論文
2-6-3

XML DOM을 이용한 웹문서 검색 알고리즘 (Retrieval algorithm for Web Document using XML DOM)

김 노 환* 정 충 교**
(No-Whan Kim) (Choong-Kyo Jeong)

요 약

현재까지 웹 검색엔진은 각 문서가 어떤 키워드를 얼마나 갖고 있는지, 키워드의 빈도수에 따라서, 문서에 키워드를 많이 포함하는 문서가 가까운 문서라는 가정에 의거 문서 순위를 사용자에게 보여주는 형태였다. 이런 형태의 검색은 HTML 웹 데이터처럼 구조적인 정보를 포함하지 않은 일반 문서형태의 경우 키워드의 발생 빈도를 고려하는 형태에서는 별 문제가 없지만 구조적인 정보를 갖고 있는 XML로 표현된 웹 데이터일 경우에는 그래프 형태의 모델표현이 가능하기 때문에 단순히 키워드의 빈도만을 고려하는 형태로서는 바른 검색결과를 얻을 수 없다.

따라서 XML 문서의 구조적인 특성을 최대한 활용하여 SQL과 유사한 형태의 질의를 통하여 원하는 데이터만을 추출한다면 단순히 키워드에 의존하는 형태의 질의를 탈피하며 보다 분명한 검색결과를 획득할 수 있다고 생각한다. 본 논문에서는 XML DOM을 이용하여 XML 데이터의 정보검색 시스템을 모델링하고, 이와 관련된 알고리즘을 제안하고자 한다.

ABSTRACT

Until recently Web retrieval engine has presented a demanded document to users according to the amount and the frequency of inquired key words in each document under the assumption that the more key words a document has, the more accessible it is. This method of searching doesn't matter to a normal document such as HTML Web data in which structural information is not involved. However, Web data realized in XML contains structural information and modeling of graphic forms is also available.

Therefore, in the case of XML, this method leads to no less trouble since it depends only on the frequency of key words. We consider that this problem can be resolved by way of inquiry which is similar to SQL. This form of inquiry enables us to snatch an exact data we want in a quick and clear way with a full advantage of structural quality of XML, overcoming the shortcomings of frequency-based engine. In this paper, We aim to design a model of information retrieval system of XML data using XML DOM and consider its algorithm related with it.

* 정회원 : 동우대학 인터넷통신과 조교수
** 정회원 : 강원대학교 전기전자 · 정보통신공학부 부교수

논문접수 : 2001. 5. 28.
심사완료 : 2001. 6. 9.

1. 서론

XML(eXtensible Markup Language)은 인터넷상의 데이터 표현과 교환을 위한 새로운 표준으로서[1], 데이터 엘리먼트의 태그는 데이터를 어떻게 포맷이 되도록 할 것인가 보다는 데이터의 의미를 식별하며, 데이터 엘리먼트 사이의 관계, 네스팅(Nesting) 그리고 참조(idref)를 제공하는 등 기본 개념은 의외로 단순하다. XML은 웹 상에서 구조화된 문서가 전송이 가능하도록 설계된 SGML 기반의 표준화된 텍스트 형식으로, 웹서버와 XML 데이터 포맷이 어플리케이션 간에 정보교환을 편리하게 한다.

XML 파일의 내용은 XSL(eXtensible Stylesheet Language[2].)에 의해 별개로 렌더링되므로 동일 데이터에 대해 다양한 뷰를 제공할 수 있다. XML은 간단히 쉽게 파싱할 수 있고, 자기서술적 특성을 가지므로, XML로 제작된 웹은 유용한 정보를 추출하기에 적합하다.

예를 들어, [그림 1]과 같은 HTML 문서를 고려해 보자.[3]

```
<UL>
<LI>
R.Goldman, J.McHugh, and J.Widom.
<A href="ftp://db.stanford.edu/pub/papers/xml.ps">
From Semistructured Data to XML : Migrating the Lore
Data Model and Query Language
</A>
Proceedings of the 2nd International Workshop on the
Web and Databases
(WebDB '99), pages 25-30, Philadelphia,
Pennsylvania, June 1999.
</LI>
T. Lahiri, S. Abiteboul, and J. Widom.
<A
href="ftp://db.stanford.edu/pub/papers/ozone.ps">
Ozone: Integrating Structured and Semistructured Data
</A>
Technical Report, Stanford Database Group, October
1998.
</UL>
```

[그림 1] HTML을 이용한 문서 표현

[Fig. 1] Document Expression using HTML

XML에서는 같은 정보를 [그림 2]와 같이 표현한다.

```
<Publication
URL="ftp://db.stanford.edu/papers/xml.ps"
Authors="RG JM JW">
   <Title>From Semistructured Data to XML : Migrating
the Lore Data Model and Query Language </Title>
   <Published>Proceedings of the 2nd International
   Workshop on the Web and Database (WebDB '99)
         </Published>
   <Pages>25-30</Pages>
   <Location>
      <City>Philadelphia</City>
      <State>Pennsylvania</State>
   </Location>
   <Date>
      <Month>June</Month>
      <Year>1999</Year>
   </Date>
</Publication>
<Publication
URL="ftp://db.stanford.edu/pub/papers/ozone.ps"
Authors="TL SA JW">
    <Title>Ozone: Integrating Structured and
    Semistructured Data</Title>
    <Published>Technical Report</Published>
    <Institution>Stanford University Database
     Group</Institution>
    <Date>
       <Month>October</Month>
       <Year>1998</Year>
    </Date
</Publication>
<Author ID="SA">S. Abiteboul</Author>
<Author ID="RG">R. Goldman</Author>
<Author ID="TL">T. Lahiri</Author>
<Author ID="JM">J. McHugh</Author>
<Author ID="JW">J. Widom</Author>
```

[그림 2] XML을 이용한 구조적인 문서 표현

[Fig. 2] Structured Document Expression using XML

XML의 문서표현이 비록 서술적이긴 하지만, 데이터관리 측면에서 보면 보다 편리하며 유용한 포맷으로 정보를 제공한다. 현재 대부분의 웹 페이지가 HTML 및 자바스크립트를 이용하여 제작되고 있는데, 의미 있는 웹 페이지를 만든다는 측면에서 향후 많은 웹사이트가 XML로 제작될 것이다. 대부분의 사이트가 XML로 구축된다면 지금과 같은 검색 엔진은 유용한 검색 결과를 추출하지 못하게 된다. 왜냐하면 현재 대부분의 웹 검색엔진이 사용하는 검색방법은 웹 페이지에서 검색 키워드가 얼마나 많은 횟수 발견되느냐로 원하는 웹 페이지인지 아닌지를 결정한다. 하지만 완전한 구조는 아니지만 구조적인 특성을 갖는 반구조적(semi-structured)인 XML 문서에서 키워드의 빈도수는 그렇게 중요하지 않다.

웹 상의 XML 문서에서 데이터를 효율적으로 추출하기 위한 질의어로는, XSLT와 XPointer기반의 XML 문서 요소들에 대한 구조적인 Addressing 기법을 제공하며 XML 문서의 특정 부분 접근을 주된 목적으로 하는 XPath, DBMS 언어 중 가장 널리 사용되는 언어인 SQL, SQL의 확장판 언어로서 질의·구조·변환, 그리고 XML 데이터의 통합 및 새로운 XML data의 생성을 지원하는 XML-QL, XSL의 패턴정의 부분을 확장한 언어로서 XSL에 비해서 훨씬 간단하게 문장을 구성할 수 있도록 제안된 XQL 등이 있다.

이들 언어들은 XML에 접근하여 구조적인 질의를 통하여 우리가 원하는 정보를 문서에서 얻을 수 있다. 하지만 [그림 1]과 같은 HTML 문서로 된 웹문서는 구조적인 형태가 없으므로 단지 태그 정보를 모두 제거한 다음, 필요한 키워드가 얼마나 많은 빈도수를 갖는지 횟수에만 관심이 있다.

본 논문의 2장에서는 관련분야에 대한 특성과 표현방법에 대해 기술하고, 3장은 제안된 알고리즘에 대해 설명한다. 4장은 제안된 알고리즘의 구현 결과를 도출하였고, 5장은 결론 및 향후 계획을 제시한다.

2. 관련연구

2.1 XSL

XML 문서에서 출력을 담당하는 XSL은 자체적으로 표현능력이 없는 XML을 HTML처럼 보여주기 위한 마크업 언어로서, XSL로 생성한 문서는 XSL 스타일 시트(StyleSheet)라고 한다.

XSL을 위한 데이터 모델은 순서 트리(ordered tree)이며, XSL 패턴은 템플릿 룰의 집합이다. 각 룰은 패턴과 템플릿으로 구성된다. XSL은 루트 엘리먼트로부터 시작하고 이 노드에 패턴을 적용하기 시작한다. 만일 이것이 성공되면, 일치하는 템플릿을 실행한다.

[그림 3]과 같은 XML 도서목록 문서를 검토해 보자.

```
<bib>
    <book>
        <title>t1</title>
        <author>a1</author>
    </book>
    <paper>
        <title>t2</title>
        <author>a2</author>
        <author>a3</author>
    </paper>
</bib>
```

[그림 3] 간단한 도서목록 문서

[Fig. 3] Document For Simple Publication List

[그림 4]의 XSL 프로그램은 XML 문서 내의 모든 제목을 웹상에 출력한다.

```
<xsl:template>
   <xsl:apply-templates/>
</xsl:template>
<xsl:template match="/bib/*/title">
    <result>
       <xsl:value-of/>
    </result>
</xsl:template>
```

[그림 4] 제목추출을 위한 XSL 프로그램

[Fig. 4] XSL Program for "title" Retrieval

<xsl:template>은 title 태그의 모든 데이터를 메모리로 읽어오며, 템플릿 룰을 구성한다. match 속성은 패턴을 정의하므로 임의의 경로에 존재하는 title 엘리먼트를 만나면 우리가 원하는 패턴이 되고, 이것의 내용은 템플릿을 정의하게 된다. 그러나 일치하는 속성이 없다면 템플릿은 어떤 노드에 일치할지 알 수 없다.

<result><xsl:value-of/></result>는 템플릿으로서, <result> 엘리먼트가 태그가 되고 <xsl:value-of>에 해당하는 엘리먼트의 값을 가져오므로, 다음과 같은 결과를 얻는다.

```
<result>t1</result>
<result>t2</result>
```

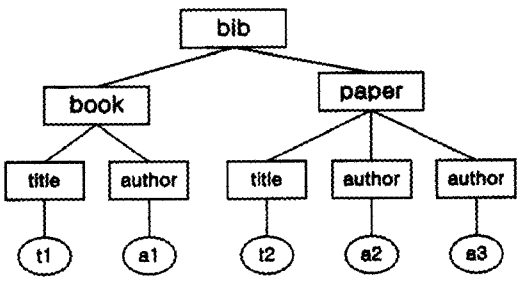
2.2 API

XML 문서에는 두 개의 다른 API(Application Programming Interface)가 있다. DOM(Document Object Model)은 문서를 접근하고 조작하기 위한 방법으로 문서의 논리적 구조를 정의하는 반면에 SAX(Simple API for XML)는 태그를 탐지해서 문서를 다루는 구문 추론형태이다.

2.2.1 DOM

DOM은 XML 문서를 위한 API로서, XML 문서를 접근하고 조작하기 위한 방법으로 문서의 논리적 구조를 정의한다. 따라서 DOM을 이용하여 사용자들은 문서를 생성하고 그 문서의 구조에 따라 항해(navigation)하고, 엘리먼트와 문서 내용을 추가/수정/삭제할 수 있다. 현재까지는 문서의 접근이란 문서 전체를 의미했는데 표준 DOM 인터페이스를 사용함으로써 문서 전체가 아닌 문서의 일부분에 대한 접근도 가능해졌다.[4]

DOM은 문서 전체를 파싱 처리한 다음에 문서를 위한 문서 트리를 구성한다[5]. [그림 5]는 [그림 3]의 구조적인 XML 문서 내에 정의된 엘리먼트들을 핸드링하기 위하여 각각 하나의 객체(Object)로 모델링(modeling)한 후, DOM을 위한 트리(tree)구조로 구조화한 모습이다.



[그림 5] DOM 트리-도서목록

[Fig. 5] DOM tree - Publication List

DOM은 노드 인터페이스, 서브클래스 엘리먼트, 속성, 문자데이타 인터페이스를 노드로부터 상속받아 정의한다.

노드 인터페이스는 가장 기본적인 타입으로써 노드의 컴포넌트에 접근하기 위한 메소드를 정의하는데, 부모노드{parentNode()}와 정해지지 않은 순서적 자식노드{ChildNodes()}를 가질 수 있다.

또한, 새로운 노드를 만들기 위해 노드 값을 변경하기 위한 메소드를 제공한다. 따라서, 어플리케이션은 이 인터페이스를 이용하여 문서를 재구성할 수 있으므로, DOM은 XML에 대한 데이터 뷰라고 할 수도 있다. 즉 XML 문서의 특정 엘리먼트의 자식엘리먼트의 텍스트 내용을 알고 싶다면, DOM 인터페이스를 이용하면 된다. DOM은 시스템 리소스를 많이 차지하므로 XML 문서의 파일크기가 크면 문제가 될 수 있지만, 노드의 생성, 삭제 등 풍부한 API 제공이 가능하다.

2.2.2 SAX

SAX는 XML 문서를 파싱하기 위한 API로서 DOM 대신 사용할 수 있다.

SAX는 이벤트 중심의 인터페이스이므로 handler에 의해 startDocument, endDocument, startElement, endElement, characters 등의 이벤트를 설정해 놓으면, SAX 파서는 XML 데이터 스트림을 읽어 파싱하면서 태그를 인터프리트할 때 이벤트를 탐지하여 해당 이벤트가 발생하면 SAX가 제어권을 가지고 상황을 처리하여 어플리케이션에 전달한다. 참고로, SAX는 시스템 리소스를 적게 차지하고 속도가 빠른 반면, 엘리먼트의 생성, 삭제 등을 지원하는 API가 없다.

2.3 XML에서 질의처리를 위한 경로표현

반 구조적인 데이터는 일반적으로 라벨이 있는 그래프로 표현되어 있다. 그래프의 노드는 객체(entity)에 대응되고, 간선은 그 객체의 속성(attribute)에 대응된다. 간선-라벨 $l_1, l_2, \ldots l_n$은 경로의 표현이라고 한다. 경로의 표현을 간단한 질의로서 볼 수 있고 이 질의 결과 노드의 집합을 얻을 수 있다[6].

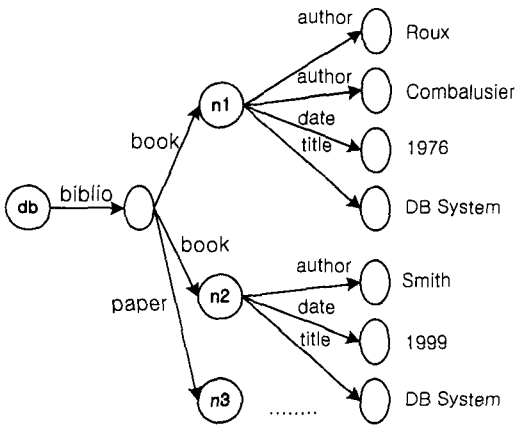
예를 들어, [그림 6]에서 경로표현 "biblio.book"의 결과는 노드 {n1, n2}의 집합이다. 경로 표현 "biblio.book.author"의 결과는 문자열의 집합 {"Combalusier", "Roux", "Smith"}와 연관된 노드의 집합이다.

일반적으로, 데이터 그래프상의 경로 표현 $l_1, l_2, \ldots l_n$의 결과는 노드 v_n의 집합이 존재한다. 그렇기 때문에 간선 (r, l_1, v_1), (v_1, l_2, v_2), …, (v_{n-1}, l_n, v_n)이 데이터 그래프에 존재한다. (r, l_1, v_1)에서 r은 루트, l은 라벨, v는 노드로서, 루트 노드 r에서 노드 v1에 이르는 라벨 l1이 존재한다는 의미이다. 따라서 경로표현은 노드의 집합을 얻을 수 있다.

본 논문에서 데이터 그래프 DB를 다음과 같이 정의한다. 먼저 O를 무한한 객체 식별자의 집합, C를 O와 교집합을 가지지 않은 무한한 상수의 집합이라 가정한다.

(정의 1) 데이터 그래프 DB=(V,E,R)는 루트 노드를 가진 그래프로서 V⊆O는 노드의 집합, E⊆V×C×V는 방향성 있는 간선(edge)의 집합, 그리고 R∈V는 루트 노드를 의미한다.

XML 데이터는 XML의 엘리먼트를 노드 V에, 엘리먼트-속성 관계와 엘리먼트-부엘리먼트 관계, 엘리먼트간 참조 관계를 간선 E에 맵핑하고 XML 문서의 값을 그래프의 단말노드에 맵핑하면 비정형 데이터 모델에 맵핑될 수 있다.



[그림 6] 그래프 형태로 표현한 데이터베이스

[Fig. 6] Database Expressed by Graph Model

이때, 그래프 모델은 엘리먼트-부엘리먼트 관계와 엘리먼트 간의 참조(idref)를 구분하지 않고, 속성과 부 엘리먼트 관계를 구분하지 않으므로 XML 문서의 특정 정보를 잃어버릴 수 있지만 간단한 데이터 구조를 첨가함으로써 XML 데이터를 그래프 모델로 맵핑할 수 있다.

이러한 DB 그래프에서 질의는 다음의 정규 경로식(regular path expression)을 기반으로 한다.

(정의2) 정규 경로식

$r = \epsilon | a | _ | (r_1.r_2) | (r_1|r_2) | r^*$으로 표현한다. 여기서 r, r_1, r_2는 정규 경로식, a∈C는 특정 상수, _은 임의의 레이블, ϵ은 빈 문자열(empty string), *는 반복을 나타낸다.

이러한 정규 경로식을 기반으로 하는 질의 언어로는 *UnQL[7]*, *Lore[8]*, *XML-QL[9]*, $STRUQL_0$[10], 그리고 마이크로소프트의 XQL 등이 제안되었다.

본 논문에서는 이미 제안된 위의 질의 언어를 사용하지 않고 이미 우리에게 친숙한 관계형 질의 언어인 SQL를 사용하여 질의하고 질의 결과를 검토한 후 향후에 XPath나 XQL과 같은 웹 상에서 편리하게 사용할 수 있는 질의 형태로의 변형된 언어를 개발하여 일반 사용자가 손쉽게 이해할 수 있는 언어를 구현하고자 한다.

3. 제안 방법

인터넷상에서 기존 검색시스템은 서론에서 언급한 바와 같이 아직까지는 웹 문서가 대부분 HTML 문서로 되어 있어서, 웹 페이지마다 갖고 있는 문서에 대해 사용자가 원하는 키워드를 얼마나 많이 갖고 있는지, 즉 키워드의 빈도 수에 의존하는 형태의 질의를 하고 있다. 이 경우 데이터베이스와 같은 형태의 질의 표현은 사실상 불가능하여 우리가 원하는 검색결과를 얻지 못하므로 사용자가 매번 검색된 페이지를 자세히 살피지 않으면 안되는 문제점을 안고 있다.

본 논문에서는 웹 페이지가 구조적 형태인 XML로 되어 있다는 가정 하에서, XML DOM을 이용하여 질의를 처리하는 알고리즘에 대한 구현 방법을 제안하고자 한다.

문서의 내부 구조에 접근해서 XML 문서를 처리

하려면 API를 사용할 필요가 있는데, XML 1.0 권고안은 파일을 읽고 출력할 때 XML 프로세서의 정확한 행동을 정의하고 있지만, 어느 API를 사용해야 한다는 것은 언급하고 있지 않다.

DOM은 DOM 트리 구조를 접근하고 다루는 API를 제공한다. DOM이 XML 문서를 읽어서 트리 구조로 주기억장치에 갖고 있으며, 이때 각각의 노드는 엘리먼트, 텍스트가 되며, DOM을 기반으로 하는 XML 프로세서는 DOM을 이용하여 구문 트리를 만들고 전체 문서를 다루기 위한 어플리케이션 프로그램을 다룬다.

따라서, DOM은 XML 문서를 구조적으로 변경하거나 메모리에 있는 문서를 다른 어플리케이션과 공유할 때 적절하게 사용할 수 있다.

XML을 위한 웹 검색 시스템에서는 현재 RDB에 저장되지 않는 XML 문서를 대상으로 한다. 즉 XML 문서가 파일 단위로 디스크에 저장된 것으로 가정하고 원하는 파일을 검색하게 된다.

SQL이 작동하기 위해서는 RDB처럼 DBMS의 내부 구조가 DDL · DML 처리, Query Optimization 등의 구성요소로 되어 있어야 하며, DBMS 내부의 DB에 자료를 저장해야만 한다.

제안된 알고리즘은 [그림 7]과 같다.

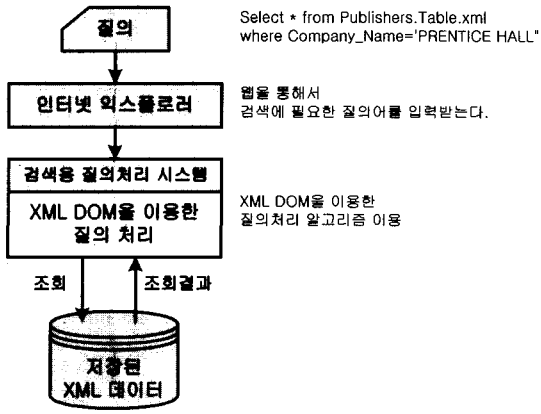
- 질의에서 문자열 "from"을 찾는다.
- "from" 문자열 다음의 문자열이 질의처리에 이용되는 파일이름이 된다.
- "where" 문자열 다음의 문자열을 추출하여 조건절로 이용한다. 조건을 만족하는 엘리먼트 또는 속성을 찾으면 "select"절 다음을 문자열을 이용하여 다음과 같이 엘리먼트를 추출한다.
- 문자열 "select" 이하의 문자열을 추출한다.
- 추출된 문자열을 getElementsByTagName을 이용하여 XML 문서에서 추출한다.
- 추출된 결과를 사용자에게 보여준다.

[그림 7] 제안된 알고리즘

[Fig. 7] Proposed Algorithm

본 논문에서는 검색 입력창을 통해 SQL 질의어가 입력되면, 현재 시스템처럼 DBMS에 의해 동작하는 것이 아니고 XML DOM에 의해서 문서의 여러 부분을 직접 참조할 수 있도록 하여, SQL이 필요로 하는 부분에 대한 정보를 제공할 수 있도록, 비주얼베이직을 이용하여 알고리즘을 구현하였다.

[그림 8]은 이러한 질의처리 알고리즘의 시스템 구성도이다.



[그림 8] 제안된 질의처리 구성도

[Fig. 8] Configuration of Proposed Query Manipulation

제안된 알고리즘은 XML DOM에 의해 구성된 구조정보를 SQL 질의방법으로 질의함으로써 XML 문서에서 키워드에 일치하는 정보를 찾아서 조건에 맞는 정보를 정확히 추출할 수 있었다.

한편, 웹 상에서 [그림 9]의 ①과 같은 SQL 질의어는 기존의 데이터베이스를 사용하고 있는 전문가가 아니면 조금은 어려운 질의어이므로, 향후 인터넷의 표준 질의어로서 많은 사람들이 이용할 것으로 기대되는 ②와 같은 XPATH 형태로 질의하도록 하여 SQL을 모르는 사용자도 간단한 지식만으로 편리한 질의를 할 수 있도록, 향후 XPATH를 이용한 질의 처리 프로세서를 개발할 예정이다.

이러한 질의어를 검색창에 직접 입력하면 XML DOM의 도움으로 XML 파일에서 원하는 자료를 검색하여 보여주게 된다. 참고로, 이 질의어는 출판사명이 "PRENTICE HALL"인 것의 모든 엘리먼트를 보이라는 명령문이다.[11]

```
select *
from Publishers.xml
where Company_Name="PRENTICE HALL"          ---①

Publishers[Company_name="PRENTICE HALL"]    ---②
```

[그림 9] SQL 및 XPATH 질의어

[Fig. 9] Query Language of SQL & XPath

4. 구현

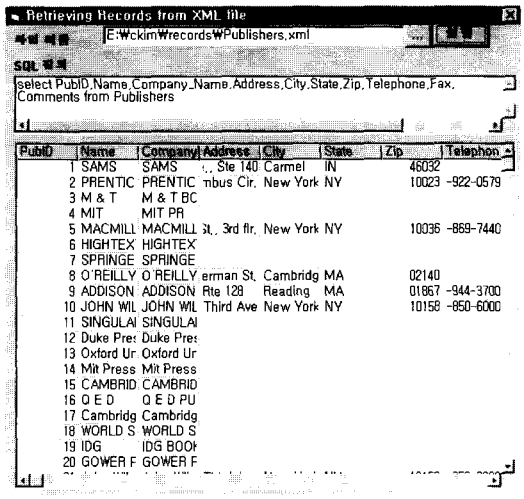
제안된 알고리즘을 구현하기 위하여 [그림 10]과 같은 XML 문서를 대상으로 질의 처리를 하였다.

```
<Publishers.table>
   <Publishers>
         <PublD> 1</PublD>
         <Name> SAMS</Name>
         <Company_Name> SAMS</Company_Name>
         <Address> 11711 N. College Ave., Ste 140
        </Address>
         <City> Carmel</City>
         <State> IN</State>
         <Zip> 46032</Zip>
         <Telephone> </Telephone>
        <Fax> </Fax>
        <Comments> </Comments>
   </Publishers>
   <Publishers>
         <PublD> 2</PublD>
         <Name> PRENTICE HALL</Name>
         <Company_Name> PRENTICE HALL
        </Company_Name>
         <Address> 15 Columbus Cir.</Address>
         <City> New York</City>
         <State> NY</State>
         <Zip> 10023</Zip>
         <Telephone> 800-922-0579</Telephone>
         <Fax> </Fax>
         <Comments> </Comments>
   </Publishers>
    ⋮
```

[그림 10] Publishers.xml 문서의 일부

[Fig. 10] A Portion of Document(Publishers.xml)

[그림 11]은 XML 문서로부터 VB상에서 SQL 질의어로 필요한 정보를 추출한 처리결과이다. 이 와 같은 구현을 위해서 XML DOM을 사용하였다. 본 논문에서는 MS사에서 제공하는 MSXML3.0 버전을 VB의 라이브러리로 이용하여 프로그래밍 작업을 하였는데, 참고로 MSXML은 다섯 개의 새로운 인터페이스와 COM 객체를 지원하고 있다.



[그림 11] SQL 질의어와 처리 결과

[Fig. 11] SQL Query Language & Manipulation Result

5. 결론 및 향후 계획

웹 문서에서 우리가 원하는 정보를 빠르고 정확하게 추출하는 것은 인터넷이 시작된 이래로 활발하게 연구되어온 주제이다. 하지만 인터넷 검색엔진을 통해서 유용한 데이터의 검색을 시도하는 경우 정확한 데이터만을 제공해주기 보다는 원하는 데이터의 수백 배에 해당하는 데이터를 결과로 보여주게 된다. 이 경우 한 두 페이지만을 눈으로 검색한 후, 결국 많은 페이지는 보지 않게 된다.

이러한 문제점을 해결하기 위해서, 웹 문서에서 XML의 필요성을 제기하고, 검색 방법을 개선하는 방안으로 본 논문에서는 XML DOM을 이용하여 XML 문서를 읽고 사용자의 질의에 합당한 결과 문서를 정확하고 신속하게 제공하기 위한 제안을 하였다. 본 시스템의 문제점은 웹에서 사용할 수 있는 형태가 아니라 SQL을 사용하는 형태로서, 이러한 단점을 해결하기 위해 향후 웹상에서 질의 및 실행할 수 있는 시스템을 개발할 예정이다.

※참고문헌

[1] The W3C's XML Web page, 1998, http:// www.w3c.org/XML.

[2] Extensible Stylesheet Language(XSL). http:// www.w3c.org/TR/WD-xsl, 1998.

[3] Jennifer Widom, "Data Management for XML : Research Directions", IEEE, 1999.

[4] http://dblab.ce.cnu.ac.kr/~dolphin/xml/atoz/dom.html

[5] The W3C's DOM web page, 1999. http://www.w3c.org/DOM/.

[6] Serge Abiteboul, Peter buneman, Dan Suciu, Data on the Web, Morgan Kaufmann Publishers, 1999

[7] Peter Buneman, Susan Davidson, Gerd Hillebrand, and Dan Suciu. A query language and optimization techniques for unstructured data. In Proceedings of the ACM SIGMOD International Conference on the Management of Data, 1996.

[8] S.Abiteboul, Dallan Quass, Jason McHugh, Jennifer Widom, Janet Wiener. The lore query language for semistructured data. International Journal on Digital Libraries, 1996.

[9] A. Deutsch, M. Fernadez, D. Florescu, A. Levy, and D. Suciu. Query Language for XML. In Proceedings of Eighth International World Wide Web Conference, 1999.

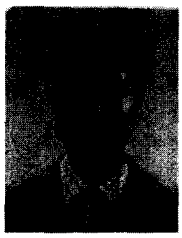
[10] Dana Florescu, Alon Levy, and Dan Suciu. *Query containment for conjunctive query with* regular expressions. In Proceedings of ACM Sympsium on Principles of Database Systems, 1998.

[11] James Clark, Steve DeRose, "XML Path Language (XPath) Version1.0", http://www.w3.org/TR/xpath

김 노 환

1978년 숭실대학교 전자공학과 (공학사)
1985년 연세대학교 대학원 (공학석사)
2000년 강원대학교 대학원 박사과정 수료
1980년 금성전기(주) 기술연구소
1983년 현대전자산업(주) 시스템연구소
1993년~현재 : 동우대학 인터넷통신과 조교수
관심분야 : XML, 인터넷응용

정 충 교



서울대학교 전기공학과 졸업 (공학사)
한국과학기술원(KAIST) 졸업 (공학석사)
한국과학기술원(KAIST) 졸업 (공학박사)
1989년 LG정보통신(주) 책임연구원
1995년~현재 : 강원대학교 부교수
관심분야 : network(TCP/IP, ATM)