SGML/XML 검색 시스템의 설계 및 구현

고승규°, 조승기, 고견*, 최윤철
연세대학교 컴퓨터과학과
멀티미디어/그래픽스 연구실
*청주대학교 컴퓨터정보공학과

Design and Implementation of a SGML/XML Document Retrieval System

SeungKyu Ko, SeungKi Cho, Kyun-Koh*, YoonChul Choy
Department of Computer Science, Yonsei University
*Department of Computer and Information Engineering, Chongju University

요 약

이기종 간의 문서 교환 표준으로 제안된 SGML은 문서의 구조정보를 표현할 수 있는 장점으로 인해 CALS(Commerce At Light Speed), EC(Electronic Commerce), EDI(Electronic Data Interchange), 전자도서관(Digital Library) 등 여러 분야에서 사용되고 있다. 이렇게 SGML이 여러 분야에서 사용됨에 따라 많은 SGML 문서 중에서 원하는 문서를 효율적으로 찾아줄 수 있는 검색 시스템의 필요성이 증가하고 있다. 이에 본 연구실에서는 기본적인 구조 검색을 지원하는 SGML 문서 관리시스템을 기개발하였다[1]. 그러나 이 시스템은 구조 검색을 효과적으로 지원하지 못하기 때문에 본 연구에서는 구조 검색의 기능을 정의하고, 이를 지원하는 새로운 구조 질의어를 정의하였다. 또한 이러한 구조 검색을 효과적으로 지원하기 위한 구조 색인을 정의하였다. 그리고 구조 검색 방식으로 세가지 방식을 각각 구현 및 실험하여 그 중에서 성능이 뛰어난 질충식을 이용하여 검색 시스템을 구현하였다.

1. 서론

이기종간의 문서 교환 표준으로 제안된 SGML (Standard Generalized Markup Language)[7]은 문서의 논리적인 구조정보를 표현할 수 있는 장점으로 인하여 다양한 분야에서 널리 사용되고 있고, 근래에는 SGML의 간략화된 형태인 XML(eXtensible Markup Language)[8]이 인터넷 상에서 사용되고 있다. 이렇게 SGML이 여러 분야에서 사용됨에 따라 SGML 문서를 효과적으로 관리할 수 있는 관리 시스템과 효율적으로 검색할 수 있는 검색 시스템의 필요성이 증가하고 있다. 이에 본 연구실에서는 효율적으로 SGML문서를 관리하는 시스템을 이미 개발하였다[1]. 이 시스템에서는 간단한 구조검색을 지원하는 언어로 eXQL(eXtensible Query Language)을 제공하며 이에 따른 구조 검색을 수행한다. 그러나 이 eXQL은 구조검색의 일부 기능만을 비효율적으로 지원하고 있다. 그래서 본 연구에서는 일반적인 구조 검색 기능을 정의하고, 이를 효율적으로 지원하는 질의어로 eXQL+를 제안한다. 또한 효과적으로 구조 검색을 수행하기 위한 구조 색인을 정의하였다. 그리고 이러한 eXQL+와 색인을 지원하는 검색도구를 개발하였다. 이러한 검색 시스템은 기 개발된 SGML 문서 관리시스템 환경하에서 개발하였다.

본 논문의 구성은 다음과 같다. 2절에서는 SGML 문서에서 제공해야 하는 검색의 종류와 eXQL+에 대해 설명하고, 3절에서는 효율적인 구조 검색을 위한 구조 색인에 대해 설명한다. 4절에서는 구조검색을 수행하는 방법에 대해 소개하고, 마지막으로 5절에서는 결론 및 향후 연구 방향에 대해 기술한다.

* 본 연구는 1999년 산업자원부가 지원한 "SGML 문서처리 시스템" 과제의 연구 결과임

2. 구조 검색 질의어

2.1 구조 검색 기능

기존의 전문 검색(full-text) 시스템은 문서를 단어의 연속으로 간주하고 항상 문서 전체에 대하여 검색하기 때문에 검색 효율이 낮고 SGML 문서의

중요한 특징인 구조를 고려하지 않는다. 그래서 본 연구에서는 SGML의 논리적인 구조를 이용할 수 있는 구조 기반의 질의어로 eXQL+를 제안한다.

구조 문서에서의 검색은 문서의 구조를 이용하여 문서 전체가 아니라 사용자가 원하는 특정 문서 부분에 대한 처리가 가능해야 한다. 이러한 구조 검색의 기능은 여러 가지가 있을 수 있으나[4], 기본적으로 다음과 같은 3가지 검색 기능을 제공해야 한다.

- 검색 범위의 제한
  검색 대상을 문서가 아닌 문서를 구성하는 요소로 제한할 수 있다.
- 검색 결과의 제한
  검색의 범위 뿐만 아니라 검색 결과도 문서가 아니라 문서의 특정 부분으로 제한할 수 있다.
- 구조에 대한 검색
  구조 정보에 대한 검색이 가능하다. 예를 들어 'book'이라는 엘리먼트(element)의 자식 엘리먼트인 'title', 또는 'chapter'라는 엘리먼트의 자식 엘리먼트인 'title'을 찾는 검색이 가능하다. 의미상으로 앞의 예는 책의 제목을 찾는 것이고, 뒤의 예는 장의 제목을 찾는 검색이다. 이 예와 같이 같은 엘리먼트라고 하여도 위치에 따라 표현하는 정보가 달라지게 된다. 그래서 구조를 이용하여 검색을 수행할 경우에 사용자는 원하는 정보를 좀 더 명확하게 지정할 수 있다.

이러한 분류도 각각의 기능을 어느 정도까지 지원할 것인가에 따라 좀 더 세부적으로 분류 가능하다. 예를 들어 구조에 대한 검색에서 특정 엘리먼트를 지정할 경우에 항상 최상위 문서 요소(root)로부터 시작하는 절대경로만을 지원할 것인지, 임의의 문서 요소로부터 시작하는 경로를 지원할 것인지, 또한 경로에서 어떠한 관계를 지원할 지 등이 될 수 있다. 본 eXQL+ 에서는 상대경로를 지원하며, 경로 관계는 부모-자식, 조상-손자 관계를 지원한다.

또한 SGML에서는 이러한 기본적인 검색 이외에 속성(attribute)이나 링크(link)에 대한 검색도 지원해야 한다.

2.2 eXQL+

eXQL+는 기존의 eXQL[1]에 비해 검색 결과를 제한하는 기능과 사용자가 원하는 엘리먼트를 선택할 수 있는 기능을 제공한다. 제안된 질의 언어인 eXQL+는 2.1에서 정의한 검색 기능을 모두 지원한다.

<그림 1>은 eXQL+의 문법을 BNF(Backaus-Naur Form) 형식으로 표현한 것이다.

eXQL의 전체적인 구조는 from절과 select절로 구성된다. From절에서는 검색이 수행될 문서의 종류를 지정하는 <DTD_LIST>가 온다. 그리고 select절은 검색결과를 지정하는 부분인 *<select_element_list>*와 실제의 검색 조건 부분인 <component_statements> 부분이 오게 된다. *<select_element_list>*에서는 검색 조건을 중심으로 어떠한 엘리먼트를 검색할 것인지를 지정하는 부분으로 엘리먼트가 오게 되고 <component_statements>부분은 실제 검색 조건이 오는 부분이다.

```
<query_expression> ::=
      FROM [ <DTD_LIST>]
      SELECT ( <select_element_list >) /
      SELECT ( <select_element_list >)
      [ <component_statements>] /
      SELECT [ <component_statements>]
<DTD_LIST> ::=
      <DTD> | <DTD> <DTD_LIST>
<select_element_list > ::= <element_names>
<component_statements>                    ::=
      <component_statement>|
      <component_statement> <op>
      <component_statements>
<component_statement> ::= <pElement>|
      <pElement>% <query_terms>|
      <pElement>( <attribute_conditions>)|
      <pElement> ( <attribute_conditions> ) %
      <query_terms>
<element_names> ::=tag_name |
      tag_name <element_name>
<pElement> ::=tag_name |
      tag_name <path_op> <pElement>
<path_op> ::= .|..
<query_terms> ::= "query_term" /
      ( <query_terms>) /
      "query_term" <op> <query_terms>
<op> ::= & /'|' / and / or
<attribute_conditions> ::=
      <attribute_condition>/
      <attribute_condition> <op>
      <attribute_conditions>
<attribute_condition> ::= @ <attribute_name> /
      @ <attribute_name>= <query_terms>
<attribute_name> ::=attribute_name
```

<그림 1> eXQL+의 BNF 형태

eXQL+ 의 예는 다음과 같다.

- from [patent2] select (catalog) [catalog = "중공업" & "부산"]

위의 질의식은 "중공업"과 "부산"이라는 단어를 포함하고 있는 'catalog' 엘리먼트를 찾는 식이다. 이러한 질의식을 정의할 때 고려해야 할 사항으로 2가지가 있다.

- 선택할 엘리먼트와 조건식 엘리먼트와의 관계 위 질의식의 예에서 선택할 'catalog'라는 엘리먼트와 조건식에서 사용된 'catalog'라는 엘리먼트와의 관계를 의미한다.
- 다른 엘리먼트간의 'AND'나 'OR' 연산 처리

eXQL+ 에서는 조건식에 사용된 엘리먼트가 선택식에서도 나오게 되면 조건식의 결과에서 엘리먼트를 선택한다. 즉 검색이 엘리먼트 수준에서 수행된다. 또한 모호성을 제거하기 위해서 다른 엘리먼트 간의 'AND'나 'OR' 연산은 허용하지 않는다.

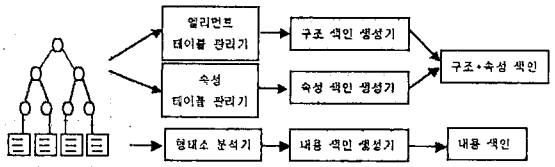
3. 구조 색인

대량의 문서에서 효율적인 검색을 지원하기 위해서는 색인이 필수적이다. SGML 문서는 논리적인 구조를 지니고 있기 때문에 구조 검색이 가능하다. 이러한 구조 검색을 효율적으로 수행하기 위해서는 색인에서도 이러한 구조 정보를 표현해야 한다. 기존의 데이터베이스의 색인은 문서와 같은 가변적인 데이터를 효과적으로 지원하지 못하고, 정보 검색 시스템의 색인은 구조 정보를 효과적으로 지원하지 못하고 있다. 그러므로 구조 정보와 문서 정보를 효과적으로 표현할 수 있는 새로운 색인 구조가 필요하다.

본 시스템의 색인은 eXQL+ 및 SGML/XML 문서에 대한 검색을 효과적으로 지원하기 위하여 구조와 내용에 대해 각각 색인을 구성한다.

구조에 대한 색인은 해쉬된 극대 트리(hashed maximun tree)라는 색인 구조를 이용한다. 극대 트리란 시스템 내의 모든 문서 구조를 표현할 수 있는 트리 구조를 의미한다. 그리고 각 엘리먼트를 검색할 경우 트리를 전부 순회하지 않고 해당 엘리먼트에 대해 바로 접근 가능하도록 엘리먼트 테이블에서 구조 색인에 대해 해쉬 테이블을 유지한다. 이 구조는 [4]의 DataGuides와 유사한 구조이지만 DataGuides는 그래프 중심이고, 절대경로만을 지원하는데 비해 본 색인은 트리 중심이고 엘리먼트 테이블을 이용하여 빠른 상대 경로를 지원한다.

내용 검색을 위한 색인은 앞의 해쉬된 극대 트리에서 구조 정보를 이용할 수 있으므로 실제 데이터를 가지고 있는 엘리먼트에 대해서만 생성한다. 또한 색인을 구성할 때 엘리먼트에 대한 효과적인 저장과 빠른 접근을 위하여 엘리먼트 테이블을 유지하고, 속성 이름에 대해서도 유사한 테이블을 유지한다. 본 시스템의 색인은 각 색인이 단독으로 사용되기 보다는 다른 색인들과 함께 사용되어 원하는 결과를 얻을 수 있도록 하여 저장과 검색 속도의 향상을 도모하였다. 전체적인 색인 생성의 흐름도는 <그림 2>와 같다.

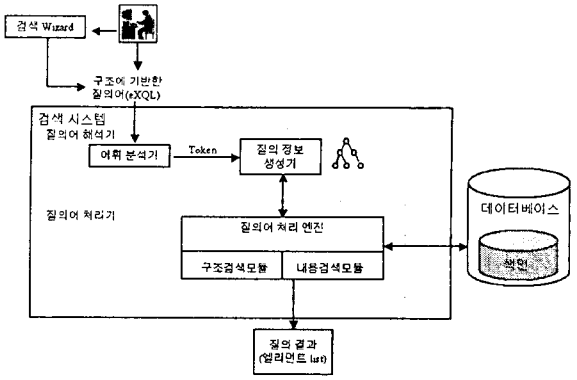


<그림 2> 색인 생성 흐름도

4. 검색 시스템

4.1. 개요

eXQL+의 형태로 들어온 질의식은 질의어 해석기에 의해 후위식 형태로 변환되며, 변환된 질의는 질의어 처리기로 전달된다. 질의어 처리기는 이 질의를 기본 단위별로 검색한 후 각각의 결과를 저장한 후, 이 결과들에 대해 'AND' 나 'OR' 연산을 수행한다. 처리된 최종 결과는 데이터베이스 내 해당 엘리먼트의 리스트(OID)로서, 질의어 처리기를 선언하여 사용한 모듈로 전달된다.



<그림 3> 검색 시스템 구성도

실제 질의 처리는 3장에서 설명한 색인을 통하여 이루어지며, 색인 접근 방법에 따라 상향식, 하향식, 절충식의 세가지 검색 방식으로 수행 가능하다. 사용자가 질의어를 입력하여 처리되는 과정은 <그림 3>과 같다.

4.2 검색 방식

질의에 맞는 검색결과를 찾아내는 검색 수행 방법은 구조를 먼저 검사할 것인지 내용을 먼저 검사할 것인지에 따라 다음의 세가지로 나눌 수 있다.

(1) 하향식 방법(Top-down)

구조를 먼저 검사하여 해당하는 엘리먼트를 얻은 후 각각의 엘리먼트에 대해 내용 검사를 한다. 구조를 검사할 경우에도 상위 구조를 먼저 찾을

수도 있고, 하위 구조를 먼저 찾을 수도 있다. 스탠포드 대학의 LORE시스템[4]은 절대경로만을 지원하지만 본 시스템은 특정 엘리먼트에 대해 직접적인 접근이 가능하다. 이 방법은 복잡한 구조를 검색하거나 구조의 변별력이 높을 경우에 유리하다.

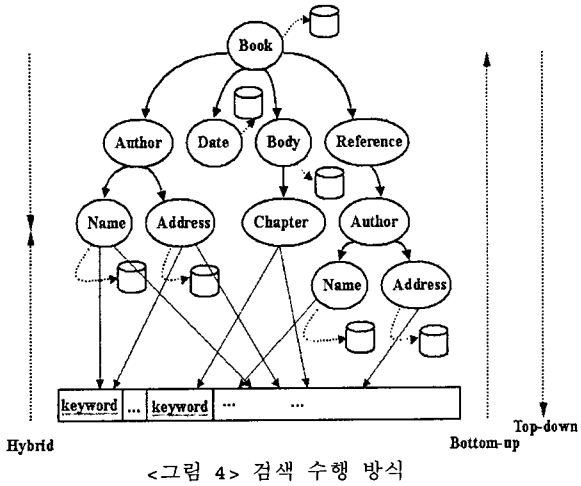
(2) 상향식 방법(Bottom-up)

하향식 방법과 반대로 내용을 먼저 검사한 후 구조를 검사하게 된다. 이 방법은 키워드의 변별력이 높을 경우에 유리하다.

(3) 절충식 방법(Hybrid)

이 방법은 위의 두 가지를 경우를 절충하는 방법으로 두 가지의 방식으로 나눌 수 있다. 하나는 질의를 수행하기 전에 비용을 예측하여 경우에 따라서 상향식이나 하향식 방법 중 하나를 선택하는 방식이다. 다른 하나는 상향식 방법을 수행한 후 결과가 될 수 있는 엘리먼트의 집합을 구하고, 하향식 방법을 수행하여 결과가 될 수 있는 엘리먼트의 집합을 구한 후 두 집합의 교집합을 구하는 방법이다. 이 방법을 이용하면 상향식과 하향식을 이용할 때의 최상의 결과를 얻을 수는 없지만, 마찬가지로 최악의 경우를 피할 수 있다.

본 시스템에서는 위 세가지 방법을 전부 구현하여 실험을 수행하였다. 그 결과 일반적으로 상향식 방식이 대부분 결과에 대해 성능이 좋았으나 특정 검색식에서 성능이 상당히 떨어졌고, 하향식 방식은 이와 반대로 대부분 결과에 대해 성능이 좋지 않았으나 특정 검색식에서 성능이 좋았다. 이에 반해 절충식 방식은 성능이 상향식과 유사하게 나오고, 상향식이 아주 나쁜 결과를 내는 검색식에서도 어느 정도의 성능을 보여주었다. 그래서 본 시스템은 위 세 방법 중 절충식 방식을 사용하여 검색을 수행한다.



<그림 4> 검색 수행 방식

5. 결론 및 향후 연구 방향

본 연구에서는 효율적인 구조 기반 검색을 위한 질의어로 eXQL+를 정의하고, eXQL+를 지원하는 검색시스템을 설계 및 구현하였다. 또한 효율적인 검색을 위해 구조 색인을 정의하였다.

본 시스템은 현재 절충식 방식만을 지원하고 있는데, 앞으로 질의식에 따라 효과적인 검색 방식을 선택할 수 있도록 질의어 평가와 최적화 기능을 추가하여야 한다. 이를 위해서는 특정 질의식에서 어떤 방식이 제일 우수한지에 대한 성능 평가가 이루어져야 할 것이다. 또한 기존의 정보검색 분야에서 사용되는 랭킹(ranking) 기법을 적용하여 사용자에게 적합한 문서를 먼저 보여주는 등의 검색 성능을 향상시키는 방법이 필요하다.

[참고문헌]

[1] 고승규, 백승욱, 이경호, 최윤철, "구조문서 검색을 위한 질의어 및 검색 시스템의 설계 및 구현." 멀티미디어학회 가을 학술발표논문집, 한국멀티미디어학회, 1999

[2] 김현기, 노대식, 강현규, "DTD 의존 스키마에 기반한 SGML 문서 저장 시스템 개발에 관한 연구," 정보처리학회 논문지 제 6권 5호, 1999

[3] D. Shin, H.Jang, and H. Jin, "BUS:An Effective Indexing and Retrieval Scheme in Structured Documents", Proceedings of the 1998 ACM DL Conference, pp.235-243, Pittsburgh PA, USA, 1998

[4] R. Goldman and J. Widom, "DataGuides: Enabling query formulation and optimization in semistructured databases," In Proceedings of the Twenty-Third International Conference on Very Large Data Bases, pages 436-445, Athens, Greece, August 1997

[5] R. Sacks-Davis, T. Arnold-Moore and J. Zobel "Database Systems for Structured Documents," IEICE TRANS. INF. & SYST., Vol.E78 D. No. 11, pp.1335~1341,1995

[6] 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).

[7] ISO 8879:1986. Information Processing - Text and Office System - Standard Generalized Markup Language (SGML), Oct. 15 1986.

[8] XML 1.0 www.w3.org/TR/1998/REC-xml-19980210, February 1998. W3C Recommendation 10-February-1998