나 자식의 Uid가 실제 문서에 존재하는 Uid인지 가상 노드의 Uid인지를 판별하기 위해 r_c만큼 구조 색인을 접근하고 있다.

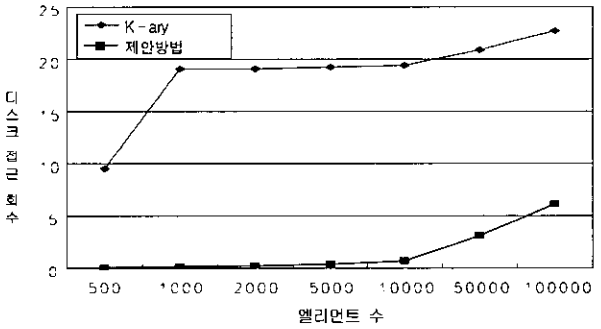
- 내용 검색
  - K-ary기반 색인 : $(a+\lceil r_c*s_{k1}/s \rceil)*t$
  - 제안한 모델의 색인 : $(a+\lceil r_c*s_{k2}/s \rceil)*t$
- 구조 검색 (부모 검색)
  - K-ary기반 색인 : $(a+\lceil r_c*s_{e1}/s \rceil + a_t*(a+1))*t$,

    (단 $a_t=\begin{cases} r_c & \text{if } r_c<1000 \\ 1000 & otherwise \end{cases}$ 이다.)
  - 제안한 모델의 색인 : $(2a+1+\lceil r_c*s_t/s \rceil)*t$
- 구조 검색 (형제, 자식 검색)
  - K-ary기반 색인 : $(a+\lceil r_c*s_{e1}/s \rceil + a_t*(a+1) + r_c)*t$,

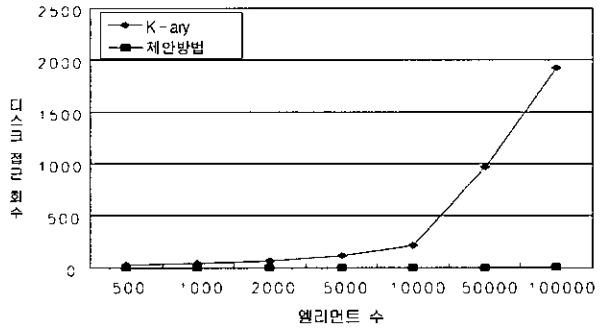
    (단 $a_t=\begin{cases} r_c & \text{if } r_c<1000 \\ 1000 & otherwise \end{cases}$ 이다.)
  - 제안한 모델의 색인 : $(2a+1+\lceil r_c*s_t/s \rceil)*t$
- 혼합 검색 (내용+구조 검색)
  - K-ary기반 색인 : $(2a+1+\lceil r_c*s_{k1}/s \rceil + a_t*(a+1))*t$,

    (단 $a_t=\begin{cases} r_c & \text{if } r_c<1000 \\ 1000 & otherwise \end{cases}$ 이다.)
  - 제안한 모델의 색인 : $(2a+1+\lceil r_c*s_{k2}/s \rceil)*t$
- 혼합 검색 (애트리뷰트+구조 검색)
  - K-ary기반 색인 : $(2a+1+\lceil r_c*s_{a1}/s \rceil)*t$
  - 제안한 모델의 색인 : $(2a+1+\lceil r_c*s_{a2}/s \rceil)*t$

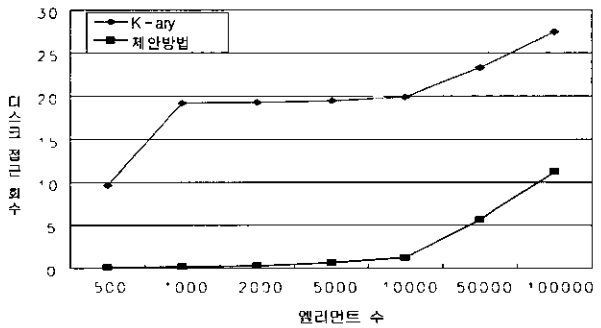
(그림 10)과 (그림 11)은 부모, 형제, 자손 검색에 대한 구조 검색 성능평가 결과이다. (그림 12)은 내용+구조 검색, (그림 13)은 구조+애트리뷰트 검색에 대한 검색 성능평가 결과이다. (그림 14)는 K-ary기반 색인 구조와 제안한 색인 모델의 색인 구조에 대한 내용 검색 성능평가 결과이다.



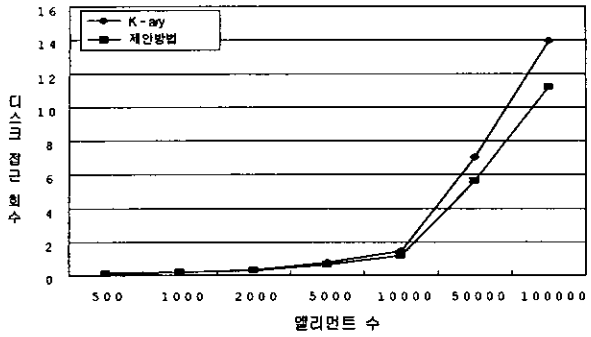
(그림 10) 구조 검색 (부모 검색)



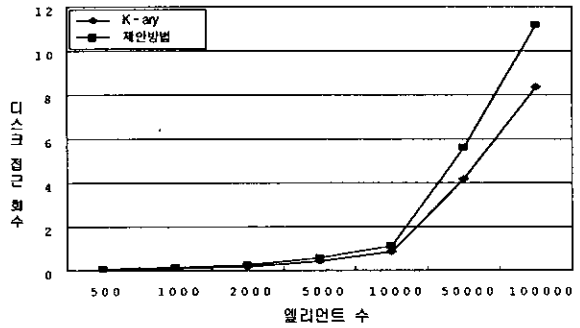
(그림 11) 구조 검색 (형제, 자식 검색)



(그림 12) 혼합 검색 (내용+구조 검색)



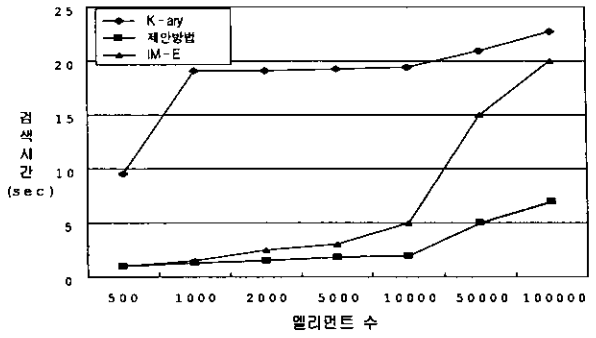
(그림 13) 혼합 검색 (구조+애트리뷰트 검색)



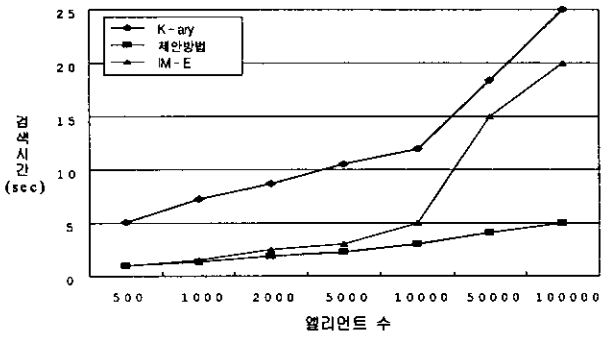
(그림 14) 내용 검색

5.2 실 험

실험적 성능평가 환경은 듀얼 CUP 168MHz, 메모리 320M의 SUN 울트라 2이고, 자바로 구현하였고, 사용한 DBMS는 오라클 8.0.1이다. 성능평가는 크게 내용 검색, 구조 검색, 혼합 검색의 3가지 항목으로 이루어진다. 성능평가 대상으로 K-ary 기반 색인모델과 엘리먼트 단위 구문 트리에 기반한 색인모델(이하 IM-E)[15]에 대해 실시하였다. IM-E에 대한 성능평가는 논문의 5절 성능평가 부분의 실제 데이터를 참고해서 비교평가를 수행하였다. (그림 15)과 (그림 16)은 부모, 형제, 자손 검색에 대한 구조 검색 성능평가 결과이다. 제안한 색인 모델의 색인에 비해 K-ary기반 색인이 구조 검색 오버헤드가 매우 많음을 알 수 있다. 이것은 K-ary기반 색인의 경우 부모, 형제, 자식의 Uid 계산에 이용되는 각 문서의 K값을 구조 색인에 저장하고 있어서, K값이 필요할 경우 매번 구조색인을 접근해야 때문이다. 특히, 형제나 자손 검색의 경우, K-ary기반 색인 특성상 존재하는 가상노드 여부를 구조 색인을 접근하여 확인하여야 하기 때문에 그 검색 시간은 엄청나게 늘어난다. 부모 검색에서 검색된 엘리먼트 수가 1000개일때 검색 시간이 완만해지는 이유는 검색 대상인 문서수가 1000개이므로 K값을 구하기 위해 구조 색인을 1000번 이상 접근할 필요가 없기 때문이다. 또한 IM-E보다 더 좋은 성능을 낸 이유는 구조 정보 표현방법에서 엘리먼트 표현을 디렉토리 구조로 하여 바로 부모 엘리먼트를 알 수 있기 때문이고, 또한 형제, 자식 경우도 마찬가지이다.

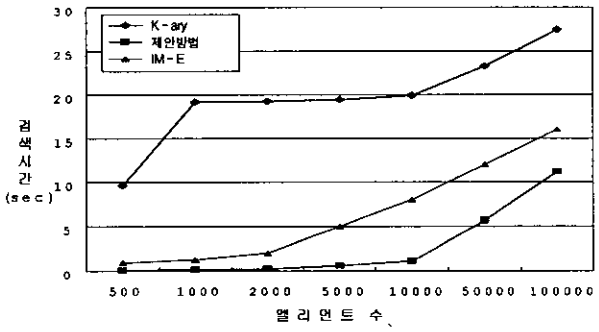


(그림 15) 구조 검색 (부모 검색)



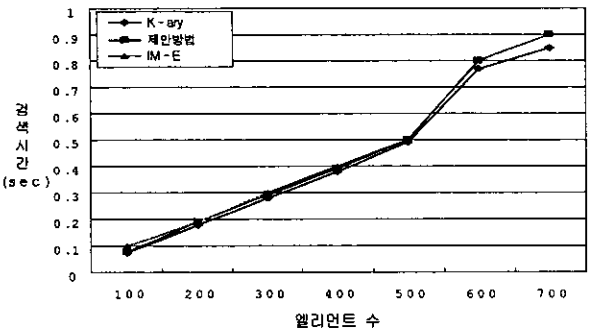
(그림 16) 구조 검색 (형제, 자식 검색)

다음 (그림 17)은 내용+구조 검색에 대한 검색 성능평가 결과이다. 혼합 검색에 포함된 구조 검색이 단순할 경우 두 색인 구조가 비슷한 검색 성능을 보였다. 그러나 K-ary기반 색인은 ["XML"을 포함하는 <단락>을 갖는 두 번째 <장(chapter)>을 검색해 주시오.]와 같이 부모, 형제, 자식 엘리먼트를 구해야 하는 경우 앞에서 보여준 구조 검색 결과와 비슷한 결과를 나타내었다.



(그림 17) 혼합 검색 (내용+구조 검색)

다음 (그림 18)는 K-ary기반 색인 구조, IM-E 색인구조와 제안한 색인 모델의 색인 구조에 대한 내용 검색 성능평가 결과이다. (그림 18)에서 많은 차이는 아니나 K-ary기반 색인 구조에 비해 성능이 떨어지는 것을 볼 수 있는데 그 이유는 K-ary기반 색인 구조에 비해 제안한 색인 모델의 내용 색인 포스팅의 크기가 크기 때문이다.



(그림 18) 내용 검색

5.3 비교 평가

앞 절에서 수행한 성능 평가의 전체적인 비교 결과를 <표 3>에서 나타낸다. 표에서 알 수 있듯이 전반적으로 제안한 방법이 기존의 방법보다 우수한 성능을 나타낸다. 그러나 제안한 방법이 기존의 방법보다 내용 검색에서 성능이 떨어지는 단점을 가진다. 그 이유는 제안하는 방법의 내용 색인 포스팅의 크기가 더 크기 때문에 검색이 시간이 더 걸리게 된다.

〈표 3〉 전체적인 성능평가 결과

		K-ary 기반 색인 모델	IM-E 색인모델	제안하는 색인 모델
평균검색시간	부모 검색	15.83초	4.16초	2.8초
	형제, 자식 검색	12.42초	4.16초	2.65초
	내용+구조 검색	19.71초	4.6초	2.74초
	내용 검색	0.43초	0.45초	0.45초

분석적 비용 모델의 정확성을 검증하기 위해 에러율을 다음과 같이 계산하였다. T는 이론적인 결과이고 E는 실험적인 결과이다.

에러율 = [최대값(T, E) - 최소값(T, E)]/최대값(T, E)

K-ary 기반 색인 구조와 제안한 색인 모델에 대한 내용 검색의 에러율은 0.5~10%이고, 혼합 검색의 에러율은 0.6~12%이고, 구조 검색의 에러율은 3~19%이다. 오차는 메모리 상에서 이루어지는 연산이 전체 검색 시간에 포함되어 에러율이 발생한 것이라 생각된다. 결과적으로 실험을 통한 성능평가와 분석적 비용모델 기반의 성능평가의 결과가 유사하게 나타남에 따라 분석적 비용모델의 정확성이 입증되어 향후 분석적 비용모델만을 사용하여 성능평가를 수행할 수 있는 계기를 마련하였다.

6. 결 론

본 논문에서는 XML 문서에 대해 효율적인 구조 검색을 지원하기 위한 색인 모델을 제안하였다. 제안한 색인 모델은 크게 두 부분으로 나뉘어 지는데 XML 문서의 구조정보를 표현하는 방법과 추출된 구조정보를 이용하여 검색을 위한 색인을 구성하는 것으로 이루어진다. 구조정보를 표현하기 위한 요소는 DTD에 나타나는 각각의 엘리먼트에 대해 유일한 값인 EID, 엘리먼트들 간의 계층정보를 나타내는 ETID, XML 문서에서 형제 엘리먼트의 순서정보인 SORD, 같은 엘리먼트 형에 대한 순서정보인 SSORD, 그리고 특정 엘리먼트가 가지는 자식 수인 Ccount로 구성된다. 이들 정보들은 효율적인 검색을 위한 색인을 설계하는데 사용된다. 색인 구조는 내용 검색을 지원하는 내용 색인, 구조 검색을 지원하기 위한 구조 색인, 애트리뷰트 검색을 지원하는 애트리뷰트 색인로 구성되며, 혼합 검색은 이들 색인의 결합으로 처리하게 된다. 이와 같이 제안한 색인 모델을 통해 특정 엘리먼트에 대한 직접적인 접근이 가능하며, 다양한 구조적 질의를 효과적으로 처리할 수 있다. 따라서 보다 효율적이고 빠른 검색을 지원할 수 있게 되었고 구조화된 문서관리 시스템 개발을 위한 기틀을 마련하였다.

또한 제안한 색인 모델을 통해 사용자 질의가 처리되는 과정을 보이고 K-ary 완전 트리 방법과 비교하여 성능 평가를 실시하였다. 성능 평가 결과 구조 검색과 혼합 검색에서 우수한 성능을 보였고 내용 검색은 비슷한 성능을 보였다.

향후 연구로써 보다 다양하고 많은 XML 문서에 대해 추가적인 성능 평가를 실시할 필요가 있고 문서에 갱신이 발생하는 동적인 환경에 제안한 색인 모델을 적용하기 위한 연구가 필요하다.

참 고 문 헌

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

[2] 연제원, 조정수, 이강찬, 이규철, "XML 문서 구조검색을 위한 저장 시스템 설계", 한국정보과학회 학술발표논문집(B), 제26권 제1호, pp.3-5 1999.

[3] 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.

[4] 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.

[5] Extensible Markup Language(XML) 1.0, "http : //www.w3.org/TR/1998/REC-xml-19980210".

[6] Jang, H. C., Kim, Y. I., Shin, D. W., "An Effective Mechanism for Index Update in Structured Documents," ACM, 1999.

[7] Lee, Y. K., Yoo, S. J., Yoon, K. R and Berra, P. B., "Index Structures for Structured Documents," Proc. Digital Library 96, pp.91-99, 1996.

[8] Kanemoto, H., Kato, H., Kinutani, H., Yoshikawa, M., "An efficiently updatable index scheme for structured documents," Database and Expert Systems Applications, pp.991-996, 1998.

[9] 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.

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

[11] 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.

[12] 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.

[13] Tuong Dao, "An Indexing Model for Structured Documents to Support Queries on Content, Structure and Attributes," Proceedings of ADL'98, pp.88-97, 1998.

[14] V. Christophides, S. Abiteboul, S. Cluet, M. Schol, "From Structured Documents to Novel Query Facilities," SIG MOD, pp.313-324, 1994.

[15] 한성근외 4명, "동적 환경에 적합한 SGML 인덱스 관리자의 설계 및 구현", 한국정보처리논문지, 제6권 제10호, 1999.

박 종 관

e-mail : parkck@disc.co.kr

1999년 충북대학교 컴퓨터공학과(공학사)

2001년 충북대학교 정보통신공학과 (공학석사)

2000년~현재 대우정보시스템(주)

관심분야 : 데이터베이스 시스템, XML, XML-EDI, 분산 객체 컴퓨팅

손 충 범

e-mail : cbson@trut.chungbuk.ac.kr

1997년 충북대학교 정보통신공학과(공학사)

1999년 충북대학교 정보통신공학과 (공학석사)

1999년~현재 충북대학교 정보통신공학과 박사과정

관심분야 : 데이터베이스 시스템, XML, 분산 객체 컴퓨팅

강 형 일

e-mail : khi69@jsc.ac.kr

1996년 목포대학교 전산통계학과(이학사)

1998년 목포대학교 전산통계학과(이학석사)

2001년 충북대학교 정보통신공학과 (공학박사)

2001년~현재 주성대학 멀티미디어정보통신공학부 전임강사

관심분야 : 멀티미디어 데이터베이스, XML, 정보검색 etc.

유 재 수

e-mail : yjs@cbucc.chungbuk.ac.kr

1989년 전북대학교 컴퓨터공학과(공학사)

1991년 한국과학기술원 전산학과(공학석사)

1995년 한국과학기술원 전산학과(공학박사)

1995년~1996년 목포대학교 전산통계학과 전임강사

1996년~현재 충북대학교 정보통신공학과 및 컴퓨터정보통신연구소 부교수

관심분야 : 데이터베이스 시스템, XML, 멀티미디어 데이터베이스, 분산객체 컴퓨팅, etc.

이 병 엽

e-mail : bylee@disk.co.kr

1991년 한국과학기술원 전산학과(공학사)

1993년 한국과학기술원 전산학과(공학석사)

1997년 한국과학기술원 경영정보공학 (공학박사)

1997년~현재 대우정보시스템 e-솔루션사업팀 차장

관심분야 : 데이터마이닝, XML, 분산 객체 컴퓨팅 분야 etc.