韓國컴퓨터情報學會 論文誌
第14卷 第5號, 2009. 5.

2009-14-5-1-1

XML 데이터의 제약조건 보존을 위한 변환 기법

조 정 길*, 금 영 욱*

A Transformation Technique for Constraints-preserving of XML Data

Jung-Gil Cho*, Young-Wook Keum*

요 약

XML 데이터를 효율적으로 저장하고 질의하기 위하여 많은 기법들이 제안되었다. 이러한 목표를 위한 한 가지 방법은 XML 데이터를 관계형 형식으로 변환하여 관계형 데이터베이스를 사용하는 것이다. 그러나 대부분의 연구가 XML의 내용과 구조만 변환하고 숨겨진 의미적 제약조건을 간과하거나 일부만 적용하였다. 따라서 이 논문에서는 XML Schema로부터 의미적 제약조건의 체계적인 추출 방법과 추출된 의미적 제약조건을 관계형 스키마로 변환할 때에 보존하는 방법을 제안한다. 변환 알고리즘은 XML Schema로부터 의미적 제약조건을 추출하고 보존하는데 이용되며, 추출된 의미적 정보들을 스키마 표기법에 따라 재작성하여 어떻게 의미적 제약조건을 보존하는지를 보여준다. 또한 변환하는 동안에 올바른 관계형 스키마를 보증하기 위하여 제약조건 확인에 필요한 의미적 지식을 제공한다. 이 방법에서는 내용, 구조와 함께 무결성 제약조건들은 동시에 유지되며, 또한 저장 중복성을 줄일 수 있다.

Abstract

Many techniques have been proposed to store efficiently and query XML data. One way achieving this goal is using relational database by transforming XML data into relational format. But most researches only transformed content and structure of XML schema. Although they transformed semantic constrainment of XML schema, they did not all of semantics. In this paper, we propose a systematic technique for extracting semantic constrainment from XML schema and storing method when the extracting result is transformed into relational schema without any lost of semantic constrainment. The transforming algorithm is used for extracting and storing semantic constrainment from XML schema and it shows how extracted information is stored according to schema notation. Also it provides semantic knowledges that are needed to be confirmed during the transformation to ensure a correct relation schema. The technique can reduce storage redundancy and can keep up content and structure with integrity constraints.

▸ Keyword : XML, XML Schema, 제약조건(Constraints), XML 스키마, 관계형 스키마, 매핑(Mapping), 변환(Transformation)

• 제1저자 : 조정길 교신저자 : 금영욱
• 투고일 : 2009. 04. 10, 심사일 : 2009. 04. 19, 게재확정일 : 2009. 05. 27.
* 성결대학교 컴퓨터공학부 교수
※ 이 논문은 2009년 성결대학교 특성화 사업(SKU_David-2009-001)의 재원으로 연구되었습니다.

Ⅰ. 서 론

웹은 정보의 보급과 분배에 중요한 매체가 되었으며, 웹 환경에서 정보 표현과 교환을 위한 표준 방식으로 널리 사용되고 있는 XML 문서의 양은 급격히 증가하고 있다. 따라서 이러한 XML 문서들을 효율적으로 저장하고 검색하는 연구가 활발히 진행되어 왔다. XML 문서를 효율적으로 관리하기 위해서는 XML과 데이터베이스 기술의 통합이 필요하다[1]. XML 데이터를 질의하는 방법은 XML 데이터를 관계형 데이터베이스 기법을 사용하여 변환과 저장으로 관계형 스토리지에 저장하는 것이다. 그러나 계층적인 XML 데이터 모델과 평평한(flat) 관계형 데이터 모델은 다르기 때문에 데이터 변환 작업은 어려움이 많다.

XML 데이터를 관계형 데이터베이스에 저장하기 위한 다양한 방법들이 연구되고 발표되었다. 그러나 대부분의 연구가 XML(DTD, XML Schema)의 내용과 구조만 변환하고 숨겨진 의미적 제약조건을 간과하거나 일부만 적용하였다. [2]는 함수적 종속성만 반영하고 제약조건을 간과하였으며, [1][7][10]에서는 DTD로부터 XML 문서의 내용과 구조만 추론하여 보존하였다. [9]에서는 DTD 문서에서의 의미적인 정보들을 보존하나 DTD의 단순화 절차가 생략되어 있어서 DTD의 복잡성을 다루지 못하고 있다.

이 논문에서는 주어진 XML Schema[13]로부터 내용과 구조 및 의미적 제약조건을 추출한다. 그리고 관계형 데이터베이스 스키마로 기술함으로써 XML Schema의 내용, 구조, 의미를 보존한다. 변환 알고리즘은 XML Schema로부터 의미적 제약조건을 추출하고 보존하는데 이용된다. 그리고 이러한 정보들을 스키마 표기법[14]에 따라 재작성하여 어떻게 의미적 제약조건을 보존하는지를 보여준다. 스키마의 변환 과정은 XML Schema로부터 내용과 구조 보존 과정을 통하여 관계형 스키마로 변환한다. 그리고 XML 표기법에 있는 여러 가지 의미적 제약조건들을 변환하는 과정에서 발견하여 관계형 표기법에 따라 제약조건을 재 작성한다.

이 논문은 XML Schema 기반의 XML 데이터를 XML 함수적 종속성과 관계된 의미적 제약조건에 따라 관계형 데이터로 변환하는데 의미적 제약조건 보존과 데이터의 중복성을 제거하는 기법을 제공한다. 이 논문의 구성은 다음과 같다. 2장에서는 관련 연구에 대하여 살펴본다. 3장에서는 XML Schema에서의 의미적 제약조건에 대하여 기술한다. 4장에서는 정보 보존 변환에 대하여 제시한다. 5장에서는 제시한 기법을 설명하고 증명하기 위하여 예제를 들었으며, 6장에서는 이 논문의 결론을 논한다.

Ⅱ. 관련연구

관계형 스키마 R은 릴레이션 스키마(S)와 의미적 제약조건(Δ)으로 이루어져 있으며 다음과 같이 정의한다[16].

정의 1.

관계형 스키마 R은 R=(S, Δ) 형식의 표기법이다:

릴레이션 스키마 S는 $r(a_1, \ldots, a_k)$와 같은 테이블 스키마의 집합이다. a_i 는 테이블 r에 있는 i번째 속성이다. 의미적 제약조건 Δ는 도메인 제약조건(domain constraints), 포함 종속(inclusion dependency), 동일성-생성 종속(equality-generating dependency), 투플-생성 종속(tuple-generating dependency)의 의미적 지식 집합이다.

참조 무결성 제약조건에서 릴레이션은 참조할 수 없는 외래키 값을 가져서는 안된다. 참조할 수 없는 외래키 값은 널(null)이 아니면서 참조된 릴레이션의 어떤 기본키 값과도 일치하지 않는 값을 의미한다. 만일 릴레이션 S의 기본키 K를 참조하는 외래키 FK가 릴레이션 R에 명시되어 있다면, 이 FK의 값은 반드시 S에 나타나있는 기본키 K의 어떤 값과 같든지 아니면 널(공백) 이어야 한다. 이 때 릴레이션 R과 S는 반드시 서로 다른 릴레이션이 되어야 하는 것은 아니다.

XML 스키마에서 관계형 스키마로의 변환에 관련된 많은 연구가 진행되었다[2][7]. [7]은 스키마의 테이블 생성에 초점을 맞춘 3가지 변환 알고리즘을 제공한다. 변환 알고리즘인 인라이닝 기법은 Basic Inlining, Shared Inlining, Hybrid Inlining이 있으며 관계형 스키마를 만드는 방법에서 차이가 있다. [9]는 [7]의 Hybrid Inlining 기법을 사용하며, XML key와 keyref 정보를 보존하는 제약조건 릴레이션 개념을 도입하여 키 정보를 보존한다. 또한 XML 문서에 있는 엘리먼트들 사이의 부모-자식 관계를 보존하기 위하여 자식 엘리먼트에 parent-id 속성을 추가하여 보존한다. [10]은 개선한 인라이닝 기법을 사용하며, 수정된 DTD 단순

화 규칙을 사용한다. [1]은 [7]의 Shared Inlining 기법에서 발생하는 중첩구조의 문제점을 보완한다. [2]는 [7]의 개선된 Hybrid Inlining 기법과 함수적 종속성을 반영하여 중복성을 제거한다.

스키마 변환에 관한 기존 연구들의 대부분은 XML 스키마로 DTD를 이용하고 있다[1][7][9][10]. 한정된 표현방식을 쓰는 DTD에 비하여 XML Schema[13]는 형식 제약조건과 더욱 복잡한 출현지시자 제약조건의 특징을 제공한다[2]. [7][10]에서는 DTD로부터 XML 문서의 내용과 구조만 추론하며 의미적인 정보의 보존을 고려하지 않고 있다. [9]에서는 DTD 문서에서의 의미적인 정보들을 생성되는 릴레이션으로 보존하는 기법을 보여주고 있으나 DTD의 단순화 절차가 생략되어 있어서 DTD의 복잡성을 다루지 못하고 있다. [1]에서는 [7][10]에서와 같이 내용과 구조만 매핑하고 의미적 제약조건을 간과하고 있다. [2]에서는 의미적인 정보를 보존함에 있어서 함수적 종속성을 제외한 제약조건을 고려하지 않고 있다. 이러한 문제점들을 보완하기 위하여 이 논문의 변환 연구는 DTD의 문제점을 극복하기 위해 제시된 XML Schema[13]를 사용하여 XML 스키마 변환을 하며, XML Schema로부터 내용, 구조 보존과 의미적 제약조건의 체계적인 추출/보존 방법을 제공하는데 [2]에 있는 변환 연구의 개선된 방법이다.

III. XML Schema에서의 의미적 제약조건

XML Schema[13]에서 추론할 수 있는 의미적인 정보와 관계형 스키마로 변환 될 수 있는 의미적 제약조건들은 다음과 같다.

3.1 도메인 제약조건

한 속성이 취할 수 있는 모든 값을 총칭해서 도메인이라 한다. 도메인 제약조건은 속성의 값이 특정 집합의 값으로써 한정될 때 확인할 수가 있다. XML Schema에서의 도메인 제약조건은 엘리먼트 내용과 속성에 대한 값 제한조건인 fixed와 default가 있으며, 속성의 출현 횟수를 지정하는 use가 있다. 예를 들어, 다음의 XML Schema에서 속성 Subscribe와 SecurityCleared의 도메인은 제한된다:

〈attribute name = "Subscribe" type="string" default="yes" /〉 〈attribute name = "SecurityCleared" type="boolean" fixed="true" /〉.

앞의 XML Schema를 릴레이션 스키마로 변환하면 다음과 같이 SQL CHECK 절을 사용하여 도메인 제약조건을 보존할 수가 있다: CREATE DOMAIN Subscribe VARCHAR(10) CHECK (VALUE IN ("yes")) CREATE DOMAIN SecurityCleared VARCHAR(10) CHECK (VALUE IN ("true")).

강제적인(mandatory) 속성이 XML Schema에 있는 키워드 required에 의해 정의될 때, 변환된 릴레이션 스키마에서도 다음과 같이 반드시 값이 존재 하도록 하는 것이 필요하다:

〈element name = "CreditAccount" 〉 〈attribute name ="dateReceived" use="required"/〉.

속성 X가 널(null)이 될 수 없다는 것을 표시하는데 표기법 "$X \nrightarrow 0$"를 사용한다. 도메인 제약조건의 이러한 종류는 다음과 같이 SQL에서 NOT NULL 절을 사용하여 보존한다:

CREATE TABLE CreditAccount (dateReceived VARCHAR(20) NOT NULL).

3.2 엘리먼트 출현지시자(Cardinality) 제약조건

엘리먼트 선언에서 속성의 형태를 나타낼 수 있는 두개의 출현지시자 제약조건인 minOccurs와 maxOccurs 속성은 다음과 같이 선언된다:

〈element name="MiddleInitial" type="xs:string" minOccurs="1" maxOccurs="unbounded"/〉.

다음의 표 1은 XML Schema의 속성인 minOccurs와 maxOccurs를 이용하는 방법을 보여준다.

표 1. 4가지 가능한 출현지시자 관계
Table 1. 4 possible cardinality relationships

minOccurs값	maxOccurs값	의미
1	1	오직 한번만 출현
0	1	없거나 한번만 출현
0	unbounded	영(0)번이상 출현
1	unbounded	적어도 한번이상 출현

출현지시자 제약조건에서는 대체로 세 가지의 제약조건을 추론할 수가 있다. 첫째, 서브 엘리먼트가 널(null)

이 되거나 안 되거나 하는 경우이다. 속성의 경우와 유사하게 엘리먼트 X가 널이 될 수 없음을 표시하는 데는 표기법 "$X \nrightarrow 0$"를 사용한다. 이 제약조건은 NULL이나 NOT NULL 절에 의해 쉽게 실행된다. 둘째, 서브 엘리먼트가 없거나 하나만 생길 수 있는데, 이것은 동일성-생성 종속성의 한 종류인 싱글톤(singleton) 제약조건[15]으로 알려져 있다. 셋째, 엘리먼트를 주었을 때 서브 엘리먼트가 적어도 한개 이상 생길수가 있다. 이것은 투플-생성 종속성의 한 종류이다.

세 가지의 제약조건을 정리하면 표 2와 같으며, 두 번째와 세 번째 제약조건인 경우는 다음 절에서 정의한다.

표 2. 출현지시자와 관계된 의미적 제약조건
Table 2. cardinality and their corresponding semantic constraints

출현지시자		심볼	not null	동일성-생성 종속성	투플-생성 종속성
minOccurs값	maxOccurs값				
1	1		Y	Y	Y
0	1	?	N	Y	N
0	unbounded	*	N	N	N
1	unbounded	+	Y	N	Y

3.3 포함 종속

포함 종속은 XML 문서내의 특정 속성의 값이 다른 속성의 값으로 나타나야만 한다라는 의미로써, 참조 무결성의 일반화된 개념이라 할 수가 있다. XML Schema에 기반을 둔 포함 종속의 당연한(trivial) 형식은 "엘리먼트 X와 그것의 서브 엘리먼트 Y를 주면, Y는 X에 포함되어야만 한다($Y \subseteq X$)"이다. 예를 들어, 다음의 포함 종속은 그림 1의 XML Schema 안에 있는 conference 엘리먼트와 그것의 4개의 서브 엘리먼트인 title, date, editor, paper로부터 conference가 널이 아닌 경우에 나타난다: {conference.title $\subseteq$ conference, conference.date $\subseteq$ conference, conference.editor $\subseteq$ conference, conference.paper $\subseteq$ conference}.

XML Schema에서 포함 종속을 추론할 수가 있는 요소는 ID, IDREF(S), key, keyref 속성이 있다. 이는 관계형 스키마의 기본키와 외래키의 개념으로 보존이 될 수 있다.

```
<xs:element name="conference">
<xs:complexType> <xs:sequence>
<xs:element ref="title" />
<xs:element ref="date" />
<xs:element ref="editor" minOccurs="0" />
<xs:element ref="paper" minOccurs="0" maxOccurs="unbounded" />
</xs:sequence>
<xs:attribute name="id" type="xs:ID" use="required" />
</xs:complexType> </xs:element>
<xs:element name="editor">
<xs:complexType> <xs:sequence>
<xs:element ref="person" minOccurs="0" maxOccurs="unbounded" />
</xs:sequence>
<xs:attribute name="eids" type="xs:IDREFS" use="optional" />
</xs:complexType> </xs:element>
<xs:element name="paper">
<xs:complexType> <xs:sequence>
<xs:element ref="title" />
<xs:element ref="contact" minOccurs="0" />
<xs:element ref="author" />
<xs:element ref="cite" minOccurs="0" />
</xs:sequence>
<xs:attribute name="id" type="xs:ID" use="required" />
</xs:complexType> </xs:element>
<xs:element name="contact">
<xs:complexType>
<xs:attribute name="aid" type="xs:IDREF" use="required" />
</xs:complexType> </xs:element>
<xs:element name="author">
<xs:complexType> <xs:sequence>
<xs:element ref="person" maxOccurs="unbounded" />
</xs:sequence>
<xs:attribute name="id" type="xs:ID" use="required" />
</xs:complexType> </xs:element>
<xs:element name="person">
<xs:complexType> <xs:sequence>
<xs:element ref="name" />
<xs:choice>
<xs:element ref="email" />
<xs:element ref="phone" />
</xs:choice> </xs:sequence>
<xs:attribute name="id" type="xs:ID" use="required" />
</xs:complexType> </xs:element>
<xs:element name="cite">
<xs:complexType> <xs:sequence>
<xs:element ref="paper" minOccurs="0" maxOccurs="unbounded" />
</xs:sequence>
<xs:attribute name="format" use="optional">
<xs:simpleType>
<xs:restriction base="xs:NMTOKEN">
<xs:enumeration value="ACM" />
<xs:enumeration value="IEEE" />
</xs:restriction></xs:simpleType></xs:attribute>
<xs:attribute name="id" type="xs:ID" use="required" />
</xs:complexType></xs:element></xs:schema>
```

그림 1. 컨퍼런스 XML 스키마
Fig 1. Conference XML Schema

예를 들어, 그림 1의 XML Schema에서 contact과 editor 엘리먼트는 person 엘리먼트의 id 속성이 기본키이기 때문에 contact 엘리먼트의 aid 속성을 제한한다. 또한 editor 엘리먼트의 eids 속성은 person 엘리먼트의

다중(multiple) id 속성으로 지정될 수가 있다. 결과로써 다음과 같이 포함 종속을 나타낼 수가 있다: {editor.eids ⊆ person.id, contact.aid ⊆ person.id}.

포함 종속은 만약 참조되어진 속성이 기본키이면 외래키 개념에 의해 최선으로 시행될 수 있다. 그렇지 않으면 SQL에서 CHECK, ASSERTION, 혹은 TRIGGERS 기능사용이 필요하다.

3.4 동일성-생성 종속

싱글톤 제약조건은 서브 엘리먼트가 없거나 한번만 출현하는 엘리먼트로 제한한다. 엘리먼트 타입 X가 그것의 서브 엘리먼트 타입 Y를 위하여 싱글톤 제약조건을 만족시킬 때, 만약 타입 X의 엘리먼트 인스턴스가 타입 Y의 두개의 서브 엘리먼트 인스턴스 y_1과 y_2를 가지고 있으면, 그때 y_1과 y_2는 같아야 한다. 이 프로퍼티는 동일성-생성 종속이라 하고, 데이터베이스 이론에서는 "X → Y"로 표기한다. 일반적으로 동일성-생성 종속은 엘리먼트 출현지시자 제약조건에 있는 minOccurs="1" maxOccurs="1" 과 minOccurs="0" maxOccurs="1" 인 매핑의 경우에 발생한다. 예를 들어, 두개의 동일성-생성 종속 :
{conference → conference.title, conference → conference.date}는 그림 1에 있는 conference 엘리먼트로부터 추론할 수가 있다. 이러한 동일성-생성 종속은 SQL UNIQUE 구문에 의해 보존될 수 있다.

3.5 투플-생성 종속

관계형 모델에서의 투플-생성 종속은 일정한 형식인 몇몇의 튜플들이 테이블에서 제공되는 것을 필요로 하며, "↠" 심볼을 사용한다. XML Schema에서 투플-생성 종속의 두 가지 형식은 부모 제약조건과 자식 제약조건[15]이다.

3.5.1 부모 제약조건

"Child ↠ Parent"는 Child의 모든 엘리먼트가 Parent인 부모 엘리먼트를 가져야 되는 것을 말한다. XML Schema에서는 루트의 표기법이 없으며, XML 문서는 그것의 부모 엘리먼트의 명세가 없이도 엘리먼트의 어느 단계로부터도 시작할 수 있다. 그런 까닭에 부모 제약조건은 XML Schema 구조에서 보는 것만으로 간단하게 보증할 수가 없기 때문에 얼마간의 의미적 지식을 요구한다. 예를 들어, 그림 1에 있는 XML Schema에서 editor와 date 엘리먼트는 그들의 부모로서 conference 엘리먼트를 가질 수 있다. 또한 모든 XML 문서가 editor나 date 단계가 아니라 conference 엘리먼트 단계에서 시작한다면, 부모 제약조건 {editor, date} ↠ conference는 유지된다. title ↠ conference는 title 엘리먼트가 conference나 paper 엘리먼트의 서브 엘리먼트가 될 수 있기 때문에 부모 제약조건이 유지될 수 없다.

3.5.2 자식 제약조건

"Parent ↠ Child"는 Parent의 모든 엘리먼트가 Child인 적어도 한 개의 자식 엘리먼트를 가져야 되는 것을 의미한다. 이것은 출현지시자 제약조건에서 minOccurs="1" maxOccurs="1" 과 minOccurs="1" maxOccurs="unbounded"의 경우이다. 예를 들어 그림 1의 XML Schema에서 conference 엘리먼트가 title과 date 서브 엘리먼트를 포함하므로, 자식 제약조건 conference ↠ {title, date}는 유지된다.

IV. 정보 보존 변환

이 장에서는 XML 문서의 스키마인 XML Schema를 입력으로 받아서 정보 보존 관계형 스키마를 생성하는 알고리즘을 기술한다. XML Schema는 문서의 구조에 대한 정보와 의미적 제약조건으로 구성되어 있다. 생성된 관계형 스키마는 함수적 종속성과 제약조건을 반영하고 중복성이 제거된 릴레이션의 조합으로 이루어진다.

XML 스키마를 관계형 스키마로 매핑하는 변환 절차는 그림 2와 같으며, XML 스키마에서 관계형 스키마로 매핑하는 전체 과정을 나타낸 것이다. XML Schema로 내용과 구조 보존 과정을 통하여 관계형 스키마에 변환한다. 그리고 변환하는 과정에서 XML 표기법에 따른 여러 가지 의미적 제약조건들을 발견하고 관계형 표기법에 따라 제약조건을 재 작성한다.

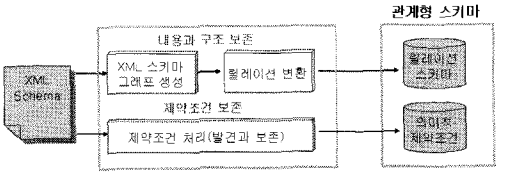

그림 2. 관계형 스키마 매핑 흐름도
Fig 2. Relational Schema Mapping Fflow

이 논문에서는 관계형 스키마를 생성하는 방법에 다음의 데이터 구조를 사용한다.

정의 2.

XML 스키마 그래프 G는 노드와 에지(edge)의 쌍 (Δ, ε)이다. Δ는 유한 집합이고 ε는 Δ에서의 이진 릴레이션이다. 집합 Δ는 XML Schema에 있는 엘리먼트와 속성이다. 각각의 에지 $e \in \varepsilon$ 는 출현지시자 관계 타입으로 이어져 있다. 또한 각각의 노드 $v \in \Delta$ 는 다음의 정보를 전달한다.

- inedge : 인입 에지의 수 저장.
- eletype : XML Schema의 내용 모델에서 엘리먼트 타입 이름 저장.
- tagvalue : 엘리먼트나 속성인(만약 속성이면 ID나 IDREF와 같은 속성 키워드를 포함) 노드의 플래그 값을 저장.
- status : 만약 노드가 깊이 우선 탐색으로 방문했다면 "visited" 값을 저장하고 아니면 "non-visited" 값을 저장.

4.1 내용과 구조 보존 매핑

함수적 종속성을 반영하여 관계형 스키마로 매핑하는 절차는 다음과 같다[2].

먼저 XML Schema로 XML Schema 그래프를 그린다(그림3). 그리고 각 $(Q, [P_{x1}, P_{x2}, \ldots, P_{xn} \rightarrow P_y])$에 대하여, P_{x1}, P_{x2}, ..., P_{xn}, P_y로부터 축출된 애트리뷰트들의 릴레이션을 생성하고, 키로서 P_{x1}, P_{x2}, ..., P_{xn}에 대응되는 애트리뷰트들을 상술하고, 루트로부터 P_y까지의 경로 중에 마지막 에지를 마크한다. 다음에 진입 차수(in-degree)가 0인 엘리먼트 노드는 개별적인 릴레이션을 생성한다. 또한 순환(recursion) 형태인 사이클이 형성되어 있는 경우에는 개별적인 릴레이션을 생성한다. 주어진 데이터 형식에 포함되어 있는 엘리먼트의 출현 지시자인 maxOccurs의 값이 2 이상일 때 독립적인 릴레이션을 생성한다. 진입 차수가 2 이상인 엘리먼트 노드는 독립적인 릴레이션을 생성하며, 진입차수가 1인 노드들은 인라인한다. 그리고 복잡 형식들의 유도 관계에서 확장에 의한 유도인 경우에 상속으로 연결된 최종 노드인 조상 노드는 개별적인 릴레이션을 생성한다. 마지막으로 자식 릴레이션에서 각각의 부모-자식 관계가 부모 id에 의해 기록된 것을 보증한다.

그림 3은 그림 1인 컨퍼런스 XML 스키마를 XML 스키마 그래프로 표현한 것이다. 상세한 내용에 대해서는 [2]를 참조하기 바란다.

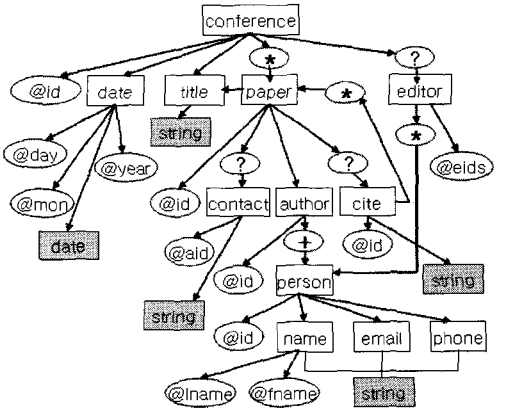

그림 3. XML 스키마 그래프
Fig 3. XML Schema Graph

4.2 제약조건 보존 매핑

표 1에 있는 출현지시자 관계는 엘리먼트 대 서브 엘리먼트 관계뿐만 아니라 엘리먼트 대 속성 관계도 고려하였다. 예를 들어, 다음의 XML Schema로부터,

```
〈xs:element name="X"〉 〈xs:complexType〉
〈xs:attribute name="Y" type="xs:string"
use="optional" /〉
〈xs:attribute name="Z" type="xs:string"
use="required" /〉
〈/xs:complexType〉 〈/xs:element〉
```

출현지시자 관계의 두 종류(표 2에서, 엘리먼트 X와 속성 Y에서의 심볼 "?", 엘리먼트 X와 속성 Z에서의 심볼 "blank")를 추론할 수가 있다.

3장에서 추론한 의미적인 정보인 엘리먼트 출현지시자 관계는 체계적인 방식으로 의미적 제약조건을 찾는데 사용된다. 표 2는 출현지시자 관계로부터 추론할 수 있는 세 가지의 중요한 의미적 제약조건을 요약했다. 제약조건 처리 알고리즘은 표 2에 있는 속성으로부터 XML Schema에 있는 의미적 제약조건을 획득할 수 있다.

제약조건 처리 알고리즘에 의해 발견된 의미적 제약조건은 새롭게 생성된 관계 스키마로부터 올바른 의미를 수행하여야 하며, XML 술어에서의 의미적 제약조건을 관계형 술어로 재 작성되는 것이 필요하다. 다음의 그림 4에 있는 제약조건 처리 알고리즘은 XML Schema에서 출현지시자 관계를 고려하여 의미적 제약조건을 찾는다.

1. XML 스키마 그래프에서 제약조건을 발견할 때 까지 깊이-우선 탐색 방법으로 자세히 조사한다. (그림 3의 XML 스키마 그래프로부터 표 2에 있는 조건에 만족하는 제약조건을 찾는다.)
2. (입력 : XML 스키마 그래프의) 노드 n 이
   2.1 "?"인 경우: X → X.Y
   2.2 "*"인 경우: 의미적 제약조건이 없음
   2.3 "+"인 경우: X ↛ 0, X → X.Y
   2.4 없는 경우: X ↛ 0, X → X.Y, X ↠ X.Y
3. X ↛ 0인 경우(not null)
   테이블 스키마 A에서 X가 속성 X'로 사상되면, A[X']는 널이 되어서는 안 된다.
4. Y ⊆ X인 경우(포함 종속)
   테이블 스키마 A, B에서 X, Y가 속성 X', Y'로 사상되면, A[X'] ⊆ B[Y']로 재 작성한다.
5. X → X.Y인 경우(동일성-생성 종속)
   엘리먼트 X, Y가 같은 테이블 스키마 A로 사상되고 Z가 A의 키 종속이면, A[Z] → A[Y]로 재 작성한다.
6. X ↠ X.Y인 경우(투플-생성 종속)
   6.1 만약 엘리먼트 X, Y가 같은 테이블에 사상되면, A를 테이블로 하고 Z는 A의 키 속성이 되게 한다. 그때 A[Z] ↠ A[Y]로 재 작성한다.
   6.2 그렇지 않으면(엘리먼트 X, Y가 다른 테이블에 사상되면), 테이블은 A, B로 하고 Z는 A의 키 속성이 되게 한다. 그때 B[fk_A] ⊆ A[Z]로 재 작성한다.
7. XML 스키마 그래프의 EOF까지 2~6번 반복

그림 4. 제약조건 처리 알고리즘
Fig 4. Constraints Processing Algorithm

제약조건 처리 알고리즘에서 2번은 입력되는 노드가 표2의 조건에 만족하는 의미적 제약조건을 구별한다. 3번에 조건이 맞는 경우에는(X ↛ 0인 경우) "CREATE TABLE A(... X' NOT NULL ...)" SQL 문이 실행되는 것과 같다. 4번에 조건이 맞는 경우에는(Y ⊆ X인 경우) "CREATE TABLE A(... FOREIGN KEY (X') REFERENCES B(Y') ...)" 혹은 "CREATE TABLE A(... (X') CHECK (X' IN (SELECT Y' FROM B)) ...)" SQL 문이 실행되는 것과 같다. 5번 조건이 맞는 경우에는(X → X.Y인 경우) "CREATE TABLE A(... UNIQUE (Y) PRIMARY KEY (Z) ...)" SQL 문이 실행되는 것과 같다. 6번의 6.1 조건이 맞는 경우에는 "CREATE TABLE A(... Y NOT NULL, PRIMARY KEY (Z) ...)" SQL 문이, 6.2 조건이 맞는 경우에는 "CREATE TABLE B(... FOREIGN KEY (fk_A) REFERENCES A(Z) ...)" SQL 문이 실행되는 것과 같다.

알고리즘은 XML 스키마 그래프에서 제약조건을 발견할 때까지 깊이-우선 탐색 방법으로 자세히 조사하며 리프 노드에서 새로운 필드를 인라인한다. 마지막 출력 스키마는 모든 테이블 스키마와 의미적 제약조건의 합집합이다.

V. 사례 연구

내용과 구조 보존 알고리즘과 제약조건 보존 알고리즘이 어떻게 예제에서 실행되는지를 실례로 설명한다. 그리고 그림 1의 컨퍼런스 XML 스키마로부터 획득한 XML 스키마 그래프[2]는 그림 4에서 제공한다. 또한 [2]에서 정의한 방법으로 XML 스키마 그래프로 내용과 구조 보존을 보증한 최종 릴레이션 스키마가 그림 5와과 같이 생성된다.

마찬가지로 그림 1의 컨퍼런스 XML 스키마를 가지고 그림 4의 알고리즘에 따라 제약조건 보존을 보증한 관계형 표기법의 의미적 제약조건은 다음의 표 3과 같이 생성된다.

conference(#id, date.day, date.mon, date.year, title, editor.eids, person.name.lname, person.name.fname, parentID);
paper(#id, paper.id, contact.aid, author.id, cite.id, parentID);
person(#id, person.id, person.name.lname, person.name.fname, parentID);
FD1(conference.paper.id, conference.paper.author.id, conference.paper.author.person.id,conference.paper.author.person.name);
FD2(conference.paper.author.person.name,conference.paper.author.person.email);
FD3(conference.paper.author.person.name,conference.paper.author.person.phone);

그림 5. 릴레이션 스키마
Fig 5. Relational Schema

표 3은 XML 스키마로부터 관계형 스키마로 재 작성되어 추가되는 의미적 제약조건이다. 생성된 릴레이션 스키마뿐만 아니라 의미적 제약조건은 NOT NULL, KEY, UNIQUE 혹은 CHECK 구문 사용에 의해 보증된다.

릴레이션 스키마(그림 5)와 의미적 제약조건(표 3)이 최종적으로 완성되었다. 그림 5에 있는 릴레이션 스키마들은 내용과 구조의 손실이 없는 함수적 종속성을 만족시키며, 또한 제3 정규형도 만족시킨다. 그리고 표 3에 있는 의미적 제약조건들 역시 의미의 손실이 없는 무결성

표 3. 의미적 제약조건
Table 3. Semantic Constraints

종류	의미적 제약조건
포함 종속	conference.editor.eids(eids) ⊆ person(id), paper(contact.aid) ⊆ person(id)
not null	conference(id,date.year,date.mon,date.day,title,root_element) ↛ 0 conference.editor.eids(id,root_element) ↛ 0 paper(id,title,root_element) ↛ 0 person(id,name.lname,root_element) ↛ 0
동일성-생성 종속성	conference(id) → conference(title,date.year,date.mon,date.day) paper(id) → conference(title,contact.aid,cite.id) person(id) → conference(name.lname,name.fname,email)
투플-생성 종속성	conference(id) ↠ conference(title,date.year,date.mon,date.day) paper(id) ↠ conference(title,contact.aid,cite.id) person(id) ↠ conference(name.lname,name.fname,email) conference.editor.eids(fk_conference) ⊆ conference(id) paper(fk_conference) ⊆ conference(id) paper(fk_cite) ⊆ paper(cite.id) person(fk_conference) ⊆ conference(id) person(fk_paper) ⊆ paper(id)

제약조건을 만족시키며, 또한 제3 정규형도 만족시킨다.

기존의 Hybrid Inlining 기법을 적용하여 생성된 릴레이션은 XML Schema에서 추론할 수 있는 NULL, NOT NULL, CHECK 절과 같은 제약조건을 보존하지 못하기 있기 때문에 XML 문서의 저장 시에 데이터 무결성을 위해 저장 프로시저나 트리거를 이용해야 하는 번거로움이 생기게 된다. 그러나 이 논문에서는 이러한 의미적 제약조건들의 보존을 보장함으로써 XML 문서를 저장할 때에 데이터 무결성을 보장하기 위하여 저장 프로시저나 트리거를 사용할 필요가 없다.

VI. 결론

관계 데이터베이스 이론에서 데이터들 간에 존재하는 함수적 종속성과 무결성 제약조건은 중요한 개념이다. 따라서 계층적인 XML 구조를 어떻게 평평한 관계형 구조로 적용할 수가 있는가가 관건이다. 계층적 데이터에서 데이터 중복은 피할 수 없는 현상이다. 그러나 XML FD 정의와 제약조건에 의해 중복된 정보를 기술할 수가 있으며, 생성되는 릴레이션 스키마에서 데이터 중복 문제를 최소화 하는 것이 가능하다.

XML Schema는 DTD의 부족분을 매우기 위하여 W3C에 의하여 제안되었고, XML Schema의 장점으로 인하여 앞으로 급속도로 DTD를 XML Schema로 대체하게 될 것이다. XML Schema는 형식 제약조건과 더욱 복잡한 출현지시자 제약조건의 특징을 제공하기 때문에 관계형 스키마로 변환하는데 많은 어려움이 있다.

이에 이 논문에서는 XML Schema를 구조적이고 의미적 관점 하에 관계형 스키마로 변환하는 방법을 제공하였다. XML Schema에 숨겨져 있는 의미적 제약조건을 제약조건 처리 알고리즘을 사용하여 발견하고, 관계형 표기법으로 재 작성 하였다. 이 논문의 실험결과인 사례연구에서는 XML Schema에서 구조뿐만 아니라 의미적 제약조건이 릴레이션 스키마로 변환하는 동안에 체계적으로 보존되는 것으로 나타났다. 이러한 제약조건은 의미적 질의 최적화로 사용될 수가 있다.

참고문헌

[1] 홍은지, 이영호, "엘리먼트의 중첩 문제를 해결한 Shared Inlining 저장 기법," 한국정보과학회논문지: 데이터베이스, 제35권, 제5호, 411-420쪽, 2008년 10월.

[2] 조정길, "함수적 종속성을 반영한 XML 문서의 관계형 스키마 매핑 기법," 한국인터넷정보학회논문지, 제8권, 제2호, 95-103쪽, 2007월 4월.

[3] Sven Hartmann, Thu Trihh, "Axiomatising Functinal Dependencies for XML with Frequencies,"

FoIKS 2006, LNCS 3861, pp. 159-178, Feb. 2006.

[4] Irena Mlynkova, Jaroslav Pokprny, "UserMap-an Adaptive Enhancing of User-Driven XML-to-Relational Mapping Strategies," ADC2008, Wollongong, Australia, pp. 165-174, Jan. 2008.

[5] Jana Bauckmann, "Efficienty Identifying Inclusion Dependencies in RDBMS," Proc. of the 22nd International Conference on Data Engineering Workshops (ICDEW'06), LNCS 823, Apr. 2006.

[6] Lee ML, Ling TW, Low WL, "Designing functional dependencies for XML," Jensen CS, et al., eds. Advences in Database Technology--EDBT 2002, 8th International Conference on Extending Database Technology, Lecture Notes in Computer Science 2287, Prague, Czech Republic:Springer-Verlag, pp. LNCS 2287, 145-158, Jan. 2002.

[7] Shanmugasundaram, J., Tufte, K., He, G., Zhang, C., DeWitt, D., Naughton, J. "Relational Databases for Query XML Documents: Limitations and Opportunities," Proc. VLDB'99, Edinburgh, Scotland, pp. 302-314, Sep. 1999.

[8] 이혜자, 정병수, 김대호, 이영구, "경로 정보의 중복을 제거한 XML 문서의 저장 및 질의처리 기법," 한국정보처리학회논문지D, 제13-D권, 제5호, 663-672쪽, 2005년 10월.

[9] Chen Y, Davidson S, Zheng Y, "Constraint Preserving XML Storage in Relation," In WebDB, 2002.

[10] Lu S, Sun Y, Atay M, Fotouhi F, "A New Inlining Algorithm for Mapping XML DTDs to Relational Schemas," Proc. of the 1st International Workshop on XML Schema and Data management, LNCS 2814, pp. 366-377, Sep. 2003.

[11] 이상태, 임종선, 주경수, "관계형 DBMS를 이용한 XML 스키마 기반에 XML DBMS 설계," 한국컴퓨터정보학회논문지, 제9권, 제4호, 19-26쪽, 2004년 12월.

[12] 박준범, 박경수, 오수열, "ODMG 객체 모델 기반의 XML 문서 저장 관리 시스템에 관한 연구," 한국컴퓨터정보학회논문지, 제8권, 제2호, 16-23쪽, 2003년 6월.

[13] XML Schema Part 1: Structures, *http://www.w3.org/TR/xmlschema-1/*

[14] XML Path Language(XPath), *http://www.w3.org/TR/xpath*,

[15] Wood, P. T. "Optimizing Web Queries Using DTD," Proc. 2nd Int'l Workshop on Web Information and Data Management(WIDM), pp. 28-32, Nov. 1999.

[16] 이석호, "*데이터베이스 시스템*," 정익사, 125-126쪽, 2006년 1월.

저 자 소 개

조 정 길

1987: 숭실대학교 전산학과 공학사
1993: 숭실대학교 정보과학대학원 석사
2003: 충북대학교 전산학과 이학박사
현재: 성결대학교 컴퓨터공학부 교수
관심분야 : XML 문서관리, 정보 검색, 시멘틱 웹

금 영 욱

1978: 서울대학교 수학과 학사
1992: 서강대학교 전산학과 석사
1997: 서강대학교 전산학과 박사
현재: 성결대학교 컴퓨터공학부 교수
관심분야 : 병렬/분산처리, 컴포넌트 소프트웨어 개발, XML 문서관리