XML 문서의 변환을 위한 온톨로지 갱신 기반 XML 스키마 매칭

(XML Schema Matching based on Ontology Update for the Transformation of XML Documents)

이 경 호 † 이 준 승 ††
(Kyong-Ho Lee) (Jun-Seung Lee)

요 약 서로 다른 XML 스키마로 작성된 XML 문서간의 변환을 위해서는 두 스키마 사이의 의미적 연관관계를 계산하는 스키마 매칭 과정이 필수적이다. 본 논문에서는 XML 문서의 변환을 위한 효율적인 스키마 매칭 알고리즘을 제안한다. 제안된 알고리즘은 두 단계로 구성된다. 먼저 제안된 온톨로지와 어휘 유사도에 기반하여 단말노드 사이의 후보매칭을 계산한다. 또한 문맥 정보를 반영하는 제안된 경로 유사도 비교를 통해 후보매칭관계 중에서 최종 매칭 결과를 선택한다. 특히 제안된 방법은 기존 연구와 달리 사용자 피드백에 의해 점증적으로 갱신되는 온톨로지에 기반한다. 제안된 온톨로지는 IsA나 PartOf와 같은 다양한 관계를 표현할 수 있기 때문에 일대일 매칭은 물론이고 다대일 및 일대다 관계의 복합매칭을 계산할 수 있다. 제안된 알고리즘의 성능 평가를 위해 다양한 도메인의 XML 스키마를 대상으로 실험한 결과, 평균 97%의 정확률과 83%의 재현율을 나타내어 기존 연구보다 우수하였다. 특히 제안된 온톨로지의 갱신을 통하여 약 9%의 성능 향상을 확인할 수 있었다.

키워드 : XML, 스키마 매칭, 온톨로지, 복합매칭

Abstract Schema matching is important as a prerequisite to the transformation of XML documents. This paper presents a schema matching method for the transformation of XML documents. The proposed method consists of two steps: preliminary matching relationships between leaf nodes in the two XML schemas are computed based on proposed ontology and leaf node similarity, and final matchings are extracted based on a proposed path similarity. Particularly, for a sophisticated schema matching, the proposed ontology is incrementally updated by users' feedback. Furthermore, since the ontology can describe various relationships between concepts, the proposed method can compute complex matchings as well as simple matchings. Experimental results with schemas used in various domains show that the proposed method is superior to previous works, resulting in a precision of 97 % and a recall of 83 % on the average. Furthermore, the dynamic ontology increased by 9 percent overall.

Key words : XML, Schema matching, Ontology, Complex matching

1. 서 론

XML(eXtensible Markup Language)[1] 문서는 논리적 구조정보를 표현할 수 있으며 플랫폼에 독립적이라는 장점 때문에 다양한 분야에서 정보의 교환을 위한 표준으로 널리 사용되고 있다. 한편 XML은 설계자가 마음대로 XML 스키마[1]를 설계할 수 있다는 유연성을 갖기 때문에 동일한 분야에서 조차 서로 다른 구조의 XML 스키마가 사용되고 있다. 따라서 서로 다른 스키마에 따라 작성된 XML 문서를 교환하기 위해서는 서로의 스키마에 적합하도록 XML 문서간의 변환을 수행하여야 한다. 최근 XML 스키마가 급증함에 따라 XML 문서간의 자동변환이 중요한 이슈로 떠오르고 있다.

· 이 논문은 2003년도 한국학술진흥재단의 지원에 의하여 연구되었음 (KRF-2003-003-D00429)
† 종신회원 : 연세대학교 컴퓨터과학과 교수
khlee@cs.yonsei.ac.kr
(Corresponding author임)
†† 정 회 원 : 연세대학교 컴퓨터과학과
jslee@icl.yonsei.ac.kr
논문접수 : 2005년 2월 21일
심사완료 : 2006년 8월 24일

1) 본 논문에서 XML 스키마는 XML DTD(Document Type Definition)[1]와 XML Schema[2]를 모두 포괄하는 의미로 사용된다.

XML 문서 변환을 위한 기존 방법은 XML 문서를 정의하고 있는 XML 스키마 사이에 의미적 연관관계를 찾고 연관관계를 이용하여 변환용 XSLT 스크립트를 생성한다[4-6]. 그러나 크기가 큰 스키마를 대상으로 할 경우, 스키마 사이에 연관관계를 계산하는 스키마 매칭 과정은 많은 비용을 요구하며 높은 오류 발생률을 갖는다. 이러한 문제를 해결하기 위해선 스키마 사이의 매칭 관계를 자동으로 계산하는 연구가 필요하다. 따라서 본 논문에서는 XML 문서변환을 위하여 XML 스키마 사이의 정확한 매칭관계를 계산하는 방법을 제안한다.

기존의 스키마 매칭에 관한 연구의 대부분은 정확률이 비교적 낮아 문서변환을 위한 스키마 매칭에 적합하지 않다. 예를 들어, 기존의 스키마 매칭 연구는 몇몇 유사도 값들을 단순히 합하여 매칭을 선택하여 잘못된 매칭을 계산할 수 있으며 경우에 따라 추가적인 선택을 요구하는 다대다(*many-to-many*) 관계의 매칭을 계산한다.

본 논문에서는 정교한 수준의 스키마 매칭을 위해 두 단계로 구성된 매칭 알고리즘을 제안한다. 제안된 방법은 어휘 유사도(lexical similarity) 에 기반하여 단말노드 사이의 후보매칭을 계산하고, 문맥 정보를 반영하는 경로 유사도(path similarity) 비교를 통해 후보매칭 중에서 최종 매칭을 선택함으로 기존 연구에 비해 정확한 매칭을 계산한다.

특히, 매칭에 사용되는 부가정보를 위한 온톨로지 구조를 제안한다. 기존 연구는 정적인 사전형태의 부가정보를 사용하여 구축비용이나 도메인에 따라 다양하게 적용될 수 없는 단점이 있는 반면에 제안된 온톨로지 구조는 동일한 도메인의 매칭 결과를 기반으로 갱신됨으로 보다 정확한 부가정보를 제공한다.

특히 제안된 온톨로지는 어휘사이의 유사도 뿐만아니라 일반화 부분합 관계를 기술할 수 있는 구조로, 이를 활용하여 일대일의 단순매칭(simple matching) 뿐만 아니라 문서 변환시 분할(spilt) 또는 병합(merge) 연산에 해당하는 일대다(one-to-many) 또는 다대일(many-to-one) 의 대응관계를 계산할 수 있는 방법을 제안한다.

제안된 방법의 성능을 평가하기 위해 다양한 스키마를 대상으로 실험을 하였다. 실험에 사용된 스키마는 성능 비교를 위해 기존의 스키마 매칭 연구에 사용된 스키마는 물론이고, 기존 실험데이타 보다 크고 복잡한 스키마를 대상으로 하였다. 실험 결과, 평균 97%의 정확률과 83%의 재현율을 나타내어 기존 연구보다 우수하였다. 특히 제안된 온톨로지의 갱신을 통하여 약 9%의 성능 향상을 확인할 수 있었다.

본 논문의 구성은 다음과 같다. 2절에서는 XML 스키마를 효과적으로 표현할 수 있는 문서모델을 정의하고, 스키마 매칭에 대한 개념과 기존에 진행된 스키마 매칭에 관한 연구들을 간략히 소개한다. 3절에서는 본 논문에서 제안한 온톨로지의 구조와 온톨로지 갱신을 위한 연산의 종류를 자세히 기술한다. 4절에서는 스키마 매칭 알고리즘을 단말노드간의 후보 매칭 계산과 경로 유사도에 기반한 최종 매칭 추출의 두 단계로 구분한 후, 각 단계에 대한 자세한 기술을 제공한다. 5절에서는 제안된 방법을 평가하기 위한 실험 결과와 분석을 기술한다. 끝으로 6절에서는 결론 및 향후연구 방향을 기술한다.

2. 문서모델 및 관련연구

본 절에서는 XML 스키마를 표현하기 위해 제안한 문서 모델을 소개한다. 또한 스키마 매칭에 대한 개요와 현재까지 진행된 스키마 매칭에 관한 연구를 간략히 기술한다.

2.1 스키마 매칭

일반적으로 XML 문서의 변환 과정은 그림 1과 같이 스키마 매칭과 변환용 XSLT(eXtensible Stylesheet Language Transformations)[3] 스크립트 생성의 두 단계로 구성된다. 스키마 매칭 단계에서는 소스 스키마(source schema)와 타겟 스키마(target schema)를 입력으로 받아 의미적인 연관 관계를 계산하고, 변환 스크립트 생성 단계에서는 계산된 매칭관계를 이용하여 XML 문서를 변환시킬 수 있는 스크립트를 생성한다.

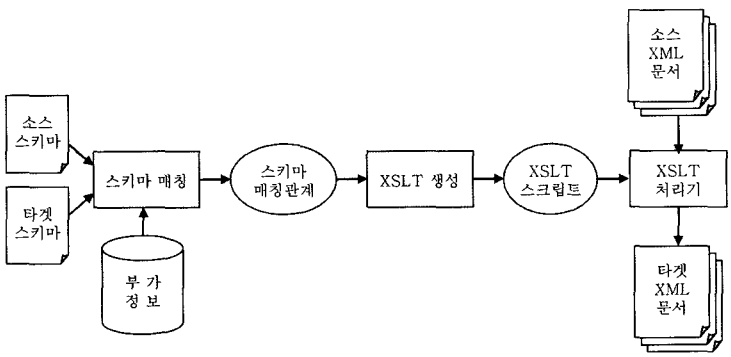

그림 1 XML 문서의 변환과정

2.2 문서모델

XML 스키마는 XML 문서에 포함될 엘리먼트(element)의 이름과 구조, 속성(attribute)의 이름 등을 정의한다. 특히 문서의 논리적 구성요소에 해당하는 엘리먼트의 이름과 계층구조는 물론이고, 각각의 엘리먼트가 포함할 수 있는 데이타 타입(data type) 등을 정의한다. 그림 2는 전자 상거래 분야에서 물품의 구매요청을 위해 사용중인 XML 스키마의 예이다.

예를 들어, 20번째 줄은 *ShipTo*라는 이름의 엘리먼트를 선언하고 자식 엘리먼트로서 *PostalCode, Country, City*, 그리고 *Street*(23~26번째 줄)를 포함하고 있음을 알 수 있다. 특히 *type="xs:string"*은 해당 엘리먼트의 데이타 타입이 문자열 형태임을 나타낸다.

본 논문에서는 XML 스키마를 표현하기 위하여 뿌리 노드(root node)를 포함하며 형제 노드간에 순서가 존재하는 순서 트리(ordered tree)에 기반한 문서 모델을 제안한다. 문서 모델은 XML 스키마에 정의된 엘리먼트와 속성을 노드로 갖는다. 각각의 노드는 레이블(label)과 값(value)를 갖는다. 레이블은 XML 스키마에 정의된 엘리먼트와 속성의 이름에 해당하며, 값은 XML 스키마에 정의된 엘리먼트나 속성의 데이타 타입으로서 단말 노드에만 존재한다. 그림 3은 그림 2의 스키마를 제안된 문서 모델로 표현한 결과이다. 한편 본 논문에서는 특정 노드의 부모 노드로부터 뿌리 노드까지의 노드들의 순차적인 집합을 해당 노드의 경로(path)라고 정의한다. 예를 들어, 그림 3에서 노드 UOM의 경로는 {*Item, Line, PO*}에 해당한다.

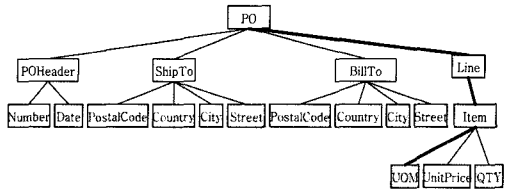

그림 3 그림 2의 XML 스키마를 제안된 문서 모델로 표현한 결과

2.3 관련연구

전술한 바와 같이 본 논문에서는 문서의 자동 변환을 위한 전처리 단계로서 온톨로지에 기반한 스키마 매칭 알고리즘을 제안한다. 스키마 매칭에 관한 연구는 표 1와 같이 다양한 방법으로 진행되어 왔다[7].

Bergamaschi 등[9]이 제안한 ARTEMIS는 관계형 데이타베이스 등에서 사용되는 스키마를 통합하기 위해 제안된 방법으로서 동의어 사전을 사용하며 구조적인 유사도를 이용하여 매칭을 계산하지만 XML 스키마와 같이 계층적 스키마에는 적합하지 않다. 한편 Lee 등[16]이 제안한 XClust는 스키마의 클러스트링을 위해서 스키마 매칭을 계산한다. XClust는 유사도를 이용하여 매칭관계를 찾고, 매칭되는 정도에 따라 두 스키마 사이의 전체 유사도를 계산한다.

```
1 : <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
2 :     <xs:element name="PO">
3 :         <xs:complexType>
4 :             <xs:sequence>
5 :               <xs:element ref="POHeader"/>
6 :               <xs:element ref="ShipTo"/>
7 :               <xs:element ref="BillTo"/>
8 :               <xs:element ref="Line"/>
9 :             </xs:sequence>
10:         </xs:complexType>
11:     </xs:element>
12:     <xs:element name="POHeader">
13:         <xs:complexType>
14:             <xs:sequence>
15:               <xs:element name="Number" type="xs:string"/>
16:               <xs:element name="Date" type="xs:string"/>
17:             </xs:sequence>
18:         </xs:complexType>
19:     </xs:element>
20:     <xs:element name="ShipTo">
21:         <xs:complexType>
22:             <xs:sequence>
23:               <xs:element name="PostalCode" type="xs:string"/>
24:               <xs:element name="Country" type="xs:string"/>
25:               <xs:element name="City" type="xs:string"/>
26:               <xs:element name="Street" type="xs:string"/>
27:             </xs:sequence>
28:         </xs:complexType>
29:     </xs:element>
30:     <xs:element name="BillTo">
31:         <xs:complexType>
32:             <xs:sequence>
33:               <xs:element name="PostalCode" type="xs:string"/>
34:               <xs:element name="Country" type="xs:string"/>
35:               <xs:element name="City" type="xs:string"/>
36:               <xs:element name="Street" type="xs:string"/>
37:             </xs:sequence>
38:         </xs:complexType>
39:     </xs:element>
40:     <xs:element name="Line">
41:         <xs:complexType>
42:             <xs:sequence>
43:               <xs:element ref="Item"/>
44:             </xs:sequence>
45:         </xs:complexType>
46:     </xs:element>
47:     <xs:element name="Item">
48:         <xs:complexType>
49:             <xs:sequence>
50:               <xs:element name="UOM" type="xs:string"/>
51:               <xs:element name="UnitPrice" type="xs:string"/>
52:               <xs:element name="QTY" type="xs:string"/>
53:             </xs:sequence>
54:         </xs:complexType>
55:     </xs:element>
56: </xs:schema>
```

그림 2 구매요청서 스키마의 예

표 1 스키마 매칭에 관한 연구

저자	연도	명칭	특 징	대상	매칭 종류
Li와 Clifton [8]	1994	SemInt	신경회로망을 이용한 학습기법을 적용하여 매칭을 계산	ER-DB	단순 매칭
Bergamaschi 등 [9]	1998	ARTE-MIS	동의어 사전과 데이타 타입을 이용하여 각 요소의 어휘 및 구조적 유사성을 계산 및 통합	OO-DB ER-DB	단순 매칭
Milo 등 [10]	1998	Trans-Scm	그래프에 이용하며 규칙기반의 알고리즘 제안	SGML OO-DB	단순 매칭
Lerner [11]	2000	Tess	스키마가 변화하더라도 그 변화를 인식하고 새로운 스키마와 비교한 후 스키마간에 갱신이 가능한 변환을 지원할 수 있는 방법 제안	ER-DB	단순 매칭
Doan 등 [12]	2001	LSD	기계 학습을 이용하며 샘플 스키마에 대한 학습을 통하여 새로운 문서에 대한 매칭관계를 계산하는 방법 제안	XML	단순 매칭
Miller 등 [13]	2001	Clio	이기종간의 스키마를 통합 및 관리하기 위하여 부가정보에 기반하여 매칭관계를 계산하고 사용자 상호작용을 지원하는 방법 제안	XML ER-DB	단순 매칭
Madhavan 등 [14]	2001	Cupid	요소 중심의 매칭과 구조 중심의 매칭을 적절히 혼합한 스키마 매칭 알고리즘 제안	XML ER-DB	단순 매칭
Su 등 [15]	2001	Xtra	XML 문서의 변환을 지원하기 위하여 스키마 매칭을 수행하며 이로부터 XSLT 스크립트를 생성하며 이를 위하여 변환연산과 비용모델을 제안	XML	단순 매칭
Lee 등 [16]	2002	XClust	트리로 표현된 스키마의 어휘 및 구조적 유사도를 계산하여 두 스키마 사이의 유사도 계산. 스키마 사이의 유사도를 이용하여 클러스터링하는 방법 제안	XML	단순 매칭
Do와 Rahm [17]	2002	COMA	여러 가지 매칭 알고리즘을 조합하여 적용할 수 있는 시스템 제안. 사용자의 피드백과 기존의 매칭결과를 이용할 수 있는 방법 제안	XML	단순 매칭
Melnik 등 [18]	2002	SF	유사도가 주변노드로 전파된다는 가정과 방향 그래프에 기반한 스키마 매칭 알고리즘 제안	ER-DB	단순 매칭
Xu와 David [19]	2003	·	매칭패턴을 포함하고 있는 온톨로지를 이용하여 복합매칭을 찾는 방법 제안	XML	복합 매칭
Dhamankar 등 [20]	2003	iMAP	기계 학습을 이용하여 복합매칭을 계산할 수 있는 방법 제안	ER-DB	복합 매칭

기존에 XML 문서의 자동변환을 위한 방법으로 Su 등[15]이 제안한 Xtra가 있다. Xtra는 소스 DTD를 타겟 DTD로 변환하는데 필요한 최소 비용의 변환 연산을 계산한다. Xtra는 스키마의 구조 정보를 고려하지 않으며 최소 비용의 변환 연산에 해당하는 매칭을 계산하기 때문에 정확한 매칭 결과를 계산하는데 있어서 한계를 갖는다. 정확한 매칭을 계산하기 위해서는 최소비용의 변환 연산보다 의미적 및 구조적으로 유사한 노드 사이의 매칭관계를 계산하여야 한다. 문서 변환을 위한 또 다른 연구로서 Milo 등[10]이 제안한 TransScm이 있다. TransScm은 정의한 변환규칙에 따라 그래프 모델로 표현된 스키마 사이의 매칭관계를 계산한다.

한편 스키마 매칭에 관한 일반적인 연구로서 Li와 Clifton[8]이 제안한 SemInt와 Doan 등[12]이 제안한 LSD와 같은 학습기법에 기반한 방법이 있다. SemInt는 관계형 스키마를 대상으로 해당 데이타를 여러 측면에서 학습시킨 후, 이를 이용하여 스키마 사이의 매칭관계를 찾는다. LSD 역시 많은 양의 XML 문서를 이용한 학습과정을 통해 스키마 사이의 매칭을 찾는다. 사용된 어휘, 기술된 형태, 빈도수 등 다양한 측면에 대해 학습시킨다. 따라서 본 방법은 학습을 위하여 많은 시간과 노력을 필요로 하며 학습을 위한 데이타 역시 사전에 준비되어야 한다. 특히 학습 데이타가 새로운 스키마를 포괄할 수 있을 만큼 충분해야 정확한 매칭을 찾을 수 있다.

Lerner[11]가 제안한 Tess는 기존의 스키마를 수정하여 새로운 스키마를 설계할 경우 기존의 스키마와 새로운 형태의 스키마 사이의 매칭관계를 계산한다. 이름을 분석하고 구조 정보를 이용하여 매칭관계를 찾는다. 또한 Miller 등[13]이 제안한 Clio는 여러 가지 부가정보를 활용하여 다대다 관계의 매칭을 찾는 시스템으로 사용자의 상호작용을 지원하고 스키마를 통합 및 관리하는데 사용된다. Madhavan 등[14]이 제안한 Cupid는 다양한 스키마를 대상으로 하며 복합적인 방법을 사용하여 매칭을 계산한다. XML 스키마와 데이타베이스 스키마에 적용 가능한 방법으로서 어휘 및 구조 정보를 이용하여 매칭관계를 계산한다. 특히 XML 스키마의 계층적인 특징을 반영하여 구조적인 유사도를 계산하고 있다.

한편 Do와 Rahm[17]이 제안한 COMA와 Melnik 등[18]이 제안한 SF가 있다. COMA는 여러 가지 방법들을 모듈화하여 적용할 수 있는 방법을 제안한다. 또한 사용자의 피드백과 기존 매칭결과의 재사용 등을 통해

정확성을 좀 더 향상시킬 수 있는 방법을 제안한다. 그러나 매칭결과가 다대다의 매칭결과를 생성하기 때문에 문서의 자동변환에 적용하는데 부적절하다. SF는 그래프로 나타낸 문서모델에서 한 노드의 유사도가 주위의 다른 노드의 유사도에도 영향을 준다는 아이디어를 이용한다. 즉, 유사도를 주위 노드로 전파시킴으로 전체적으로 스키마 사이의 구조적 유사도를 반영시킬 수 있는 방법을 제안한다.

최근에 복합매칭을 지원한 연구로서 Xu[19]는 계층적 구조의 온톨로지를 활용하여 복합매칭을 찾을 수 있는 방법을 제안한다. Xu가 제안한 온톨로지는 특정 도메인에 포함될 모든 개념(concept)들의 관계를 방향 그래프(directed graph)로 표현하며 각 온톨로지를 구성하는 개념에 대응하는 어휘리스트를 포함한다. 그러나 제안된 온톨로지는 전문가에 의해 수동으로 설계되어 어휘목록을 구축하는데 많은 비용이 소요된다. 또한 온톨로지가 처리하지 못한 스키마를 반영하기 위해서 온톨로지를 수동으로 갱신하여야 하는 단점을 갖는다. Dhamankar 등[20]이 제안한 iMAP은 학습기법을 활용하여 복합매칭을 찾는 방법으로서 미리 학습된 분류기에 해당 스키마에 따라 작성된 문서를 입력하여 스키마 사이의 매칭을 계산한다. 특히 iMAP은 소스와 타겟 문서에서 반복되는 데이타에 대한 정보를 이용하여 복잡한 연산에 해당하는 매칭을 계산할 수 있다. 그러나 iMAP은 관계형 데이타베이스를 대상으로 제안되어 XML 스키마의 계층적 특성을 고려하지 못한다. 또한 대부분의 학습과정이 XML 문서에 포함된 데이타 형태에 기반하기 때문에, 데이타의 형태는 비슷하지만 문맥정보에 따라 구분되어야 하는 매칭관계를 다루기 어려운 단점이 있다.

본 논문에서는 XML 문서의 변환을 위한 보다 정교한 수준의 스키마 매칭 알고리즘을 제안한다. 이를 위해서 단순매칭은 물론이고 복합매칭을 지원하는 방법을 제안하며 사용자의 피드백에 따라 동적으로 갱신 가능한 온톨로지에 기반한다.

3. 온톨로지

본 절에서는 제안된 온톨로지를 자세히 기술한다. 스키마 매칭과정에서 사용자는 잘못된 매칭을 제거하거나 찾지 못한 매칭을 추가하는 피드백을 제공할 수 있다. 제안된 온톨로지는 이러한 사용자 피드백을 이용하여 온톨로지를 갱신한다. 또한 제안된 온톨로지는 해당 도메인에서 사용하는 개념사이의 다양한 관계를 정의하기 때문에 복합매칭의 계산을 가능케 한다.

3.1 온톨로지 구조

제안된 온톨로지는 개념노드와 개념노드사이의 관계를 나타내는 간선으로 구성된 트리의 집합으로 표현된다. 본 논문에서는 개념을 노드로 갖는 트리를 개념 트리라고 정의한다. 개념노드는 XML 스키마에 기술된 요소나 속성의 이름을 레이블로 갖는다. 개념노드 사이의 관계는 IsA, PartOf, 그리고 Similar의 세 가지로 구분된다. 여기서 IsA, PartOf, 그리고 Similar는 각각 일반화, 부분합, 그리고 유사 관계를 나타낸다.

예를 들어, 그림 4는 주소록 분야에서 사용될 수 있는 온톨로지의 예이다. 그림에서 타원은 개념을 정의하며 간선은 관계를 나타낸다. 즉 개념 *MobilePHONE*은 개념 *PHONE*과 IsA 관계로 연결되고 *Cell Phone*은 *Mobile Phone*과 Similar 관계로 연결된다.

한편 개념노드 사이에 연관강도가 존재한다. 연관강도는 어휘 유사도 계산과정에서 어휘사이의 유사도로 활용되며 식 (1)과 같이 사용자 피드백의 빈도에 따라 지수적으로 증가 혹은 감소한다. 즉, 시스템이 정확한 매칭을 찾거나 시스템이 찾지 못한 매칭을 사용자가 추가한 경우, 두 어휘사이의 관계가 온톨로지에 정의되어 있다면 해당 온톨로지의 연관강도는 증가한다. 반대로 시스템이 찾은 결과를 사용자가 제거한 경우, 누적된 매칭 횟수를 감소시킴으로 그에 따른 연관강도가 감소한다.

$$Association(x) = \frac{1}{1+e^{-x}} \quad (1)$$

x : 사용자 피드백 입력 횟수

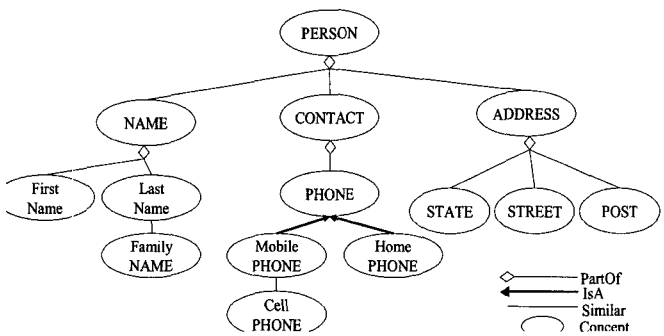

그림 4 온톨로지의 예

제안된 온톨로지는 스키마 매칭의 결과에 대한 사용자 피드백에 따라 개념트리를 추가하거나 병합하는 등의 연산을 적용하여 온톨로지를 갱신한다. 제안된 온톨로지 연산에 대한 자세한 설명은 다음 절에서 기술한다.

3.2 온톨로지 연산

제안된 온톨로지는 사용자의 피드백에 해당하는 온톨로지 연산에 의하여 갱신된다. 온톨로지 연산의 선택을 위한 사용자의 피드백은 다음과 같은 형태로 정의된다.

사용자 피드백 = (소스노드, 타겟노드, IsA | PartOf | Similar | Remove)

예를 들어, 시스템이 계산한 매칭관계를 사용자가 잘못된 것으로 판단하여 제거했다면 피드백 (소스노드의 이름, 타겟노드의 이름, Remove)이 입력된다. 반면에 시스템이 찾지 못한 매칭관계를 사용자가 새로이 추가했다면 해당 소스노드, 타겟노드, 그리고 두 노드 사이의 IsA, PartOf, 또는 Simlar 관계를 입력받는다. 사용자 피드백에 따라 제안된 온톨로지는 개념노드의 삽입, 개념트리의 삽입, 개념트리의 병합, 그리고 연관관계 삭제의 네 가지 연산을 수행한다. 각각의 연산에 대한 자세한 설명은 다음과 같다.

3.2.1 개념노드 삽입

사용자 피드백에 의해 입력된 노드 중 한 개가 온톨로지에 존재하면 온톨로지에 포함되어 있지 않은 노드에 해당하는 개념노드를 추가하며, 이 노드와 이미 포함되어 있는 개념노드와의 관계를 생성한다. 예를 들어, 사용자 피드백이 (*PHONE, MobilePHONE,* IsA)와 같다면 삽입 연산이 적용되어 그림 5와 같이 기존 온톨로지에 새로운 개념이 추가된다. 그림에서 회색 타원은 기존 온톨로지에 포함된 개념을 나타내고 공백 타원은 사용자 피드백에 의해 추가된 새로운 개념을 나타낸다.

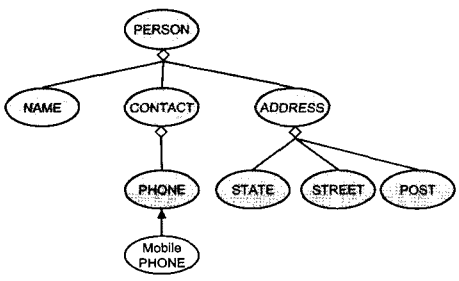

그림 5 사용자 피드백에 의해 삽입된 개념노드의 예

3.2.2 개념트리 삽입

사용자 피드백에 의해 입력된 두 노드가 온톨로지에 존재하지 않는 경우, 입력된 두 노드와 매칭관계를 이용하여 새로운 트리를 생성한다. 예를 들어, 그림 6과 같이 사용자 피드백(*Telephone, HomePhone,* IsA)이 입력된 경우, 개념 트리의 삽입 연산이 적용된다.

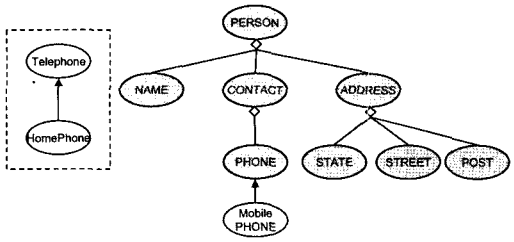

그림 6 사용자 피드백에 의한 새로운 개념트리의 삽입

3.2.3 연관관계 추가에 의한 개념트리 병합

사용자 피드백을 구성하는 두 개념이 각각 서로 다른 개념트리에 포함되어 있다면 병합연산이 적용된다. 그림 7은 온톨로지에 (*Telephone, PHONE,* Similar)가 입력된 경우, *Telephone*과 *PHONE*이 Similar 관계를 이루면서 하나의 개념트리로 병합된 모습을 나타낸 것이다.

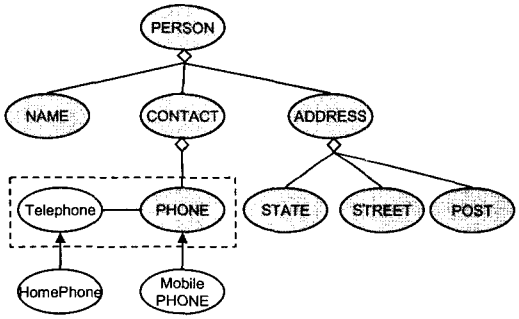

그림 7 사용자 피드백에 의한 개념트리의 병합

3.2.4 유사 개념노드의 그룹화

온톨로지 연산시 유사관계로 새로운 개념이 추가되거나 기존의 개념트리 간 병합이 발생할 경우, 유사관계의 개념간에는 그룹화 과정을 적용한다. 그룹화된 유사관계의 개념들은 동일한 IsA, PartOf 관계를 갖는다. 예를 들어, 그림 7과 같은 병합과정의 경우, *Telephone*과 *PHONE*은 유사관계로 병합되기 때문에 그룹화가 이루어진다. 그룹화 결과에 따라 *HomePhone*과 *PHONE*은 IsA관계를 갖으며 초기 연관강도는 *Telephone*과 *PHONE*의 연관강도와 *Telephone*과 *HomePhone* 연관강도의 평균값으로 설정된다. *Telephone*과 *Mobile-PHONE* 사이에도 마찬가지의 과정이 적용된다.

3.2.5 관계의 제거

사용자는 피드백을 통하여 잘못된 매칭을 제거할 경우, Remove 피드백이 입력된다. 이 경우, 먼저 매칭 오류의 원인이 노드 자체에 있는지 혹은 문맥적 분석 실

패에 의한 것인지를 판단하여 온톨로지 갱신 여부를 결정한다.

예를 들어, 노드의 이름은 모두 *NAME*이지만 한쪽은 *Employer*의 *NAME*이고 다른 쪽은 *Employee*의 *NAME*인 경우, 문맥정보가 다르기 때문에 사용자가 잘못된 매칭으로 판단한 것이다. 이와 같이 문맥정보에 따라 사용자가 매칭을 제거한 경우, 기존 온톨로지에 영향을 주지 않는다. 반면에 어휘 분석에 대한 실패로 인한 매칭 오류의 경우, 사용자 피드백이 온톨로지에 반영된다. 예를 들어, 매칭결과에 소스 노드 *NAME*과 타겟노드 *DeptNAME*간의 매칭이 포함되었다면 이것은 어휘 분석과정에서 *NAME*과 *DeptNAME*이 유사하다고 잘못 판단한 것이다. 이 경우 잘못된 노드 분석 때문에 생긴 오류이기 때문에 사용자 피드백이 온톨로지에 반영된다. 제거된 두 노드가 온톨로지에 포함되어 있다면 두 노드의 연관강도를 낮추고, 그렇지 않다면 가장 낮은 연관강도인 0으로 온톨로지에 추가한다. 전술한 것과 같이 연관강도는 매칭과정에서 유사도로 활용되기 때문에 0의 연관강도로 연결된 두 어휘는 매칭으로 선택될 수 없다. 즉, 기존 연구에 사용된 대부분의 온톨로지가 어휘 사이에 긍정적 관계만을 기술했던 것에 반해, 제안된 온톨로지는 매칭되지 않아야 하는 부정적 관계 또한 기술할 수 있다.

4. 스키마 매칭 알고리즘

제안된 스키마 매칭 알고리즘은 그림 8과 같이 크게 두 단계로 구성된다. 먼저 온톨로지와 어휘분석에 기반하여 단말노드 사이의 후보매칭을 계산하고, 계산된 후보매칭 사이의 경로 유사도를 비교하여 최종 매칭을 추출한다. 특히 단말노드 매칭과정에서 제안된 온톨로지를 이용하여 단순매칭은 물론이고 복합매칭을 계산한다.

4.1 단말노드간 후보매칭 계산

그림 9는 제품주문서에 사용되는 소스스키마 S1과 타겟스키마 S2 사이의 단말노드간 후보매칭을 계산한 모습을 도식한 것이다.

S1의 단말노드 *Name*은 S2의 단말노드 *Name*과 ①과 ②의 매칭으로 연결된다. 이와 같이 후보매칭을 찾는 과정에서는 단말노드만 고려하면 다대다의 매칭관계가 계산될 수 있다. 한편 매칭관계 ③의 경우, *Phone*과 *Telephone*은 동의어사전을 활용하여 대응관계를 갖는다. 또한 온톨로지에 *City*와 *Post*가 *Address*와 PartOf 관계로 정의되어 있을 경우, *City-Address, Post-Address* 관계를 계산할 수 있다. 이때 *City*와 *Post*는 동일한 서브트리에 포함되어 있고, *Address*와 모두

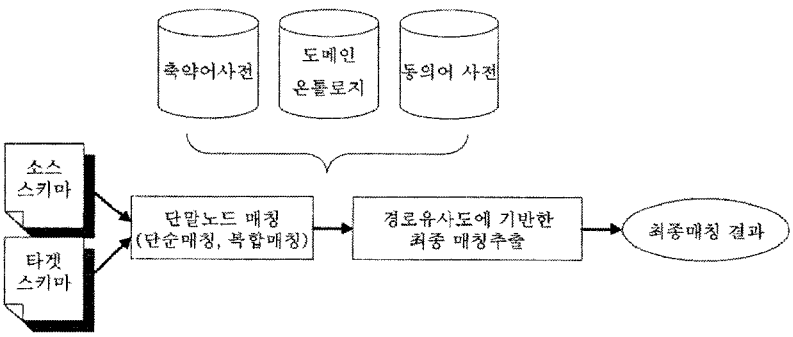

그림 8 제안된 스키마 매칭 과정

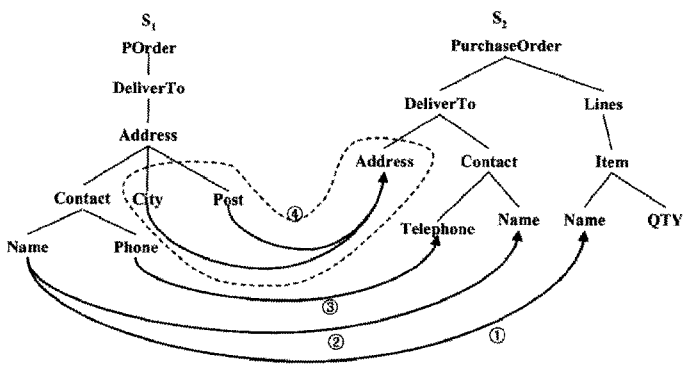

그림 9 단말노드간 후보매칭 계산의 예

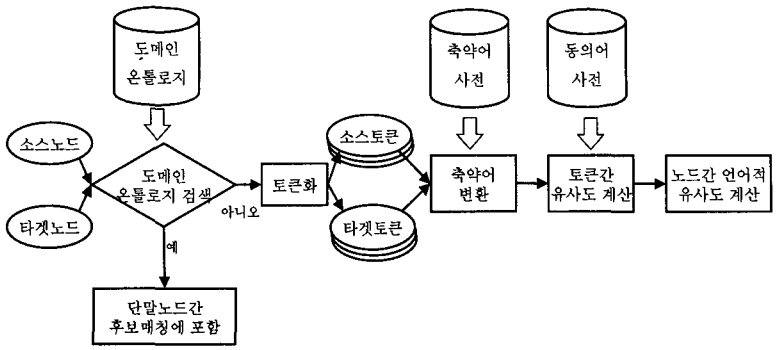

그림 10 단말노드간의 유사도 계산과정

PartOf 관계로 매칭되기 때문에 2:1로 그룹화되어 하나의 복합매칭 ④를 이루게 된다. 단말노드 사이에 유사도를 계산하여 후보매칭으로 선택하는 과정은 아래 단계별로 상세히 기술한다.

두 스키마 트리를 구성하는 모든 단말노드를 비교하면서 단말노드 사이에 후보매칭을 찾는다. 후보매칭의 계산과정은 그림 10과 같이 두 단말노드의 관계가 온톨로지에 포함된 경우와 포함되지 않은 경우로 나뉜다.

4.1.1 온톨로지를 이용한 후보매칭의 계산

대상이 되는 두 단말노드가 온톨로지에 포함되어 있다면 해당 연관강도를 단말노드 임계값과 비교하여 매칭 여부를 결정한다. 먼저 온톨로지에 포함된 두 단말노드의 연관강도가 단말노드 임계값 이상인 경우 후보매칭에 포함시킨다. 이때 두 단말노드의 관계가 Similar 관계이면 단순매칭으로 후보매칭에 포함시키고, IsA나 PartOf 관계인 경우 복합매칭으로 후보매칭에 포함시킨다. 선택된 후보매칭 중 동일한 서브트리에 포함된 두 개 이상의 단말노드가 한 노드와 동일한 관계로 복합매칭을 형성하면, 하나의 복합매칭으로 그룹화한다. 특히 그룹화된 복합매칭은 하나의 후보매칭으로 다뤄진다.

4.1.2 단말노드 유사도를 이용한 후보매칭의 계산

입력된 단말노드가 온톨로지에 포함되어있지 않다면, 두 노드간의 단말노드 유사도를 계산하여 단말노드 임계값 이상의 관계를 갖을 경우 후보매칭에 포함시킨다. 단말노드 유사도는 식 (2)와 같이 노드 레이블 사이의 어휘 유사도와 데이타 타입 유사도(data type similarity) 의 합으로 정의한다. 여기서 가중치 w_l과 w_t는 각각 어휘 유사도와 데이타 타입 유사도의 가중치를 의미한다.

$$\text{단말노드유사도}(N_s,N_t) = w_l * \text{어휘유사도}(N_s,N_t) + w_t * \text{데이터타입유사도}(N_s,N_t) \quad (2)$$

어휘 유사도는 두 노드의 레이블 사이의 유사도를 나타내는 값으로서 해당 레이블을 대문자나 특수기호를 기준으로 토큰화한 후, 각 토큰사이의 유사도를 계산한다. 토큰 사이의 유사도는 문자열 비교와 동의어 사전[21]을 이용하여 계산한다. 최종적으로 두 노드의 어휘 유사도는 소스와 타겟노드를 구성하는 전체 토큰 중에서 매칭되는 토큰의 비율로 계산한다.

두 노드 사이의 유사도를 계산하기 위해선 어휘 유사도 뿐만 아니라 데이타 타입 유사도를 고려한다. 트리를 구성하는 단말노드는 다양한 종류의 데이타 타입을 값으로 갖는다. 특히 데이타 타입이 서로 다른 노드간의 변환은 정보 손실을 가져올 수 있다. 데이타 타입 유사도는 이와 같이 서로 다른 데이타 타입을 갖는 노드간의 변환에 의하여 발생 가능한 정보 손실을 표현하기 위해서 제안되었다. 본 논문에서는 노드의 데이타 타입간의 유사한 정도를 해당 노드가 포함하는 정보의 손실정도에 따라 '동일', '비손실 변환가능', '손실 변환가능', 그리고 '변환불가'의 네 종류로 구분한다. 동일한 데이타 타입의 경우, 가장 큰 유사도를 부여하며 정보의 손실정도에 따라 점점 낮은 유사도를 부여한다. XML 스키마에서 지원하는 44개의 기본적인 데이타 타입 중에서 7개의 주요 데이타 타입간의 유사도는 표 2와 같다.

4.2 경로 유사도를 이용한 최종매칭 추출

이전 단계에서 추출한 후보매칭은 다대다 관계의 매칭을 포함할 수 있다. 제안된 방법은 대응관계를 갖는 단말노드 사이의 경로 유사도를 비교하여 일대일 매칭을 추출한다. 전술한 바와 같이 복합매칭 역시 일대일 매칭으로 간주한다.

$$\text{경로유사도}(Ps,Pt) = \frac{\sum S_i}{|Ps|} + \frac{\sum S_i}{|Pt|} \quad (3)$$

Ps: 소스 노드의 경로, Pt: 타겟 노드의 경로
S_i: Ps와 Pt 사이에 대응관계를 갖는 중간노드의 유사도

서로 대응하는 두 경로 Ps와 Pt 사이의 경로 유사도는 식 (3)과 같이 계산할 수 있다. 먼저 서로 대응되는 두 경로에 포함되어 있는 중간노드를 비교하여 매칭되는 중간노드를 찾는다. 경로 유사도는 소스와 타겟의 각각의 경로에 포함된 매칭되는 중간노드 유사도에 기반

표 2 데이타 타입간의 유사도

소스의 데이타 타입 \ 타겟의 데이타 타입	string	decimal	double	float	boolean	duration	dateTime
string	3.0	1.0	1.0	1.0	0.0	1.0	1.0
decimal	2.0	3.0	2.0	2.0	1.0	0.0	0.0
double	2.0	1.0	3.0	1.0	0.0	0.0	0.0
float	2.0	1.0	2.0	3.0	0.0	0.0	0.0
boolean	2.0	2.0	0.0	0.0	3.0	0.0	0.0
duration	2.0	0.0	0.0	0.0	0.0	3.0	1.0
dateTime	2.0	0.0	0.0	0.0	0.0	1.0	3.0

한다. 특히 각 경로에 대하여 매칭된 중간노드의 유사도의 합을 경로에 포함된 노드 수로 나눈 후, 이를 더하여 경로 유사도를 계산하면, 깊이(depth)가 다른 두 스키마 사이에 대해서도 유사도를 적절히 계산할 수 있다.

경로 유사도를 구하기 위해선 먼저 경로사이에 매칭되는 중간노드를 찾아야한다. 중간노드 역시 단말노드와 마찬가지로 두 노드사이의 유사도가 임계값보다 클 경우 매칭되며, 두 노드 Ns와 Nt 사이의 중간노드 유사도는 식 (4)와 같이 계산할 수 있다.

$$\text{중간노드유사도}(N_s,N_t) = w_l * \text{어휘유사도}(N_s,N_t) + w_s * \text{구조적유사도}(N_s,N_t) \quad (4)$$

어휘 유사도는 단말노드 매칭과정에서 적용된 방법을 이용하여 계산하고 구조적 유사도는 식 (5)와 같이 해당 서브트리에 포함된 서로 대응관계를 갖는 단말노드의 비율로 계산한다.

$$\text{구조적유사도}(Ns,Nt) = \frac{Ns\text{과 } Nt \text{ 사이에 매칭된 노드의 수}}{|LNs|+|LNt|} \quad (5)$$

LNs : Ns트리(Ns를 뿌리노드로 갖는 서브트리)의 대응관계를 갖는 단말노드의 집합

LNt : Nt트리(Nt를 뿌리노드로 갖는 서브트리)의 대응관계를 갖는 단말노드의 집합

경로 유사도를 비교하여 최종매칭을 추출하기 앞서 계산된 경로 유사도가 임계값 이하인 매칭은 제거한다. 이 과정은 어휘는 비슷하지만 실제 다른 용도로 사용되고 있는 단말노드를 제거해주는 과정으로 어휘가 완전히 동일하더라도 사용된 문맥이 다르면 매칭에서 제외된다.

계산된 경로 유사도를 이용하여 단말노드간의 후보매칭 중 최종 매칭을 선택한다. 먼저 소스트리의 임의의 단말노드에 대해 계산된 단말노드 후보매칭 중, 가장 높은 경로 유사도를 갖는 매칭을 선택한다. 즉, 임의의 소스노드가 여러 타겟노드와 대응관계를 갖는 경우, 경로 유사도가 가장 높은 타겟노드를 선택한다. 모든 소스 단말노드에 대하여 이 과정을 반복 적용한다. 한 소스 단말노드와 대응되는 후보매칭 중 경로 유사도를 비교하여 최종매칭을 선택한 후, 타겟 단말노드에 대해서도 동일한 과정을 적용한다. 즉, 한 타겟 단말노드와 매칭된 여러 후보매칭 중 가장 높은 경로 유사도를 갖는 매칭 관계를 선택한다.

한편 경로 유사도를 비교하여 가장 유사한 매칭을 찾는 과정에서 동일한 값의 경로 유사도를 갖는 매칭이 다수 존재할 수 있다. 이러한 경우, 가장 적절한 매칭의 선택을 위해서 단말노드 유사도가 높은 것을 선택한다. 특히, 복합매칭인 경우 온톨로지에서 정의된 연관강도의 평균을 단말노드 유사도로 가정하여 다른 매칭들과 비교한다. 만약 동일한 값을 갖는 단말노드 유사도 역시 두개 이상 존재할 경우, 해당 후보매칭들이 형제(sibling) 관계이면 하나의 복합매칭으로 간주하여 최종매칭으로 선택한다. 예를 들어, 단말노드 매칭결과, 소스 스키마의 *Person-Address-street1*과 *Person-Address-street2*의 두 노드가 타겟스키마의 *Person-AddressInfo-street*과 2:1의 관계로 후보매칭에 포함되었다고 하자. 두 매칭 모두 경로 유사도와 단말노드 유사도가 동일하고, 매칭을 이루는 *street1*과 *street2*는 형제관계이기 때문에 (*street1*, *street2*) - (*street*)의 복합매칭을 최종매칭으로 선택한다.

그림 11에서 매칭 ①부터 ⑥은 모두 단말노드 매칭결과 계산된 후보매칭이다. 그러나 매칭 ①은 낮은 경로 유사도(=0.28)을 갖으며 경로 유사도 임계값(=0.3) 보다 작기 때문에 최종매칭 추출이전에 제거된다. 최종매칭 추출을 위해 다대일 매칭관계를 검색한 결과, 매칭 ②와 ③ 그리고 매칭 ④와 ⑤가 모두 1:2의 관계를 형성한다. 따라서 경로 유사도 비교를 통하여 최종 매칭으로서 각각 ③과 ⑤를 선택한다. 복합매칭 ⑥의 경우도 위와 같이 단일의 매칭으로 취급하여 경로 유사도를 계산한다. 그러나 예제의 경우 중복되는 매칭이 없이 단일의 매칭만 존재하기 때문에 최종 매칭결과에 포함된다. 따라서 위 예제의 경우, 전체 단말노드 매칭 중 실선으로 그려진 ③, ⑤, 그리고 ⑥이 최종매칭으로 추출된다.

한편, 사용자는 계산된 스키마 매칭결과에 대해 잘못된 매칭을 제거하거나 누락된 매칭을 입력하는 후처리를 수행할 수 있다. 3장에서 기술한 바와 같이 사용자의

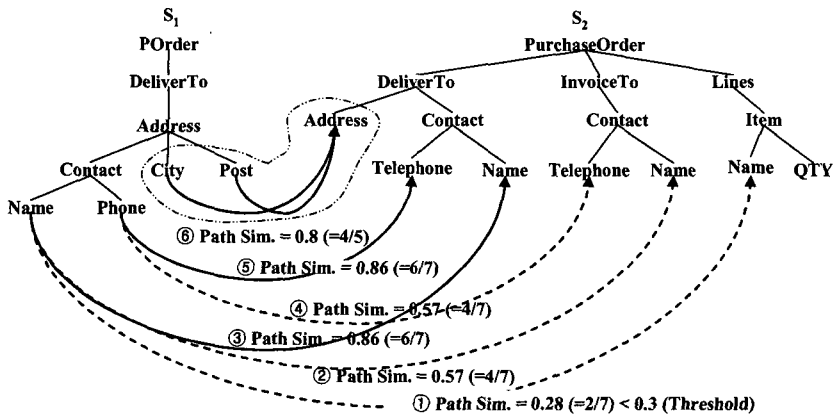

그림 11 최종매칭 추출의 예

후처리에 대한 적절한 온톨로지 연산이 적용되어 온톨로지는 자동으로 갱신된다.

5. 실험결과

본 절에서는 제안된 방법의 성능을 평가하기 위해서 4개의 도메인에서 사용 중인 21개의 XML 스키마를 대상으로 실험하였다. 기존 연구에서 사용된 3개 도메인에 속한 스키마들과 이보다 크고 복잡한 스키마를 대상으로 실험한 결과, 평균 97%의 정확률과 83%의 재현율을 보임으로 기존 연구보다 우수한 성능을 보였다. 특히 온톨로지 갱신을 통하여 단말노드 매칭과정에서 찾지 못했던 매칭을 추출함으로서 재현율이 약 7% 향상됨을 확인할 수 있었다. 실험데이타 및 실험결과에 대한 기술은 다음과 같다.

5.1 실험데이타

실험을 위해 사용된 실험데이타는 각각 부동산 정보를 위한 House List(실험데이타1), 대학에서 사용되는 Course List(실험데이타2), 제품 구매를 위한 Purchase Order(실험데이타3), 그리고 제품 송장에 대한 Invoice (실험데이타4) 용 스키마로서 표 3과 같다.

기존 연구와의 비교를 위해 3개의 도메인에 사용된 스키마는 기존의 스키마 매칭 연구에서 사용된 실험데이타를 그대로 사용하였다. 실험데이타1~3은 모두 기존 연구의 관련 홈페이지에서 제공받았으며, 실험데이타4는 전자상거래에서 실제 사용되는 데이타를 수집하였다.

House List와 Course List 스키마는 Doan 등[12]의 연구에서 사용된 실험데이타로서 실제 사용 중인 5개의 스키마와 실험을 위해 임의로 구성된 mediated 스키마로 구성된다. Purchase Order 스키마는 Do와 Rahm [17]의 연구에서 사용된 실험데이타이다. 그러나 기존의 스키마 매칭 연구에서 사용된 실험데이타는 크기가 작고 간단한 것들이 많기 때문에 실제 전자상거래에서 사용되며 크고 복잡한 구조의 Invoice 스키마를 추가하여 실험하였다.

5.2 성능분석

모든 실험 도메인에 대해 실험은 크게 두 가지 방법으로 수행되었다. 첫 번째는 도메인 온톨로지의 갱신과정 없이 매칭을 수행하였고, 두 번째는 이전 스키마 매칭결과에 대한 사용자 피드백을 입력받아, 이를 기반으로 온톨로지를 갱신하였다. 즉, 처음에 수행된 실험은 기본적인 부가정보만을 이용하지만 마지막 스키마 쌍의 경우 이전의 사용자 피드백이 모두 적용된 온톨로지를 기반으로 매칭을 계산한다.

실험방법은 기존연구와의 공정한 성능비교를 위해 각 실험데이타가 사용되었던 기존연구와 동일한 방법으로 수행하였다. 실험데이타1과 실험데이타2에 대해서는 5개의 소스스키마와 하나의 타겟스키마를 구성하여 스키마 매칭을 수행 후 매칭결과를 측정하였고, 실험데이타 3에 대해선 5개의 스키마에 대해 10개의 조합으로 각각 스키마 매칭을 수행한 후 결과를 측정하였다. 실험데이타

표 3 실험데이타

실험데이타	도메인	스키마수	평균 단말노드수	사용된 연구
실험데이타1	House List	6	21	Doan[12]
실험데이타2	Course List	6	13	Doan[12]
실험데이타3	Purchase Order	5	62	Do[17]
실험데이타4	Invoice	4	264	-

※ 실험데이타 : http://icl.yonsei.ac.kr/schemamatching/testdata

4도 실험데이타 3과 마찬가지로 4개의 스키마의 6개의 조합으로 매칭을 수행하여 결과를 측정하였다.

그림 12와 그림 13은 각각 실험데이타1과 실험데이타 2에 대한 실험결과이다. 여기서 정확률은 시스템이 찾은 매칭결과 중 정확하게 찾은 매칭의 비율을 나타내고, 재현율은 전체 찾아야 할 매칭 결과 중 시스템이 정확하게 찾은 매칭의 비율로 계산된다. Overall은 매칭 후 사용자에게 요구되는 후처리 비용을 수치화한 값으로 Melnik 등[18]의 연구에서 소개되었다. Overall은 식 (6)과 같이 계산하며 값이 작을수록 후처리 비용을 더 많이 요구함을 의미한다.

$$Overall = Recall^{*}\left(2 - \frac{1}{\mathrm{Precision}}\right) \quad (6)$$

실험결과, 제안한 방법으로 실험한 결과 온톨로지 갱신을 하지 않았을 경우 실험데이타1의 경우, 평균 96%의 정확률과 70%의 재현율을 보였다. 반면에 온톨로지를 사용하였을 경우, 100%의 정확률과 78%의 재현율을 나타내었다. 실험데이타2의 경우 온톨로지 갱신을 하지 않은 경우, 94%의 정확률과 67%의 재현율을 보였지만, 온톨로지를 갱신하여 실험한 경우, 평균 95%의 정확률과 84%의 재현율을 보였다.

그림 14는 실험데이타3에 대한 실험 결과이다. 제안된 방법으로 실험한 결과 온톨로지 갱신을 사용안한 경우 평균 96%의 정확률과 88%의 재현율을 보였으며 온톨로지 갱신을 사용한 경우 96%의 정확률과 93%의 재현율을 나타냄으로서 기존 연구 결과보다 좋은 성능을 보임을 확인할 수 있었다.

그림 15는 실험데이타4에서 사용 중인 크고 복잡한 스키마에 대한 실험 결과이다. 실험 결과, 평균 95%의 정확률과 78%의 재현율을 나타내어 복잡한 스키마에 대해서도 성능이 떨어지지 않음을 확인할 수 있었다. 한편 Invoice 스키마에 대한 실험에서는 온톨로지 갱신이 성능 향상에 기여하지 못했다. 왜냐하면 스키마 수가 4개로 적었고, 스키마를 구성하는 노드 레이블이 여러 개의 토큰으로 구성되는 등 서로 상이하여 다른 스키마에서 다시 사용될 확률이 적었기 때문이다. 특히 다른 스키마 쌍에 비해 1-4 스키마 쌍의 재현율이 낮은 이유는 1번 스키마와 4번 스키마의 중간노드의 레이블 형태가 매우 상이하여 경로 매칭이 이루어지지 않았기 때문이다.

한편 실험에 사용된 임계값은 사전에 실험을 통해 사전에 결정하였다. 단말노드 임계값은 모두 공통적으로 0.65로 설정하였고, 경로 유사도 임계값은 0.47~0.49로

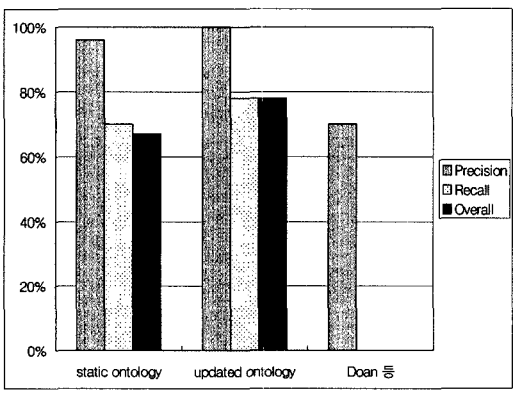

그림 12 실험데이타1에 대한 실험 결과

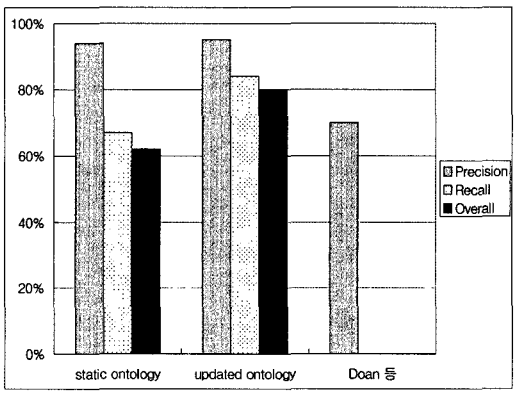

그림 13 실험데이타2에 대한 실험 결과

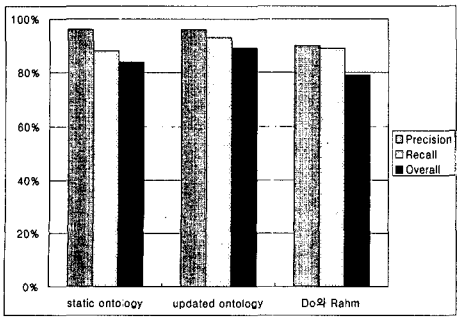

그림 14 실험데이타3에 대한 실험 결과

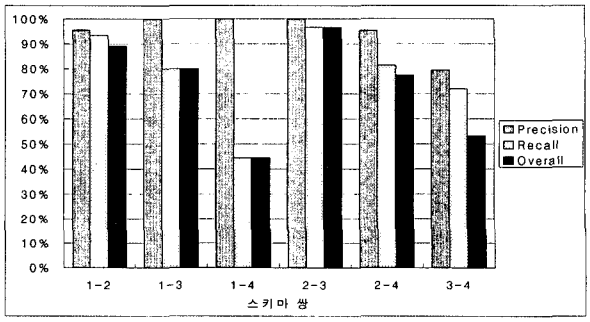

그림 15 실험데이타4에 대한 실험 결과

각 도메인에 따라 설정되었다. 어휘 유사도와 데이타 타입 유사도 사이의 가중치는 각각 0.8과 0.2로 설정하였다. 대부분의 스키마가 문자열 등의 포괄적인 데이타 타입을 많이 사용하였기 때문에 어휘 유사도에 보다 큰 가중치를 부여하였다. 이와 같은 이유로 중간노드 유사도 계산시, 사용되는 어휘 유사도와 구조적 유사도의 가중치도 각각 0.8과 0.2로 설정하여 실험하였다.

제안된 방법은 어휘 정보를 이용하여 가능성 있는 후보매칭 집합을 계산하고, 문맥정보인 경로 유사도를 비교하여 최종매칭을 선택하기 때문에 계층적 구조로 설계된 XML 스키마에 효과적임을 확인할 수 있었다. 그러나 정확률에 비해 재현율이 떨어지는 것을 볼 수 있다. 그림 14에서 온토롤지 갱신을 사용하지 않을 경우, 재현율은 Do와 Rahm의 방법이 우수하였다. 이것은 어휘 정보를 이용한 후보매칭 과정에서 매칭을 적절히 추출하지 못했기 때문이다. 제안된 단말노드 유사도 계산 방법은 토큰화를 통한 문자열 비교에 의존하고 있다. 한편 일반적으로 스키마는 동의어 사전이나 축약어 사전으로 해결할 수 없는 다양한 형태로 축소 및 결합된 어휘를 사용한다. 특히 유사관계가 아닌 IsA나 PartOf 관계로 매칭되는 복합매칭의 경우, 어휘 유사도를 이용하여 후보매칭을 계산하기는 더욱 어렵다. 이러한 한계를 극복하고자 사용자 피드백에 의해 점증적으로 갱신되는 온톨로지를 제안하여 실험한 결과, 언어적 유사도만으로 계산하지 못했던 매칭을 찾을 수 있음을 확인하였다. 또한 복합매칭을 계산함으로서 전체적으로 재현율을 향상시킴을 확인시킬 수 있었다.

5.3 기존 연구와의 비교

동일한 실험데이타를 사용한 기존 연구와의 비교에서 볼 수 있듯이 제안된 방법이 정확률과 재현율에서 우수함을 확인할 수 있었다. 특히 온톨로지 갱신을 적용하지 않더라도 평균 95%의 정확률을 보여 기존의 스키마 매칭에 관한 연구 결과인 [22]와 비교할 때 우수한 성능을 나타냈다. 따라서 본 논문에서 제안하고 있는 스키마 매칭 알고리즘은 정확한 스키마 매칭이 요구되는 문서변환에 더욱 효율적임을 알 수 있다.

기존의 스키마 매칭 연구 역시 다양한 종류의 유사도에 기반한다. Madhavan 등[14]은 어휘 유사도와 구조적 유사도 등을 계산하여 매칭관계를 추출하고, Melnik 등[18]은 스키마 트리의 구조적 유사도를 계산할 수 있는 방법을 제안한다. 그러나 기존 연구의 대부분은 다양한 유사도를 더하는 방법에 중점을 두고 있다. 이러한 방법은 스키마를 설계하는 스타일에 따라 일관된 성능을 보장할 수 없다. 예를 들어, 크기가 크고 복잡한 스키마의 경우, 문맥정보를 고려할 때 서로 대응관계를 갖지 않으나 단말노드간의 유사도는 매우 높아 매칭되는 경우가 존재한다. 반대로 서로 다른 단말노드임에도 불구하고 문맥정보가 매우 유사하여 전체적으로 유사도가 높게 계산되는 경우가 발생할 수 있다.

제안된 방법은 실험 결과에서도 확인할 수 있듯이, 매우 크고 복잡한 스키마에 대해서도 높은 수준의 정확성을 보인다. 이것은 스키마에 동일한 서브트리가 여러번 반복되더라도 제안된 경로 유사도를 이용한 문맥정보의 비교를 통해 가능성 있는 후보매칭 중 가장 적절한 매칭을 선택하기 때문이다. 그러나 전 절에서도 기술한 바와 같이, 후보매칭을 계산하는 단계에서 임의의 매칭을 계산하지 못하면 이후 단계에서 전혀 고려되지 않는다는 단점을 갖는다. 따라서 본 논문에서는 점증적으로 갱신가능한 도메인 온톨로지를 사용하였다.

기존 연구의 대부분은 도메인 사전 혹은 도메인 온톨로지를 사용하여 매칭관계를 계산한다[7]. 특히 Xu 등[19]의 연구는 제안된 방법과 유사하게 계층적 구조를 갖는 도메인 온톨로지를 활용하여 복합매칭을 지원한다. Xu 등은 어휘 목록과 데이타 값의 패턴을 포함하고 있는 그래프 형태의 온톨로지를 사용한다. 예를 들어, 주소를 나타내는 온톨로지가 있다면 주(state)에 해당하는 개념은 부가정보로 모든 주의 이름을 포함하고 있으며, 이름을 나타내는 온톨로지가 있으면 성을 나타내는 개념은 현재 사용되고 있는 성을 포함한다. 특히 온톨로지에 기술된 정보를 바탕으로 XML 문서의 내용을 이용하여 온톨로지와 문서 사이의 매칭관계를 계산한 후, 최종적으로 두 XML 스키마 사이의 매칭을 계산한다. Xu 등의 방법은 해당 도메인에 대한 많은 정보를 포함하고 있어, Xu 등의 논문에서 사용한 실험데이타에 대해 91%의 정확률과 93%의 재현율을 보여 비교적 좋은 성능을 나타냈다. 그러나 Xu 등의 매칭 방법은 온톨로지가 매우 완벽하게 구축된 도메인에 대해서만 적용이 가능하다.

한편, 기존에 Doan 등[12]과 Dhamakar 등[20]과 같이 기계학습에 기반한 연구가 발표되었다. 다수의 XML 문서를 이용해서 학습한 분류기를 이용하여 매칭관계를 선택한다. 주로 데이타 타입, 문자열의 길이, 특정 어휘의 빈도 수 등의 다양한 특징을 학습하며 학습결과를 종합적으로 적용한다. 그러나 기계학습을 이용한 매칭 알고리즘은 계층적 구조의 XML 스키마에 직접 적용하는데 제약이 있다. 왜냐하면 계층적 구조의 XML 스키마는 이름은 같지만 문맥정보가 상이한 요소를 다수 포함할 수 있다. 예를 들면, 본 논문에서 사용한 Invoice 스키마는 *Invoice-ShipTo-ContactInfo-phone, Invoice-BillTo-ContactInfo-phone* 등과 같이 전화번호 형태의 데이타 값을 갖는 단말노드를 5~6개 포함한다. 이는 문맥적 분석이 아니면 구분할 수 없는 경우에 해당한다.

특히 학습에 기반한 스키마 매칭의 대부분은 XML 문서를 학습 대상으로 하기 때문에 문맥정보에 대한 학습이 매우 어렵다. 따라서 동일한 형태의 데이타 값이 여러 개 포함된 크고 복잡한 스키마에 대해선 좋은 성능을 기대하기 힘들다.

6. 결론 및 향후연구

본 논문에서는 XML 문서 변환을 위해 두 단계로 구성된 스키마 매칭 알고리즘을 제안하였다. 먼저 온톨로지, 어휘정보, 그리고 데이타 타입 정보를 이용하여 단말노드 사이의 후보매칭을 계산하고, 문맥정보를 나타내는 경로 유사도 비교를 통해 최종매칭을 추출한다. 또한 보다 정교한 수준의 매칭을 계산하기 위해서 이전 매칭 결과에 대한 사용자 피드백을 이용하여 점증적으로 갱신되는 온톨로지를 제안하였다. 특히 제안된 온톨로지는 IsA나 PartOf와 같은 다양한 관계를 기술함으로 단순 매칭은 물론이고 복합매칭을 지원한다. 제안된 알고리즘의 성능을 평가하기 위해서 네 가지 도메인에서 사용중인 스키마를 대상으로 실험한 결과, 평균 97%의 정확률과 83%의 재현율을 보여, 동일한 실험데이타를 사용한 기존 연구보다 우수하였다. 특히 제안된 온톨로지의 적용을 통하여 후보매칭 추출 단계에서 계산하지 못했던 매칭을 계산함으로서 재현율을 향상시킴을 확인할 수 있었다.

점점 더 많은 정보가 XML 형태로 생산되고 있다. 전자상거래는 물론, 웹에서의 정보 표현이나 정보의 수집이나 교환에도 XML이 표준으로 사용되고 있다. 심지어, 디지털 기기들의 프로파일이나 제공되는 서비스를 표현하기 위해서도 XML 문서를 사용하고 있다. 하지만 동일한 분야의 XML 문서간의 상호운용은 XML의 유연성으로 인해 어려운 것이 현실이다. 따라서 자동화된 XML 스키마 사이의 매칭 관계 계산은 더욱더 필요할 것이다.

현재까지 제안된 스키마 매칭에 관한 연구는 아직 높은 성능을 갖지 못하며, 대부분의 연구에서 수행한 실험은 작고 간단한 스키마들을 대상으로 하였으며 실제로 사용 중인 크고 복잡한 스키마에 대한 성능평가를 수행하지 않았다. 이에 본 논문에서는 기존 연구에서 사용된 실험데이타 외에 실제 전자상거래에서 사용 중인 크고 복잡한 스키마에 대한 실험을 통하여 제안된 방법의 우수성을 확인할 수 있었다. 한편 엘리먼트나 속성의 이름을 독특한 형태로 설계한 스키마의 경우, 단말노드 매칭 과정에서 매칭을 찾지 못하는 경우가 발생하였다. 결과적으로 해당 실험데이타에 대해선 낮은 재현률을 보였다. 그러나 사용자 피드백 분석을 통한 점증적인 온톨로지의 갱신을 통하여 이후 매칭에서는 비슷한 형태의 레이블에 대해서 보다 향상된 결과를 얻을 수 있을 것이라 기대된다.

한편 현재 동일한 분야에서 조차 엘리먼트 및 속성의 이름을 정의하는데 다양한 방식을 사용한다. 따라서 향후에는 보다 정교한 수준의 매칭을 지원하기 위해서 온톨로지에 포함되는 개념의 어휘 분석 결과와, 다양한 어형이나 합성어 정보에 대한 피드백을 스키마 매칭에서 활용할 수 있는 방법을 연구할 것이다. 또한 스키마마다 노드를 나누어 정의하는 기준이 매우 다양할 수 있다. 예를 들어, 임의의 스키마는 엘리먼트 이름을 *ShipTo-PartInfo*와 같이 기술하는 반면에, 다른 스키마는 *ShipTo-Partner-Information*와 같이 각각의 엘리먼트로 나누어 기술할 수 있다. 이 같은 경우, 제안된 방법으로는 매칭을 적절히 계산할 수 없다. 이에 노드 단위가 아닌 전체 경로를 하나의 문자열로 간주하는 유사도 계산 방식을 추가로 적용할 수 있을 것이다. 향후 더욱 정교한 경로 유사도 계산 방법을 연구할 것이다.

향후 연구로는, 지금까지의 연구 결과를 바탕으로 특화된 도메인에서 사용될 수 있는 XML 스키마 매칭알고리즘과 매칭을 통해 실제 XML 데이타를 변환해줌으로 상호운용을 보장해주는 프레임워크에 대해 연구할 예정이다. 예를 들어, 다른 XML 스키마를 사용하여 정보를 생성하는 모바일 폰의 전화번호부를 다른 기종의 모바일 폰이라 하더라도 정보를 공유할 수 있는 시스템 등을 위한 연구가 될 것이다.

참 고 문 헌

[1] World Wide Web Consortium, Extensible Markup Language(XML) 1.0(Second Edition), W3C Recommendation, http://www.w3c.org/TR/REC-xml, 2000.

[2] World Wide Web Consortium, XML Schema 1.0, W3C Recommendation, http://www.w3.org/TR/xmlschema-0/, 2001.

[3] World Wide Web Consortium, XSL Transformations (XSLT) 1.0, W3C Recommendation, http://www.w3.org/TR/1999/REC-xslt-19991116, 1999.

[4] Eila Kuikka, Paula Leinonen, and Martti Penttonen, "Toward Automating of Document Structure Transformations," Proc. ACM Symposium Document Engineering, pp. 103-110, 2002.

[5] MicroSoft biztalk mapper, http://www.microsoft.com/biztalk/.

[6] XSL Wiz, http://www.induslogic.com/.

[7] Erhard Rahm and Philip A. Bernstein, "A Survey of Approaches to Automatic Schema Matching," Very Large Data Bases Journal, Vol. 10, No. 4, pp. 334-350, 2001.

[8] Wen-Syan Li and Chris Clifton, "Semantic Integration in Heterogeneous Databases Using Neural Networks," Proc. Int'l Conf. Very Large DataBase,

pp. 1-12, 1994.
[9] Sonia Bergamaschi, Silvana Castano, Sabrina De Capitani di Vimercati, S. Montanari, and Maurizio Vincini, "An Intelligent Approach to Information Integration," Proc. Int'l Conf. Formal Ontology in Information Systems, pp. 253-267, 1998.
[10] Tova Milo and Sagit Zohar, "Using Schema Matching to Simplify Heterogeneous Data Translation," Proc. Int'l Conf. Very Large Data Bases, pp. 122-133, 1998.
[11] Barbara Staudt Lerner, "A Model for Compound Type Changes Encountered in Schema Evolution," ACM Transactions Database Systems, Vol. 25, No. 1, pp. 83-127, 2000.
[12] AnHai Doan, Pedro Domingos, and Alon Halevy, "Learning to Match Schemas of Data Sources: A Multistrategy Approach," Machine Learning, Vol. 50, No. 3, pp. 279-301, 2003.
[13] Renee J. Miller, Laura M. Haas, Mauricio A. Hernandez, Lingling Yan, C. T. Howard Ho, Ronald Fagin, and Lucian Popa, "The Clio Project: Managing Heterogeneity," SIGMOD Record, Vol. 30, No. 1, pp. 78-83, 2001.
[14] Jayant Madhavan, Philip A. Bernstein, and Erhard Rahm, "Generic Schema Matching with Cupid," Proc. Int'l Conf. Very Large Data Bases, pp. 49-58, 2001.
[15] Hong Su, Harumi Kuno, and Elke A. Rundensteiner, "Automating the Transformation of XML Documents," Proc. Int'l Workshop Web Information and Data Management, pp. 68-75, 2001.
[16] Mong Li Lee, Wynne Hsu, LiangHuai Yang, and Xia Yang, "XClust: Clustering XML Schemas for Effective Integration," Proc. Int'l Conf. Information and Knowledge Management, pp. 292-299, 2002.
[17] Hong-Hai Do and Erhard Rahm, "COMA - A System for Flexible Combination of Schema Matching Approaches," Proc. Int'l Conf. Very Large Data Bases, pp. 610-621, 2002.
[18] Sergey Melnik, Hector Garcia-Molina, and Erhard Rahm, "Similarity Flooding - A Versatile Graph Matching Algorithm," Proc. Int'l Conf. Data Engineering, pp. 117-128, 2002.
[19] Li Xu and David W. Embley, "Discovering direct and indirect matches for schema elements," Proc. Int'l Conf. Database Systems for Advanced Applications, pp. 39-46, 2003.
[20] Robin Dhamankar, Yoonkyong Lee, AnHai Doan, and Alon Halevy, "iMAP: Discovering Complex Semantic Mappings between Database Schemas," Proc. Int'l Conf. SIGMOD, pp. 383-394, 2004.
[21] George A. Miller, "WordNet: A Lexical Database for English," Communications of the ACM, Vol. 38, No. 11, pp. 39-41, 1995.
[22] Hong Hai Do, Sergey Melnik, and Erhard Rahm, "Comparison of Schema Matching Evaluations," Lecture Notes in Computer Science, Vol. 2593, pp. 221-237, 2002.

이 경 호

1995년 연세대학교 전산과학과 졸업(학사). 1997년 연세대학교 컴퓨터과학과 졸업(석사). 2001년 연세대학교 컴퓨터과학과 졸업(박사). 2001년 National Institute of Standards and Technology(NIST) 객원연구원. 2002년~현재 연세대학교 컴퓨터과학과 부교수. 관심분야는 Internet Computing, Service-Oriented Computing, Multimedia Document Engineering

이 준 승

2003년 연세대학교 컴퓨터과학과 졸업(학사). 2005년 연세대학교 컴퓨터과학과 졸업(석사). 2005년 3월~현재 삼성전자 연구원. 관심분야는 Internet Computing, Multimedia Document Engineering