韓國 컴퓨터情報學會 論文誌
第13卷 第6號, 2008. 11.

2008-13-6-1-9

웹 정보원 통합을 위한 내용 기반의 스키마 자동생성시스템

곽 준 영*, 배 종 민**

An Automatic Schema Generation System based on the Contents for Integrating Web Information Sources

Jun-Young Kwak*, Jong-Min Bae**

요 약

웹 정보원은 사용자에게 제공되는 가장 큰 분산 데이터베이스로 간주될 수 있다. 분산된 웹 정보원을 가상적으로 통합하여 하나의 정보원으로 보고, 이 가상의 데이터베이스에 대하여 데이터베이스 질의어를 이용하여 검색하는 기능은 여러 가지 웹 기반 응용프로그램 개발에서 중요한 역할을 한다. 이러한 기능을 지원하기 위해서 브라우징 중심의 웹 문서로부터 데이터베이스 스키마를 추출해야 한다.

본 논문은 반 구조적인 웹 문서로부터 XML 스키마를 자동으로 생성하는 경험적 알고리즘을 제시한다. 이를 위하여 미리 정의된 구조태그 기반으로 후보패턴영역을 추출하고 후보패턴영역으로부터 패턴영역을 경험적으로 결정한다. 그리고 패턴영역으로부터 스키마생성규칙을 유도한다. 스키마생성규칙은 XQuery로 표현되기 때문에 공개된 다양한 XML 도구를 사용하여 응용시스템을 개발할 수 있다. 개발된 시스템의 유효성을 보이기 위하여 다양한 웹 정보원에 대하여 실험한 결과를 제시한다.

Abstract

The Web information sources can be regarded as the largest distributed database to the users. By virtually integrating the distributed information sources and regarding them as a single huge database, we can query the database to extract information. This capability is important to develop Web application programs. We have to infer a database schema from browsing-oriented Web documents in order to integrate databases.

This paper presents a heuristic algorithm to infer the XML Schema fully automatically from semi-structured Web documents. The algorithm first extracts candidate pattern regions based on predefined structure-making tags, and determines a target pattern region using a few heuristic factors, and then derives XML Schema extraction rules from the target pattern region. The schema extraction rule is represented in XQuery, which makes development of various application systems possible using open standard XML tools. We also present the experimental results for several public web sources to show the effectiveness of the algorithm.

▸ Keyword : 정보추출(Information Extraction), XML스키마(XML Schema), XML, 반복패턴(Repeated Pattern), 정보통합(Information Integration)

• 제1저자 : 곽준영 교신저자 : 배종민
• 접수일 : 2008. 7. 21, 심사일 : 2008. 9. 22, 심사완료일 : 2008. 11. 26.
* (주)위너스텍 ** 경상대학교 컴퓨터과학부

Ⅰ. 서 론

웹 정보원은 다양한 정보를 다양한 형태로 제공한다. 일반적으로, 웹으로부터 검색된 결과로부터 사용자가 원하는 데이터를 다시 추출해야 하는 응용이 많이 있다[3]. 이들 응용들이 여러 가지 서비스를 효과적으로 제공할 수 있기 위해서는 다양한 웹 정보원으로부터 검색된 데이터를 통합하여 사용자에게 마치 하나의 웹 정보원에서 검색한 결과처럼 재구성하여 제공해야 한다. 전통적으로, 다수의 관계형 데이터베이스를 통합하여 하나의 논리적인 데이터베이스를 제공하는 문제는 소위 데이터베이스 통합이라는 고전적인 문제인데, 다양한 웹 정보원에 대해서도 유사한 기능이 필요하다. 관계형 데이터베이스의 경우에는 데이터베이스 스키마가 명시적으로 제공되기 때문에 각 데이터베이스의 스키마 통합을 통해서 통합질의가 가능하다. 비슷한 개념으로, 웹 정보원을 논리적으로 통합해서 SQL이나 XQuery와 같은 데이터베이스 질의어를 사용하여 통합 검색하기 위해서는 웹 정보원이 제공하는 문서에 대한 스키마가 필요한데, 웹 문서에 대한 스키마는 보통 제공되지 않는다. 이러한 문제를 해결하기 위해서는 웹 문서로부터 스키마를 추론해야 한다.

특정 사이트에 대한 스키마를 추출하기 위해서는 그 사이트에 관한 사전 지식이 필요하다. 즉 웹 문서에서 반복되는 패턴을 찾아서 그 패턴의 규칙을 찾는다. 웹 문서는 대개 정보원 측의 데이터베이스로부터 데이터를 추출하여 만들어지기 때문에 대부분의 문서에서 반복패턴이 존재한다. 그러한 반복패턴에서 반복규칙을 인지하여 브라우징 관련 태그를 제외하고 문서의 내용만 표현하는 XML 문서를 생성할 수 있다. 일단 XML 문서와 같은 구조화된 문서를 얻으면 XML 스키마를 생성할 수 있고, 계속해서 XQuery와 같은 질의어를 통하여 검색을 할 수 있다. XML 문서를 생성하기 위한 추출규칙을 표현하기 위해서는 그 규칙을 정의하는 특수 언어를 별도로 설계할 수도 있고, XQuery와 같은 기존의 언어를 활용할 수 있다. 그런데, 생성된 정보추출규칙은 대개 매우 복잡하여 수작업으로 만들기가 대단히 어렵다. 게다가 웹 사이트의 구조가 바뀌면 정보추출규칙을 다시 정의해야 한다[2].

따라서 본 연구에서는 완전 자동 정보추출 알고리즘을 제시한다. 즉 웹 정보원이 제공하는 문서에 대한 XML 스키마를 추론할 수 있는 정보추출규칙을 자동으로 생성하는 알고리즘을 제시한다. 먼저 HTML 문서에서 스키마 정보를 효과적으로 추출할 수 있도록 구조태그를 체계적으로 분류한다. 이 구조태그를 기반으로 후보패턴 영역을 결정한 다음, 다수의 후보패턴 영역으로부터 패턴영역을 결정하는 경험적 알고리즘을 제시한다. 반복패턴으로부터 XML문서를 생성하는 정보추출규칙을 자동으로 생성하는 알고리즘을 제시한다. 그리고 개발된 시스템의 유효성을 보이기 위하여 다양한 사이트에서 실험한 결과를 제시한다. 또한, 본 논문에서 개발된 웹 정보추출시스템이 실제 응용에서 쉽게 활용될 수 있는 미들웨어로서의 기능을 갖도록, 정보추출규칙은 표준 XML 질의어인 XQuery[10]로 표현되며, 이를 기반으로 XML 문서와 XML 스키마를 자동으로 생성함으로써, XML 문서처리에 관련된 공개된 표준 도구를 활용할 수 있도록 하였다.

본 논문의 구성은 다음과 같다. 2장에서는 관련연구를 살펴보고, 3장에서는 개발된 정보추출 시스템을 간단히 소개한 다음, 웹 정보추출규칙을 생성하는 알고리즘을 제시하고 4장에서는 정보추출 시스템에 대한 구현 및 성능평가에 대하여 논한다.

Ⅱ. 관련연구

웹 문서로부터 정보추출규칙을 자동 혹은 반자동으로 생성하는 연구는 텍스트마이닝과 함께 꾸준히 진행되고 있다. 정보추출규칙을 생성하는 대표적인 모델 중의 하나로 기계학습 기법이 있다[1]. 기계학습모델 기반의 정보추출시스템 중에서 대표적인 웹 정보추출시스템 KnowItAll[5]는 사용자가 트레이닝 데이터를 주어야 하기 때문에 사용자 개입이 어느 정도 필요하다[8]. 이를 개선한 시스템으로 TextRunner[7]가 있는데, 이 시스템은 스스로 학습하려는 시도를 하여 사용자의 개입을 최소화하여 방대한 웹 데이터베이스에 대하여 쉽게 적응할 수 있는 시스템을 개발하였다. 다른 한편으로 완전 자동으로 정보를 추출하는 시도는 주로 문서의 구조를 바탕으로 이루어져 왔다[3,4]. IEPAD[4]는 웹 문서가 비교적 규칙적이고 반복적인 패턴으로 이루어져 있다는 사실을 바탕으로 문서의 구조로부터 PAT 트리를 유도하여 패턴을 추출한다. 입력된 HTML 문서에 대해 태그와 텍스트를 구분하고 각 태그의 고유한 비트열을 이용하여 이진 데이터를 생성한다. 생성된 이진 데이터를 이용하여 PAT 트리를 생성하고 이를 이용하여 데이터 구조를 파악한다. PAT 트리 생성 과정을 통해 다수의 패턴을 찾고 확인자(validator)를 통해 고려되지 않은 패턴을 걸러낸다. 그리고 규칙합성(Rule Composer)을 거쳐 패턴에 대한 추출규칙을 생성한다.

MDR(Mining Data Records in Web Pages) 시스템[3]은 동일한 구조를 가지는 데이터 레코드의 집합인 데이터 영역을 정의하고 태그 트리에서 스트링 매칭 알고리즘을 활용하여

데이터 영역을 추출한다. 데이터 영역을 결정할 때 동일한 구조를 이루는 데이터 레코드 간의 상하 포함 관계를 파악하여 결정하기 때문에 일반적인 형태의 연속적인 정보뿐만 아니라 비연속적인 정보에 대해서도 추출이 가능한 장점을 가진다.

본 논문에서 제시한 시스템이 앞에서 제시한 시스템과 차이점은 다음과 같다. 첫째, 문서의 구조를 파악하기 위하여 구조태그를 새로 정의하고 이를 바탕으로 반복패턴을 경험적으로 결정하는 완전자동 스키마추출시스템이다. 둘째, 개발된 시스템은 정보추출규칙을 XQuery로 표현하고 웹 문서로부터 추출되는 정보는 XML문서 및 XML 스키마이기 때문에, 본 시스템을 사용하면 다양한 XML 표준 도구를 활용하여 사용자의 응용시스템 개발을 지원할 수 있는 미들웨어이다.

III. 웹 정보추출 알고리즘

본 장에서는 웹 문서에 대한 정보추출규칙을 생성하는 방법을 제시한다.

1. 시스템 개요

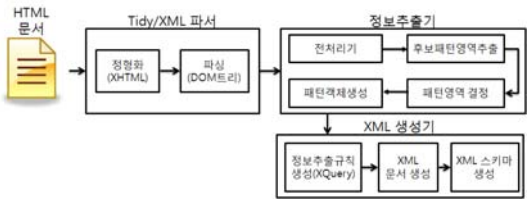

그림 1. 스키마생성시스템의 구조
Fig.1 The Architecture of a Schema Generation System

그림 1은 개발된 시스템의 개략적인 구조이다. 사용자의 질의 결과 얻어진 HTML 문서는 먼저 정형화를 통하여 XHTML 문서로 변환된다. 일반적인 HTML 문서는 XML 문법에서 요구하는 양식에 맞지 않은 경우가 많이 발생한다. 예를 들어 동일한 속성의 선언 또는 닫는 태그가 없는 태그의 사용 등이 있다. 본 시스템에서는 공개된 소프트웨어 Tidy[6]를 이용하여 HTML 문서를 정형의(well-formed) XML 문서 구조로 수정한 결과인 XHTML 문서를 생성한다. 그리고 XHTML문서를 파싱하여 DOM[11] 트리를 생성한다. 이후 전처리 과정에서는 DOM 트리에 새로운 속성을 정의하는데 이는 패턴 영역 추출 과정에서 DOM 트리의 반복적인 구조를 파악하기 위해 사용된다. 계속해서 DOM 트리의 구조 태그를 기반으로 동일한 구조로 반복되는 후보 패턴 영역을 모두 추출한다. 이 과정에서 얻어진 다수의 후보 패턴 영역에서 하나의 패턴 영역을 추출하고, 추출된 하나의 패턴영역의 반복패턴으로부터 정보추출규칙을 생성하는데 필요한 모든 정보를 포함하는 객체를 생성한다. 그리고 이 객체로부터 XQuery로 표현된 정보추출규칙을 생성한다. 계속해서 이를 이용하여 XML 스키마를 생성한다.

2. 구조 태그의 분류

HTML 문서의 태그는 크게 두 부류로 나눌 수 있다. 첫 번째는 화면상에 보여주기 위한 스타일 태그와 두 번째는 문서구조의 성격을 가지는 구조태그이다. 스타일 태그는 텍스트의 크기, 색깔 등과 같이 화면상에 보여주는 것과 관련된 태그를 말한다. 구조 태그는 스타일 태그와 달리, 텍스트의 표현되는 영역을 구조적으로 나타내기 위한 역할을 한다. 본 논문에서는 HTML 문서에서 구조 태그를 중심으로 하여 정보를 추출한다. 즉 HTML 문서에서 스타일 태그를 제외한 구조 태그 중심으로 보면 복잡한 HTML 문서를 간소화하면서 구조 파악이 쉬워지기 때문에 반복패턴을 쉽게 구할 수 있다. 본 논문에서는 구조태그를 다음과 같이 크게 3 가지의 부류로 나눈다.

(1) class 1 : table, tbody, ul, ol, dl

class 1 구조 태그는 하나의 정보 영역을 독립적으로 구분할 수 있는 태그를 말한다. class 1 구조 태그 하위로 다수의 class 2, 3 구조 태그가 올 수 있으며 하나의 패턴 영역에서의 루트가 되는 부분이다. 하나의 구조단위를 완성하는 역할을 하는데, 관계형 데이터베이스에 비유하면 테이블과 같은 역할을 한다.

(2) class 2 : tr, colgroup, li, dt, dd, div, p

class 2 구조 태그는 반복의 의미를 가지는 구조 태그를 말한다. 즉, 하나의 패턴 영역에서 패턴이 될 수 있는 구조 태그를 일컫는다. 관계형 데이터베이스에서 튜플의 개념과 같다고 보면 된다.

(3) class 3 : td, th, col, li, dt, dd, div, p

class 3 구조 태그는 하나의 데이터 항목을 의미하는 구조 태그를 말한다. 관계형 데이터베이스에서 하나의 칼럼 개념을 가진다. class 2 구조 태그와는 달리 패턴은 될 수 없고 데이터 단위만을 구분 짓는 태그이다. class 3 구조 태그 중 li, dt, dd, div, p 구조 태그는 성격상 반복의 의미뿐만 아니라 하나의 데이터 항목도 될 수 있기 때문에 class 2 구조 태그이면서 동시에 class 3 구조 태그이기도 하다.

3. 전처리기

전처리기는 후보 패턴 영역 추출을 쉽게 하기 위하여 DOM 트리에서 구조 태그에 대하여 3가지 속성을 추가한다. 첫째, textPath 속성은 텍스트 노드를 가지는 경로를 표시하는 속성인데, 0은 텍스트를 가지지 않는 경로를, 1은 텍스트를 가지는 경로를 의미한다. 예를 들어, 그림 2에서 텍스트 노드가 있는 경로만 1로 주어진다. 이는 후보 패턴 영역을 추출할 때 텍스트가 존재하는 경로만 탐색하기 위함이다.

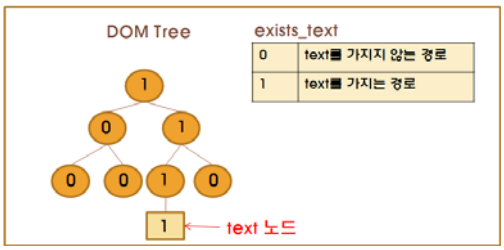

그림 2. textPath 속성의 의미
Fig 2. The Meaning of textPath

둘째, depth 속성은 DOM 트리에서 구조 태그의 깊이를 말한다. 이는 구조 태그의 계층 관계만을 쉽게 고려할 수 있다. 셋째, naming 속성은 구조 태그마다 부여된 상수를 말한다. 그림 3은 각 구조 태그에 부여된 상수이다. 텍스트노드는 구조 태그에 속하지 않지만 상수가 부여된다. 이는 depth 속성과 더불어 후보 패턴 영역을 결정할 때 구조 비교에 활용된다.

구조 태그	상수	구조 태그	상수
table	a	div	j
ol	b	p	k
ul	c	td	l
dl	d	th	m
tr	e	col	n
colgroup	f	tbody	o
li	g	text	t
dt	h		
dd	i		

그림 3. 구조태그의 코드화
Fig. 3 Coding the Structured Tags

4. 후보패턴영역 추출

웹 문서에는 반복패턴을 가지는 영역이 다수 있다. 이들을 후보패턴영역이라 부른다. 사용자에게 필요한 정보를 제공하는 패턴영역을 결정하기 위하여, 먼저 웹 문서내의 모든 후보 패턴영역을 추출한다. 후보패턴영역을 추출하기 위한 첫 단계는 DOM 트리의 루트로부터 구조 태그를 찾는데, 텍스트 정보를 가지고 있는 노드만을 탐색하도록 노드의 exists_text 속성이 1인 것만 탐색한다. 구조 태그를 만나면 하위 자식 노드의 집합에 대하여 다음 조건을 검사한다.

(1) 노드의 exists_text 속성 값이 1인가?
(2) 클래스 1, 혹은 클래스 2 구조 태그인가?

위의 2 가지 조건을 만족하는 노드의 개수가 노드의 집합에서 2개 이상이 되면 이러한 조건을 만족하는 노드의 집합에 대해 하위 구조를 비교하는 작업을 수행하는데, 이를 위해서 각 노드마다 하위의 구조정보를 미리 추출한다. 여기서 구조정보라는 것은 깊이우선탐색과정을 통해서 발생하는 구조태그의 종류(naming 속성)와 트리 상에서의 깊이(depth)를 모은 구조를 말한다.

그림 4는 구조 태그 div의 하위 부분에 대한 구조정보를 예시한 것이다. 구조정보의 첫 번째 요소인 (a, 1)이 의미하는 것은 그림 3.에서 제시된 바와 같이, <table> 태그의 깊이가 1임을 의미한다.

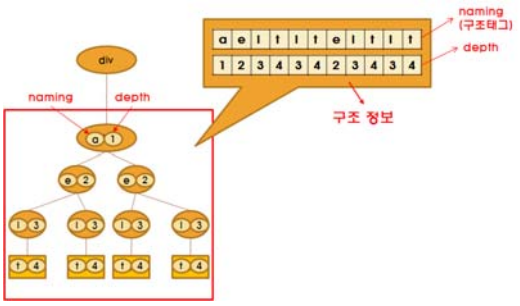

그림 4. 노드의 구조정보
Fig. 4 Structual Information of a Node

한편, 이러한 구조 정보를 추출할 때 텍스트 노드가 형제 노드 관계로 연속적으로 올 경우 이를 하나의 텍스트로 처리하는데 이는 혼합 엘리먼트 처리 과정이다.

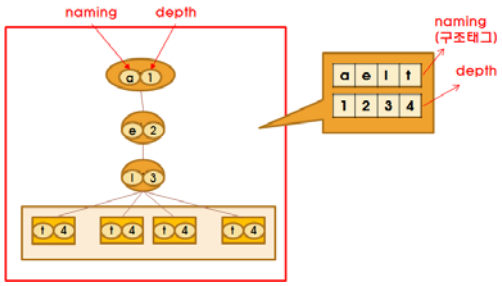

그림 5. 혼합 엘리먼트 처리
Fig. 5 Processing of Mixed Elelement

예를 들어, 그림 5에서 DOM 트리 하단의 4개의 노드는 형제 관계를 가지는 텍스트 노드인데, 하나의 텍스트로 취급되어 하나의 정보로 취급된다.

다음으로, 구조를 비교하는 방법은 다음과 같다. 예를 들어, 10개의 노드가 있다고 가정하면 그 비교 순서는 그림 6과 같다.

1. (1,2) (2,3) (3,4) (4,5) (5,6) (6,7) (7,8) (8,9) (9,10)
2. (1-2,3-4) (3-4,5-6) (5-6,7-8) (7-8,9-10)
3. (1-2-3,4-5-6) (4-5-6,7-8-9)
4. (1-2-3-4,5-6-7-8)
5. (1-2-3-4-5,6-7-8-9-10)

그림 6. 구조 비교 순서
Fig. 6 Order of Structure Comparison

첫 번째 단계에서 (1, 2) 라는 것은 1번 노드와 2번 노드를 비교하는 것을 말한다. 이렇게 하나의 노드씩 검사한 후, 만약 구조가 동일하지 않으면, 이번에는 1번과 2번 노드를 묶어서 하나의 단위로 취급하고, 이것이 2번, 4번 묶음의 구조와 동일한지 비교한다. 이와 같은 과정을 5개 노드의 묶음까지 반복한다. 동일한 구조를 가지는 묶음의 수가 각 단계에서 총 묶음 수의 반 이상이 되는 경우 그 부분을 후보패턴 영역으로 지정한다. 만약 총 5 단계의 비교 과정을 거친 후에도 동일한 구조가 발견되지 않으면 후보 패턴이 없는 것으로 간주된다.

5. 패턴영역 결정

다수의 후보패턴 영역은 우리가 얻고자 하는 서버의 DB로부터 구성된 데이터를 포함할 수도 있고 HTML 문서 자체의 태그에 대한 반복일 수도 있다. 예를 들어 그림 7에서 표시된 후보패턴영역은 4개가 있다.

그림 7. 다수의 후보패턴 영역
Fig. 7 Candidate Pattern Regions

이 중에서 서버의 DB에서 추출된 데이터를 포함하고 있는 패턴영역이 무엇인지를 결정해야 한다. 다수의 후보패턴 영역에서 하나의 패턴을 결정하기 위해 3 가지의 경험적 요소를 사용하였다.

(1) DOM 트리에서 패턴의 루트까지의 깊이(depth)

HTML 문서에서 정보가 위치하는 영역은 HTML 문서에서 깊이가 깊은 부분에 위치하고 있을 가능성이 높다는 요소이다. 서버가 DB로부터 얻어진 정보로써 HTML 문서를 구성할 때, 구조 태그를 사용하여 반복구조로 표현하는데 이러한 정보 영역은 많은 경우에 DOM 트리에서 위치적으로 깊은 부분에 위치하고 있다. 다수의 후보 패턴 영역에서 각 후보 패턴 영역의 depth를 비교함으로써 패턴 영역의 결정을 도울 수 있다.

(2) 패턴영역에서 반복패턴의 개수(count)

DB부터 얻어온 데이터는 HTML 문서에서 반복적으로 나타나는 반면에, 그 외의 HTML 문서 부분은 이러한 데이터를 보여주기 위한 하나의 틀이라고 볼 수 있다. 따라서 정보 영역과 그 외의 부분에 대한 구조의 반복성 정도를 비교함으로써 패턴영역의 결정을 도울 수 있다.

(3) 패턴의 루트 노드에서 텍스트 노드까지의 깊이 차이(size)

각 후보패턴 영역의 루트 노드의 깊이와 그 루트 노드의 가장 하위에 있는 텍스트 노드와의 깊이의 차이를 통해 패턴 영역을 추정한다. DB로부터 얻어온 데이터의 양이 많을수록 다양한 구조 태그를 이용하여 정보 영역을 결정하는 경향을 반영한 것이다.

각 후보패턴 영역마다 이 3가지 경험적 값을 결합하는 방법은 다음의 방법에 따른다.

$$A \cup B = A + B - (A \cap B)$$

각 영역에 대하여, 경험적 요소 값 각각의 백분율을 구하여 위의 결합방법을 적용하여 각 후보패턴 영역의 요소 합을 구하고 이 값이 가장 큰 후보패턴 영역을 패턴영역으로 결정한다. 이 과정을 그림 8.에서 예시하였다.

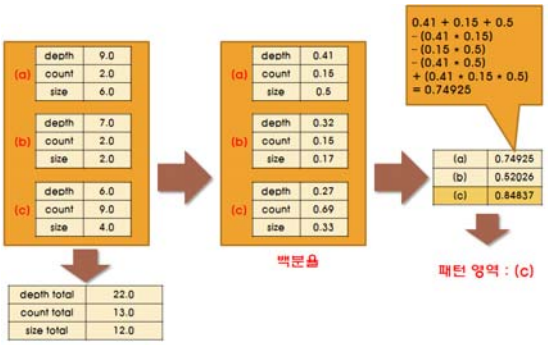

그림 8. 패턴 영역의 결정
Fig. 8 The Decision of a Pattern Region

그림 8에서 (a), (b), (c)는 각 후보패턴 영역을 의미한다. 우선 각 후보패턴 영역에서 경험적 요소별 총합을 구한다. 그 총합을 이용하여 각 후보패턴 영역의 각 요소에 대한 백분율을 구한다. 예를 들어 영역 (a)의 depth의 백분율은 9.0 / 22.0 = 0.41이다. 계속해서 영역 (a)의 각 요소를 결합하면 0.41 + 0.15 + 0.5 - (0.41 * 0.15) - (0.15 * 0.5) - (0.41 * 0.5) + (0.41 * 0.15 * 0.5) = 0.74925이다. 그 결과 가장 큰 값을 가지는 영역 (c)가 후보패턴 영역으로 결정된다.

이제 패턴영역이 결정되면 패턴에 관한 모든 정보를 포함하는 패턴정보객체(PatternInfo)를 생성한다. 이 객체의 구성은 그림 9와 같다.

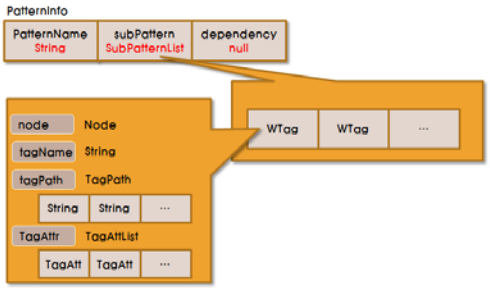

그림 9. 패턴객체의 구조
Fig. 9 The Structure of a Pattern Object

패턴정보객체(PatternInfo)는 패턴의 이름(PatternName), 서브패턴(subPattern)리스트, 그리고 패턴의 내포관계를 의미하는 의존관계(dependency)로 구성된다. 서브패턴은 패턴에 대한 노드(Node)와 그 태그 이름, 경로(TagPath), 속성리스트(TagAttList)로 구성된 객체의 리스트이다. 이는 XQuery로 표현된 정보추출규칙을 생성하는데 필요한 모든 정보를 가지고 있다.

6. 정보추출규칙 생성

그림 10은 정보추출규칙 생성 과정을 간단하게 표현한 것이다. 앞 단계에서 생성된 패턴객체와 DOM 트리를 이용하여 XQuery 언어로 표현된 정보추출규칙을 생성한다.

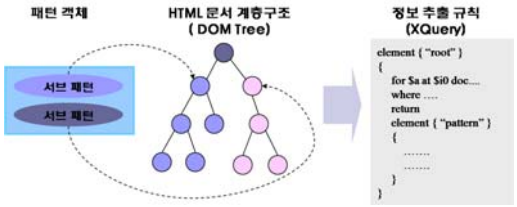

그림 10. 정보추출규칙 생성 과정
Fig. 10. The Procedure to Create Information Extraction Rules

정보추출규칙 생성 단계에서 먼저 문서의 최상위 엘리먼트인 루트 엘리먼트를 생성한다. 다음으로 패턴 객체에 저장되어 있는 서브 패턴 정보를 검사하고, 서브 패턴 경로를 추출하고 추출된 경로 정보를 변수에 바인딩하여 for절을 생성한다. 그리고 서브 패턴의 속성(Attribute) 정보를 추출하여 for절에서 바인딩된 변수를 이용하여 속성 값의 존재 여부를 판단하는 where절을 만든다. 다음으로 return절을 생성하고, return 절 하위에는 엘리먼트 pattern을 생성해 하위의

반복되는 엘리먼트들을 묶어준다. 엘리먼트 pattern 하위에는 사용자가 지정한 패턴 정보에서 생성한 엘리먼트들이 위치하게 된다.

return절은 추출하고자 하는 정보와 구조가 반영된 엘리먼트와 엘리먼트 값으로 구성 된다. 패턴으로 지정된 부분의 하위의 태그들을 깊이우선 탐색하면서 엘리먼트와 엘리먼트 값을 생성한다. 그림 11.은 패턴객체를 이용하여 만들어진 정보 추출 규칙(XQuery)의 기본 구조이다.

```
element { "root" }
{
    for $a at $i0 doc(..../html/body/...../table/tr/)
    where exsit(%a/border)
        and exsit(%a/width)
        ******
    return
        element { "pattern" }
        {
            *******
            A 부분
            *******
        }
}
```

그림 11. 정보추출규칙의 문법적 구조
Fig. 11 The Grammar for an Information Extraction Rule

그림 11의 A부분은 패턴객체에 저장되어 있는 서브패턴의 경로를 DOM 트리에서 찾고 찾은 노드를 시작으로 깊이우선 탐색으로 엘리먼트를 생성하는 부분으로, 그 과정은 그림 12와 같다. 깊이 우선 탐색을 진행하면서, 현재 위치한 노드의 타입이 엘리먼트 노드일 경우에는 구조 태그에 속하는 태그인지를 판단한다.

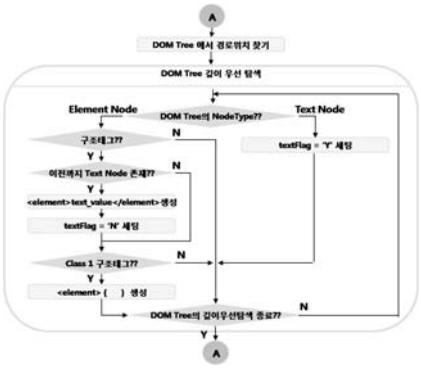

그림 12 정보추출규칙 생성 알고리즘
Fig. 12 The Algorithm of Information Extraction Rules

구조태그를 만나면 현재의 노드 이전까지 텍스트 노드가 존재했는지 판단하고, 존재했다면 엘리먼트를 생성한다. 엘리먼트를 생성할 때, 엘리먼트의 값은 이전 구조태그 하위에 있는 텍스트 노드 값으로 만들어진다. 다음으로 해당 구조태그가 Class 1에 해당하는 구조 태그인지 판단한다. Class 1에 속하는 구조 태그일 경우 하위 구조를 묶어 줄 수 있는 엘리먼트와 중괄호를 생성한다. 이 과정은 DOM 트리의 깊이 우선 탐색이 종료될 때까지 반복한다.

예를 들어 그림 13은 태그가 패턴의 루트일 때, 태그의 하위 구조를 표현하고 있다. 태그를 시작으로 DOM 트리를 깊이우선 탐색하면서 첫 번째 구조태그를 만났을 때, 첫 번째 태그 이전에 텍스트 노드가 존재하지 않기 때문에 계속 깊이우선탐색을 진행한다. 두 번째 를 만났을 때도 이전에 텍스트 노드가 존재하지 않기 때문에 계속 진행한다. 다음으로 <p> 구조 태그를 만났을 때, 이전에 "text1"이라는 텍스트 노드가 존재하므로 이전 구조태그 경로 정보를 이용하여 엘리먼트와 엘리먼트 값을 생성한다.

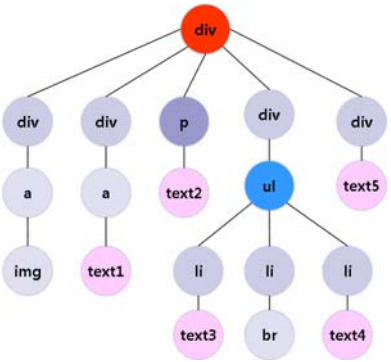

그림 13. 정출규칙 생성 예
Fig. 13. An Example to 보추Create Information Extraction Rules

계속해서 <ul> 구조태그를 만나면, <ul> 구조태그는 Class 1(구조를 완성하는 단위)에 해당하는 구조 태그이므로 하위 구조를 묶어주는 기능을 가지는 엘리먼트와 괄호({})를 생성 한다. 그림 14는 이와 같은 과정에서 생성된 정보 추출 규칙(XQuery)의 일부이다.

```
element{"pattern"}
{
  element{"name1"} { ../div/div[2]/text() } ,
  element{"name2"} { ../div/p[1]/text() } ,
  element{"name3"}
  {
    element{"name4"} { ../div/div[3]/ul[1]/li[1]/text() } ,
    element{"name5"} { ../div/div[3]/ul[1]/li[3]/text() } ,
  } ,
  element{"name6"} { ../div/div[4/text() }
}
```

그림 14 생성된 정보 추출 규칙의 일부
Fig. 14 A Part of the Created Rule

IV. 구현 및 성능평가

본 시스템은 사용자로부터 URL이 주어졌을 때, 그 문서에 대한 정보추출과정을 거쳐서, XQuery로 표현된 정보추출 생성규칙과 생성규칙에 의해서 생성되는 XML 문서, 생성된 XML 문서에 대한 XML Schema를 모두 자동으로 생성하여 파일로 저장한다.

시스템 구현시 HTML 문서를 파싱하여 XHTML 문서로 변환하는 도구는 공개 소프트웨어인 Tidy[6]를 활용하였다. 공개 소프트웨어 Tidy를 통해 올바른

표 1. 테스트 사이트
Table 1. Test Sites

No.	name	url
1	niceprot	www.expasy.org
2	에브리존	www.everyzone.com
3	swissEntry	srs.ddbj.nig.ac.jp
4	대동서적	www.ddbook.co.kr
5	생물학연구정보센터	bric.postech.ac.kr
6	의학학술지종합정보시스템	medlis.riss4u.net
7	대학교 도서 검색	library.gsnu.ac.kr
8	empas열린검색	search.empas.com
9	코리아닷컴	search.korea.com
10	DELL	searchapj.dell.com
11	GeneCards	biomed.kobic.re.kr
12	네이버 지식검색	web.search.naver.com
13	pubmed	www.ncbi.nlm.nih.gov
14	nucleotide	www.ncbi.nlm.nih.gov
15	information.com	www.information.com
16	yahoo	search.yahoo.com
17	네이버 지식쇼핑	shopping.naver.com
18	yes24	www.yes24.com
19	네이버 책	book.naver.com
20	amazon.com	www.amazon.com
21	paran 검색	search.paran.com
22	RadioShack	www.radioshack.com
23	리브로 검색	www.libro.co.kr
24	barnes&noble	search.barnesandnoble.com
25	bookpool.com	www.bookpool.com
26	두산백과	www.encyber.com
27	google1	203.255.3.152
28	google2	203.255.3.152
29	kodak	search.kodak.com
30	침코리아쇼핑몰	shop.life21.co.kr
31	lycos 검색	search.lycos.com
32	bookbay	www.bbay.co.kr
33	hmall.com	www.hmall.com
34	scirus	www.scirus.com
35	mamma	www.mamma.com
36	G마켓	www.gmarket.co.kr
37	천리안검색	search.chol.com
38	altavista	kr.altavista.com

XHTML 문서가 생성되지 않는 사이트는 약 15% 정도 있었다. 이는 주로 한글 사이트에서 발생했는데, 실험에서 그러한 문서는 제외하였다. 이는 HTML 문서의 파싱 문제로인데 이는 본 논문과 직접적인 관계는 없다. 그러나 특히 한글 사이트에서 파싱되지 않는 사이트가 많다는 것은 주목해야 할 일이다.

제안한 정보추출시스템의 성능을 보이기 위하여 정상적으로 파싱되는 38개의 사이트에 대하여 실험하였다. 표 1은 파싱이 성공적으로 되는 사이트로서 본 논문에서 실험을 한 사이트를 나열한 것이다.

실험 결과, 총 38개의 사이트 중 3개의 사이트는 후보패턴 영역을 추출하지 못하였다. 그 이유는 반복 패턴이 <table>, 등과 같은 구조태그가 아닌, 글자의 색깔, 크기 등과

같은 스타일태그로 구성되어 있기 때문이다. 따라서 문서의 구조태그 기반의 후보패턴영역 추출의 정확도는 35/38 × 100 = 92%이다. 이는 복잡 다양한 웹 문서에 대하여 구조태그 기반의 정보추출방법에 대한 정확도이다. 그리고 후보패턴 영역을 추출한 35개의 사이트에서 5개 사이트는 후보패턴 영역에서 패턴영역을 추출하지 못하였다. 따라서 패턴영역을 결정하기 위한 경험적 방법론에 대한 정확도는 30/35 × 100 = 86%이다. 본 논문에서는 간단한 경험적 요소만을 사용했음에도 불구하고 비교적 우수한 결과가 도출되었다. 일단 패턴영역이 추출되면 그로부터 XML 스키마는 100% 정확히 자동으로 생성되었다. 따라서 임의의 웹 문서로부터 XML 스키마를 완전 자동으로 성공적으로 생성하는 사이트의 비율은 86%이라고 할 수 있다. 일단 패턴영역만 추출되면 XML 스키마는 자동으로 100% 정확히 생성되기 때문에, 임의의 웹 사이트에 대한 스키마 생성의 성공률을 높이기 위해서는 패턴영역을 결정하는 경험적 방법론의 정확도를 높여야 한다. 그 정확도를 높이기 위해서는 보다 정교한 통계학적 기법을 개발할 필요가 있다. 한편 일반적으로 웹 문서는 매우 복잡하고 불규칙적인 구조가 많이 포함되어 있어서 완전 자동으로 스키마를 추출할 때의 정확성은 한계가 있다. 따라서 필요시 사용자의 개입을 통해서 패턴영역만 지정하면 본 시스템은 스키마를 정확히 추출할 수 있다.

V. 결론

방대한 정보를 가진 다수의 웹 정보원을 가상적으로 통합하여 데이터베이스 질의어를 통하여 정보를 얻는 시스템은 여러 응용에서 활용된다. 이를 지원하기 위해서는 가상적으로 통합된 웹 정보원에 대한 스키마를 정의해야 한다. 웹 문서는 일반적으로 반구조적이거나 비구조적이고 문법에 맞지 않는 문서 등으로 인하여 매우 복잡하고 다양한 불규칙성이 있기 때문에 정보원에 대한 스키마를 추출하기는 매우 어렵다. 본 연구에서는 이러한 웹 정보원에 대하여 자동으로 스키마 추출 규칙을 생성하는 시스템을 제시하였다. 제시한 웹 정보추출시스템은 정보추출을 위한 구조태그를 분류하고 이를 기반으로 문서구조정보를 이용한 후보 패턴영역 추출 알고리즘과 패턴영역 결정을 위한 경험적 알고리즘을 사용한다. 그리고 제안된 추출알고리즘의 정확성을 다양한 사이트를 통해서 실험한 결과를 보였다. 제시한 시스템은 타 시스템과는 달리, 문서의 구조정보를 활용하기 때문에 데이터 트레이닝과 같은 사용자 개입이 없고 정보추출의 효율성이 높다. 또한 XQuery로 표현된 정보추출규칙, 이를 실행시켜서 생성되는 XML 문서, 그리고 XML 스키마를 자동으로 생성하고, XML 질의어인 XQuery를 사용하는 범용 미들웨어이다. 향후 웹 문서의 다양한 불규칙성을 효과적으로 처리하기 위해서는 문서구조 기반 모델과 기계학습 모델을 혼합한 정보추출방법론에 대한 연구가 더 필요하다.

참고문헌

[1] Alberto H. F. Laender, Berthier A. Ribeiro-Neto, Altigran S. da Silva, Juliana S. Teixeira, "A brief survey of Web Data extraction tools." ACM Sigmod Record. 31(2) pp.84-93, June 2002.

[2] A. Doan, R. Ramakrishnan, S. Vaithyanathan, "Managing Information Extraction: State of the Art and Research Directions," ACM SIGMOD, 2006.

[3] B. Liu, R. Grossman, and Y. Zhai, "Mining Data Records in Web Pages", ACM SIGKDD '03, pp.601-606, August 2003.

[4] C. H. Chang, and S. L. Lui, "IEPAD : Information Extraction Based on Pattern Discovery", Proc. of WWW10, pp681-688, 2001.

[5] O. Etzioni, et. al, Unsupervised Name-Entity Extraction from the Web: An Experimental Study, Artificial Intelligence, vol. 165, No. 1 pp91-134, 2005.

[6] HTML TIDY, [Online]: http://tidy.sourceforge.net/

[7] M. Banko, M. J. Cafarella, S. Soderland, M. Broadhead, O. Etzioni, "Open Information Extraction from the Web", Proc. of IJCAI, pp.187-206, 2007

[8] M. J. Cafarella, O. Etzioni, D, Suciu, "Navigating Extracted Data with Schema Discovery", Proc. of the 10th International Workshop on WebDB, 2007

[9] World-Wide Web Consortium, "XQuery 1.0: An XML Query Language", [Online]: http://www.w3.org/TR/xquery/, W3C Candidate Recommendation 8 June 2006.

[10] World-Wide Web Consortium, "Document Object Model (DOM) Level 3 Core Specification", [Online]: http://www.w3.org/TR/ DOM-Level-3-Core/, W3C Recommendation, April 2004.

저 자 소 개

곽 준 영(Jun-Young Kwak)

2006년 경상대학교 컴퓨터과학과 학사
2008년 경상대학교 대학원 컴퓨터과학과 석사
2008년~현재 위너스텍(주)s/w연구원
관심분야: XML, 정보통합, 웹정보기술

배 종 민(Jong-Min Bae)

1980년 서울대학교 수학교육과(학사)
1983년 서울대학교 대학원 계산통계학과(석사)
1995년 서울대학교 대학원 계산통계학과(박사)
1982년-1984년 한국전자통신연구원 연구원
1997년-1998년 Virginia Tech. 객원연구원
1984년-현재 경상대학교 자연과학대학 컴퓨터과학부 교수
관심분야: XML, 정보통합, 상황인식, 의료정보