Journal of Korea Multimedia Society Vol. 11, No. 7, July 2008(pp. 940-955)

도메인 온톨로지에 기반한 XML 스키마의 통합

강혜란[†], 이경호[††]

요 약

동일한 도메인에 속하고 의미가 유사한 XML 문서들이라도 서로 다른 XML 스키마를 따르는 경우가 많다. 그러므로 XML 문서로부터 정보를 추출하고 통합하기 위해서는 의미가 유사한 XML 스키마들을 통합하는 방법이 필요하다. 본 논문은 동일한 도메인에서 사용되는 XML 스키마들을 의미를 정확하게 보존하면서 통합하는 방법을 제안한다. 제안된 방법은 사전과 함께 도메인 온톨로지를 이용하여 어휘 간 유의어 및 상하위어 관계를 검사하고 이를 스키마 통합 과정에 활용한다. 특히 본 논문은 엘리먼트 및 애트리뷰트가 갖고 있는 구조적 정보를 활용하여 유의어 및 상하위어 관계를 보다 정확하게 검사한다. 그리고 정교한 수준의 연산자 통합과 연산자 최적화 규칙을 제안한다. 제안된 방법의 성능을 평가하기 위해서 다양한 도메인의 XML 스키마를 대상으로 실험한 결과, 도메인 온톨로지와 제안된 방법의 어휘 간 구조적 관계를 이용할 경우 통합 스키마의 정확률과 재현율이 향상됨을 확인하였다.

Integration of XML Schemas Based on Domain Ontology

Haeran Kang[†], Kyong-Ho Lee[††]

ABSTRACT

Semantically similar XML documents in the same application domain might often conform to different schemas. To uniformly view and query such XML documents, we need an efficient method of integrating XML schemas. This paper proposes a sophisticated method for integrating XML schemas in the same application domain. To compute mapping relationships between schemas, the proposed method utilizes various relationships, such as synonyms and hypernyms, between lexical items based on dictionaries and domain ontologies. Particularly, the relationships between lexical items are elaborated by taking their structural information into account. In addition, this paper proposes a more accurate method for integrating compositors. Experimental results with schemas in various application domains show that the utilization of domain ontologies and the structural relationships between lexical items enhance the precision and recall of integrated schemas.

Key words: XML schema(XML 스키마), schema integration(스키마 통합), domain ontology(도메인 온톨로지)

1. 서 론

데이터 및 문서의 논리적 구조를 표현할 수 있는 XML (eXtensible Markup Language) [1]은 인터넷을 비롯한 다양한 분야에서 정보 표현 및 교환을 위한 표준으로 널리 사용되고 있다. XML 데이터로부터 정보를 추출 및 검색하기 위해서는 XML 스키마를 통해 문서의 구조와 의미가 파악되어야 한다. 서

※ 교신저자(Corresponding Author) : 이경호, 주소 : 서울특별시 서대문구 신촌동 134(126-749), 전화 : 02)2123-5712, FAX : 02)365-2579, E-mail : khlee@cs.yonsei.ac.kr
접수일 : 2007년 11월 9일, 완료일 : 2008년 5월 6일
[†] 준회원, 연세대학교 컴퓨터과학과 박사과정 (E-mail : hrkang@icl.yonsei.ac.kr)
[††] 종신회원, 연세대학교 컴퓨터과학과 부교수
※ 이 논문은 서울시 산학연 협력사업(10705)의 지원에 의해 연구되었음

로 다른 XML 스키마를 따르는 문서로부터 정보를 추출하기 위해서 해당 스키마에 대한 질의를 반복적으로 수행하여야 한다. 이로 인하여 정보 검색 비용이 증가하고 검색 결과의 질 또한 저하되게 된다. 그러므로 유사한 XML 스키마들을 통합할 수 있는 방법이 요구된다.

XML 스키마는 제작자에 따라 같은 의미의 스키마라도 다르게 표현될 수 있다. 이러한 XML 스키마의 다형성 때문에 동일한 도메인에서조차 다양한 스키마가 존재한다. 그러므로 이러한 동일한 도메인의 XML 스키마를 통합하기 위해서, 해당 도메인에 대한 정보를 활용할 수 있다. 해당 도메인에 대한 정보를 이용하면, 엘리먼트 및 애트리뷰트 간의 유의어 및 상하위어 관계를 보다 정확하게 계산할 수 있다.

Huma 등 [2]을 제외한 기존 방법의 대부분은 어휘 사전 등의 단순한 정보를 이용하며 온톨로지 등의 도메인 정보를 활용하지 않는다. XML 스키마는 반구조적이기 때문에 XML 스키마에서 유의어 및 상하위어 관계와 같은 구조적 정보를 찾을 수 있다. 그러나 기존 방법은 다른 스키마에 속한 두 개의 개념들 간의 구조적 유사도를 계산할 뿐, 같은 스키마에 속한 두 개의 개념들 간의 구조적 유사도는 고려하지 않는다. 또한 기존 연구는 연산자 간 통합 방법에 있어서 제한적이며 통합 스키마를 최적화하는데 있어서도 제한적이다.

제안된 방법은 어휘 사전에서 일반적인 정보를 활용할 뿐 아니라 도메인 온톨로지에서 각 도메인 내에서 특수하게 사용되는 정보를 활용한다. 이러한 도메인 정보를 이용하여 어휘 간 유의어 및 상하위어 관계를 계산하기 때문에 보다 정확한 통합 XML 스키마를 생성한다. 그리고 제안된 방법은 같은 스키마에 속한 두 개의 개념들 간의 구조적 정보를 이용하여 보다 정확하게 스키마를 통합하였다. 또한 제안된 방법은 보다 정교한 규칙을 이용하여 연산자를 통합 및 최적화한다.

제안된 방법은 오직 XML 스키마만을 대상으로 하기 때문에 XML 스키마들을 다른 개념 모델들로 변환시킬 필요가 없다. 그러므로 제안된 방법은 XML 스키마 통합에 더욱 효율적이고, 변환을 필요로 하는 방법보다 입력 스키마들의 의미를 더 많이 보존할 수 있다. 또한 제안된 방법의 모든 과정은 사용자의 개입을 필요로 하지 않는다. 사용자의 개입이 필요하다면, 통합해야 할 소스의 수가 증가할수록 시간 복잡도와 에러 발생 가능성이 급격하게 증가한다.

제안된 방법의 성능을 평가하기 위해 다양한 도메인의 스키마를 대상으로 실험한 결과, 도메인 온톨로지를 이용할 경우 정확률과 재현율과 통합 스키마의 크기가 개선되었음을 확인할 수 있었다. 마찬가지로 제안된 어휘 간 구조적 정보를 이용한 유의어 및 상하위어 관계 추출 방법은 정확률과 재현율과 통합 스키마의 크기를 개선시켰다. 또한 연산자 최적화 규칙을 통하여 통합 스키마의 크기가 감소하였다.

본 논문의 구성은 다음과 같다. 2절에서는 관련 연구들의 장단점을 비교하고, 3절에서는 제안된 온톨로지에 기반한 스키마 통합 방법을 자세히 기술한다. 제안된 방법은 크게 어휘 간 유의어 및 상하위어 관계 추출, 통합, 그리고 최적화의 세 단계로 구성된다. 4절에서는 실험을 통해 제안된 방법의 유효성을 검증한다. 마지막으로 5절에서는 결론 및 향후 연구 방향을 기술한다.

2. 관련 연구

본 절에서는 표 1에서 요약한 것처럼 기존의 XML 스키마 통합 방법의 특징을 간략히 기술한다. 기존의 XML 스키마 통합 방법 [2-12]은 먼저 개념 모델을 사용하는지에 따라 크게 두 가지로 구분될 수 있다. 첫 번째 방법은 서로 다른 형태의 XML 스키마를 제안된 개념 모델로 변환한 후, 이를 통합한다. 두 번째 방법은 이러한 개념 모델로의 변환 없이 XML 스키마들을 통합한다. 첫 번째 방법은 개념 모델로의 변환 과정이 요구되지만 변환 이후의 통합 과정은 덜 복잡하다는 특징을 갖는다. 두 번째 방법은 개념 모델로의 변환을 필요로 하지 않으나 XML 스키마를 통합하는 과정에서 내재된 복잡성을 해결해야 한다 [3]. 그러나 첫 번째 방법은 개념 모델이 XML 스키마들의 모든 특징을 포함하지 않을 경우, 원본 스키마의 정보를 상실할 수 있다.

또한 XML 스키마 통합 방법은 도메인 온톨로지 등 부가 정보의 사용 정도에 따라 구분할 수 있다. 일반적인 XML 스키마 통합 방법은 같은 도메인에 속한 스키마를 대상으로 하기 때문에 해당 도메인에 대한 정보를 이용할 수 있다. 그러나 Huma 등 [2]을 제외한 기존 방법은 어휘 사전 등의 단순한 정보만을

표 1. XML 스키마 통합에 관한 연구

저자	연도	특징	개념 모델	도메인 정보 활용
Huma 등 [2]	2005	지역 스키마의 엘리먼트 이름을 전역 온톨로지 상의 기준 용어로 바꾼 후 통합	-	○
Mello와 Heuser [3]	2005	개념 모델로 변환 후, 의미적으로 유사한 스키마들을 식별 후 통합하여 통합 스키마를 생성	ORM/NIAM	×
Meo 등 [4]	2006	WordNet을 이용하여 두 XML 스키마 간 특성을 추출하고 엘리먼트들을 통합하고 최적화	-	×
Cruz 등 [5]	2004	개념 모델인 로컬 RDF 온톨로지들 간 유사도를 기반으로 PROMPT [13]를 이용하여 통합 온톨로지를 생성	RDF	×
Jeong과 Hsu [6]	2003	클러스터링된 DTD로부터 내재된 문법을 추출하고 유사한 상태 및 관계를 통합 및 재구성	-	×
Yang 등 [7]	2003	개념 모델인 ORA-SS 스키마의 세트에서 충돌과 번복성을 제거하여 통합 스키마를 생성	ORA-SS	×
Castano 등 [8]	2002	X-formalism [14]을 이용하여 만든 개념 모델들 간 유사한 클래스들을 통합하여 글로벌 X-class를 생성	X-class	○
Lee 등 [9]	2002	XML 스키마 정의 언어인 XQuerySD를 이용하여 엘리먼트 및 속성 간 충돌을 해결	-	×
Mello 등 [10]	2002	개념 모델인 ORM/NIAM들 간 의미적으로 유사한 개념 간 충돌을 해결하여 통합 후 재구성	ORM/NIAM	×
Passi 등 [11]	2002	XML 스키마를 XSDM으로 변환한 후, 엘리먼트들 간 의미적 관계에 따라 충돌을 해결하고 통합	XSDM	×
Zhang과 Liu [12]	2001	XML 스키마를 UML로 변환한 후, 이들 간 충돌을 해결하고 재구성하여 통합 개념 모델을 형성	UML	×

이용하며 온톨로지 등의 도메인 정보를 활용하지 않는다.

Huma 등 [2] 은 XML 스키마 내에 있는 엘리먼트의 이름을 통합 스키마 내에 있는 용어로 대체함으로써 지역 온톨로지들을 생성한다. 두 개의 개념들을 매칭 시키기 위해, 그들은 이름과 구조적 유사도를 계산한다. 그러나 이 방법은 다른 스키마에 속한 두 개의 개념들 간의 구조적 유사도를 계산할 뿐, 같은 스키마에 속한 두 개의 개념들 간의 구조적 유사도는 고려하지 않는다.

Mello와 Heuser [3] 는 XML 스키마들을 ORM/NIAM (Object with Roles Model/Natural language Information Analysis Method) 모델들로 변환 시키고 나서 그들을 통합한다. 그러나 이 방법은 순서 연산자와 같은 XML 스키마들의 모든 특징들을 포함하지 않는 모델로 변환하며, 복잡한 연산자를 자동적으로 통합하지 못한다는 한계점이 있다.

Meo 등 [4] 은 엘리먼트 또는 애트리뷰트의 의미를 결정하기 위해 그들과 개념이 비슷한 이웃 엘리먼트 및 애트리뷰트들을 검사한다. 그 후에 유사한 엘리먼트들을 통합하고 동음이의어 엘리먼트들의 이름을 바꾼다. 그리고 의미적으로 중복되거나 모호한 부분을 제거한 후 최종적으로 통합 스키마를 생성한다. 이 방법에서, 사용자는 통합 작업이 이루어지는 엄밀도 (severity level)를 선택할 수 있다. 그러나 각 엘리먼트와 애트리뷰트의 모든 이웃들을 검사하는 것은 많은 수의 XML 스키마들에 적용될 때 심각한 과부하를 초래할 수 있다.

Cruz 등 [5] 은 XML 스키마들을 RDFS 온톨로지들로 변환하기 때문에 소스 정보가 손실된다. 또한 그들은 통합 온톨로지와 지역 온톨로지들 간의 매핑을 수동으로 생성한다.

본 논문에서는 Huma 등 [2]을 제외한 기존 방법이 동일한 도메인에 속하는 스키마들을 통합하면서, 어휘 사전에서 일반 정보만을 이용하고 해당 도메인 정보를 이용하지 않았던 한계점을 해결하고자 한다. 그리고 기존 방법 [2~12]이 입력 스키마의 구조적 정보를 충분히 활용하지 않았기 때문에 정확한 통합

스키마를 생성하지 못한 점을 극복하고자 한다. 또한 기존 연구 [2~12]가 복잡한 연산자들을 정확하게 통합하는 방법과 연산자를 최적화 하는 방법을 제안하지 못한 점을 해결하고자 한다.

본 논문에서는 Huma 등 [2], Meo 등 [4], Jeong과 Hsu [6], 그리고 Lee 등 [9] 을 제외한 기존 방법이 XML 스키마를 개념 모델로 변환함으로써 입력 스키마의 의미를 상실하여 정확한 통합 스키마를 추출하지 못한 점을 해결하고자 한다. 그리고 대부분의 기존 방법들이 통합 스키마를 추출하는 과정에서 사용자의 개입을 필요로 하므로 통합 과정에 심각한 병목 현상이 발생되는 문제를 극복하고자 한다.

3. 스키마 통합

제안된 방법은 그림 1과 같이 크게 어휘 간 유의어 및 상하위어 관계 추출, 통합, 최적화의 세 단계로 구성된다.

제안된 방법은 XML 스키마의 효율적인 표현을 위해서 Rhim과 Lee [15] 가 제안한 스키마 트리라는 문서모델을 이용한다. 스키마 트리를 구성하는 노드는 엘리먼트, 애트리뷰트, 그리고 연산자 노드로 나누어진다. 엘리먼트 및 애트리뷰트 노드는 이름을 레이블로 갖는다. 단말 엘리먼트 및 애트리뷰트는 단말 노드 (leaf node)에 해당되며 단순 데이터 타입 (simple data type)을 속성으로 갖는다. 하위 구조를 갖는 엘리먼트일 경우 중간 노드 (internal node)에 해당한다. 또한 연산자 노드의 종류로는 선택, 순서, 그리고 all 연산자 노드가 있고, 이들은 각각 '|', ',', 그리고 '&'를 레이블로 갖는다. 노드의 속성인 빈도

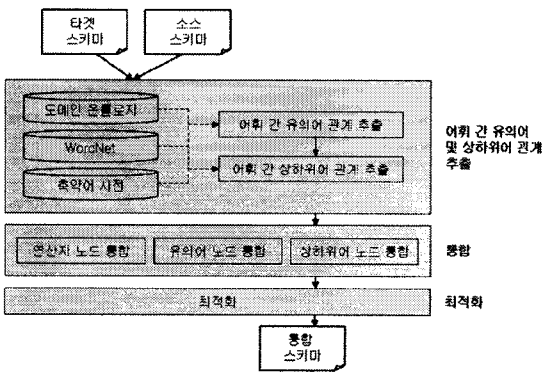


그림 1. 스키마 통합 과정

지시자는 '?', '*', '+', 또는 (최소값, 최대값) 중의 하나를 레이블로 갖는다. 빈도 지시자의 레이블이 (1, 1)인 경우 빈도 지시자는 생략될 수 있다.

3.1 어휘 간 유의어 및 상하위어 관계 추출

제안된 방법은 WordNet [16], 축약어 사전, 그리고 도메인 온톨로지를 이용하여 노드들의 어휘 간 유의어 및 상하위어 관계를 찾는다. 본 논문에서 유의어는 일반적인 의미의 유의어와 동의어를 모두 포괄하는 의미로 사용된다. 본 논문에서 이용하는 도메인 온톨로지는 OWL (Web Ontology Language)로 기술된다. 어휘 간 유의어 및 상하위어 관계는 엘리먼트 및 애트리뷰트 노드의 이름들 간에만 존재한다. 제안된 방법은 먼저 어휘 간 유의어 관계를 찾은 후, 유의어 관계가 존재하지 않는 어휘를 대상으로 상하위어 관계를 찾는다.

3.1.1 어휘 간 유의어 관계 추출

제안된 방법은 두 어휘 W_s와 W_t 간의 유사도를 계산하기 위해서 식 (1)과 같이 *LexicalSimilarity* (W_s, W_t)를 정의한다. 만일 유사도 값이 임계 값 TH_{sim}보다 클 경우, W_s와 W_t는 유의어 관계를 갖는다고 간주한다.

$$LexicalSimilarity(W_s, W_t) = \begin{cases} \sum_{i=1}^{n} (TokenSimilarity(T_{s_i}, T_{t_i}))/n, & \text{두 어휘 간에 구조 가중치가 0이 아닌 토큰의 수가 동일한 경우} \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

W_s : 소스 스키마의 어휘　　W_t : 타겟 스키마의 어휘
n : 어휘 W_s의 구조 가중치가 0이 아닌 토큰들의 수
T_{s_i} : 어휘 W_s의 구조 가중치가 0이 아닌 토큰들 중 i번째 토큰, $1 \le i \le n$
T_{t_i} : 어휘 W_t의 구조 가중치가 0이 아닌 토큰들 중 i번째 토큰, $1 \le i \le n$

식 (1)을 이용하여 두 어휘 간 유사도를 계산하기 위하여 우선 두 어휘의 구조 가중치가 0이 아닌 토큰의 수가 동일한지 동일하지 않은지 구분한다. 구조 가중치는 현재 토큰과 이 토큰의 가장 가까운 상위 엘리먼트 노드가 갖는 토큰들 간 유사성을 추출함으

로써, 현재 토큰이 트리 구조 상에서 갖는 구조적 중요도를 구하기 위한 것이다. 엘리먼트 및 애트리뷰트 노드는 일반적으로 그 노드의 가장 가까운 상위 엘리먼트 노드의 의미를 포함하고 있으므로, 현재 노드가 갖는 현재 토큰이 그 노드의 가장 가까운 상위 엘리먼트 노드의 토큰의 유의어이면, 현재 토큰을 생략하여도 현재 노드의 의미는 변하지 않는다. 구조 가중치는 이러한 엘리먼트 및 애트리뷰트 노드가 갖고 있는 구조적 정보를 기반으로 한다.

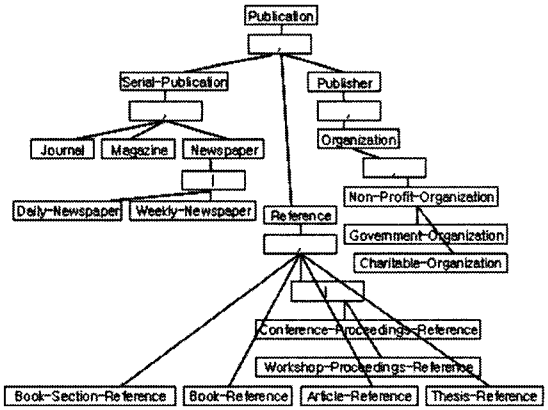
현재 토큰이 가장 가까운 상위 엘리먼트 노드의 어떤 토큰의 유의어일 경우 현재 토큰의 구조가중치는 0이고, 어떤 토큰과도 유의어 관계가 아닐 경우 현재 토큰의 구조 가중치는 1이다. 현재 어휘의 구조 가중치가 0인 토큰이 다른 어휘의 토큰들과 유의어 관계를 갖는지 여부는, 현재 어휘가 다른 어휘들과 유의어 또는 상하위어 관계를 갖는지 여부에 영향을 미치지 않는다고 가정한다.

두 어휘의 구조 가중치가 0이 아닌 토큰의 수가 동일한 경우, 두 어휘를 대문자나 특수문자를 기준으로 토큰화하여 각 토큰 간 유사도 *TokenSimilarity*를 추출한다. 본 논문에서, 두 토큰 간 *TokenSimilarity*가 임계값 $TH_{tokensim}$보다 클 경우, 두 토큰은 유의어 관계를 갖는다고 간주한다.

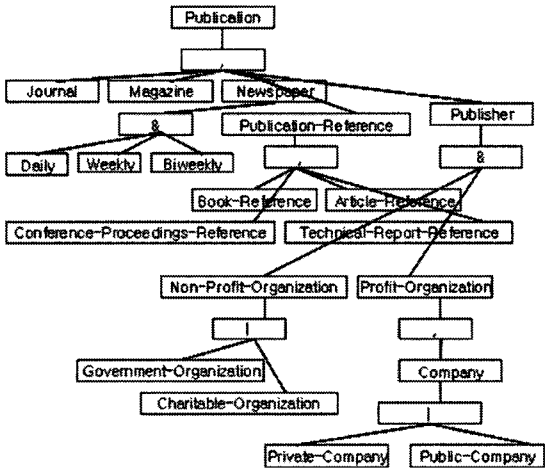
토큰 간 유사도를 추출하기 위해서 우선 축약어 사전에 해당 토큰과 같은 이름의 축약어가 있을 경우에 그 토큰을 해당 축약어의 전체 이름으로 대체한다. 대체 과정 이후, 두 토큰 간 문자열이 동일한 경우, 두 토큰 간 유사도는 1이다. 만약 두 토큰 간 문자열이 동일하지 않은 경우 WordNet 상에서 두 토큰 간에 유의어 관계가 존재하거나, 두 토큰과 같은 이름을 갖는 도메인 온톨로지 상의 클래스 (또는 개체) 간에 *equivalentClass* (또는 *sameAs*) 관계가 존재하면, 두 토큰 간 유사도는 0.8로 계산한다. 위의 경우들에 해당하지 않는 경우, 두 토큰 간 유사도는 0으로 간주한다.

그림 2는 소스 및 타겟 스키마 트리와 도메인 온톨로지의 예이다. 특히, 그림 2(c)는 클래스 간 *subClassOf* 관계를 효율적으로 표현하기 위해 UML (Unified Modeling Language) 클래스 다이어그램의 클래스 간 상속 (inheritance) 관계로 표현한 것이다.

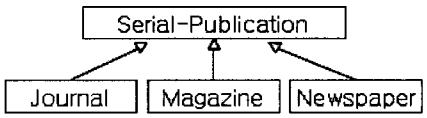
그림 2(a)의 어휘 *Reference*와 그림 2(b)의 어휘 *Publication-Reference* 간 유의어 관계를 추출하는



(a) 소스 스키마 트리



(b) 타겟 스키마 트리



(c) 도메인 온톨로지

그림 2. 소스 및 타겟 스키마 트리와 도메인 온톨로지의 예

방법은 다음과 같다. 그림 2(a)의 어휘 *Reference*의 토큰 *Reference*는 가장 가까운 상위 엘리먼트의 유일한 토큰 *Publication*의 유의어가 아니므로, 토큰 *Reference*의 구조 가중치는 1이다. 마찬가지로, 그림 2(b)의 어휘 *Publication-Reference*의 가장 가까운 상위 엘리먼트의 유일한 토큰은 *Publication*이므로, 어휘 *Publication-Reference*의 토큰 *Publication*과 토큰 *Reference*의 구조 가중치는 각각 0과 1이다. 그러므로 두 어휘에서 구조 가중치가 0이 아닌 토큰의

수가 동일하고, 서로 유의어이므로 두 어휘는 유의어 관계를 갖는다.

3.1.2 어휘 간 상하위어 관계 추출

어휘 W_s가 어휘 W_t 보다 더 한정적인 의미를 가질 경우 W_t를 W_s의 상위어 (반대로, W_s를 W_t의 하위어)라고 정의한다 [17]. 어휘 간 상하위어 관계를 추출하는 방법은 다음의 두 단계로 이루어진다. 우선 WordNet 또는 온톨로지 상에 어휘 간 상하위어 관계가 직접적으로 기술되어 있는지 검사하고, 기술되어 있지 않다면 그림 3의 어휘 간 상하위어 유추 조건을 만족하는지 검사한다.

어휘 간 상하위어 관계가 WordNet 또는 온톨로지 상에 직접적으로 기술되어 있는지 검사하는 방법은 다음과 같다. 만약 WordNet 상에서 소스 스키마 어휘 W_s가 타겟 스키마 어휘 W_t의 상위어이면, W_s가 W_t의 상위어라고 간주한다. 그리고 만약 도메인 온톨로지 상에서 W_t를 이름으로 갖는 임의의 클래스가 W_s를 이름으로 갖는 임의의 클래스의 하위 클래스(*subClassOf*)로 정의되어 있다면, W_s가 W_t의 상위어라고 간주한다.

예를 들어, 그림 2(a)의 어휘 *Organization*과 그림 2(b)의 어휘 *Profit-Organization* 간 상하위어 관계를 추출하는 방법은 다음과 같다. 우선 WordNet과 온톨로지 상에 두 어휘 간 상하위어 관계가 직접적으로 기술되어 있지 않다. 그리고 어휘 *Organization*의 오른쪽 끝 토큰 *Organization*은 어휘 *Profit-Organization*의 오른쪽 끝 토큰 *Organization*의 유의어이므로 그림 3의 어휘 간 상하위어 유추 조건 2(a)를 만족한다. 그리고 어휘 *Profit-Organization*의 구조 가중치가 1인 토큰 *Profit*은 어휘 *Organization*의 유일한 토큰 *Organization*의 유의어가 아니므로 그 조건 2(b)를 만족한다. 마지막으로 어휘 *Organization*의 구조 가중치가 1인 유일한 토큰 *Organization*은, 어휘 *Profit-Organization*의 토큰 *Organization*의 유의어이므로 그 조건 2(c)를 만족한다. 따라서 조건 2에 의해서 어휘 *Organization*은 어휘 *Profit-Organization*의 상위어에 해당한다.

소스 스키마의 어휘를 W_s, 타겟 스키마의 어휘를 W_t라고 하면,

1. W_s와 W_t가 아래 조건 (a)와 (b)를 모두 만족하면, W_s가 W_t의 상위어라고 간주한다.
   (a) W_s의 어떤 토큰이 W_t의 어떤 토큰의 상위어이다.
   (b) W_s의 모든 토큰은 W_t의 어떤 토큰의 상위어 또는 유의어이다.

2. W_s와 W_t가 아래 조건 (a), (b), 그리고 (c)를 모두 만족하면, W_s가 W_t의 상위어라고 간주한다.
   (a) W_s의 오른쪽 (또는 왼쪽) 끝 토큰은 W_t의 오른쪽 (또는 왼쪽) 끝 토큰의 유의어이다.
   (b) W_t의 구조 가중치가 1인 어떤 토큰은, W_s의 어떤 토큰의 유의어도 아니다.
   (c) W_s의 구조 가중치가 1인 모든 토큰은, W_t의 어떤 토큰의 유의어이다.

그림 3. 어휘 간 상하위어 유추 조건

3.2 통합

제안된 방법은 3.1절에서 추출된 소스 스키마와 타겟 스키마의 어휘 간 유의어 및 상하위어 관계를 이용하여 소스 스키마의 노드들을 너비 우선 탐색 (Breadth-First Search) 하면서 두 스키마의 노드들을 통합한다. 통합 방법은 연산자 노드 또는 유의어 노드 또는 상하위어 노드와 같은 통합될 노드의 종류에 따라 분류된다.

3.2.1 연산자 노드 통합

현재 엘리먼트 노드와 그 노드와 가장 가까운 후손 엘리먼트 노드 간에는 임의 개수의 연산자 노드가 존재할 수 있다. 그러므로 본 논문에서는, 현재 엘리먼트 노드와 그 노드와 가장 가까운 후손 엘리먼트 노드 간에 존재하는 전체 연산자 노드들의 집합을 현재 엘리먼트 노드의 자식 연산자 집합 또는 그 후손 엘리먼트 노드의 상위 연산자 집합이라고 지칭한다.

제안된 방법은 두 유의어 노드의 자식 연산자 집합들 간 통합, 상위어 노드의 상위 연산자 집합과 하위어 노드의 상위 연산자 집합 간 통합, 그리고 상위어 노드의 자식 연산자 집합과 하위어 노드의 상위 연산자 집합 간 통합의 세 가지 경우에 연산자 노드 통합을 실행한다.

이러한 연산자 노드 통합은 두 연산자 집합이 모두 하나의 연산자 노드로 이루어진 경우, 두 연산자 집합 모두 하나 이상의 연산자 노드로 이루어졌으며 적어도 한 개 연산자 집합이 두개 이상의 연산자 노드로 이루어진 경우, 하나의 연산자 집합은 연산자 노드를 갖지만 다른 연산자 집합은 연산자 노드를

갖지 않을 경우의 세 가지 경우에 따라 통합 방법이 나누어진다.

Case 1 : 두 연산자 집합이 모두 하나의 연산자 노드로 이루어진 경우

두 연산자 집합이 모두 하나의 연산자 노드로 이루어진 경우에는 표 2의 연산자 통합 규칙 I 을 이용해 통합한다. 표 2의 연산자 통합 규칙 I 을 통해 통합된 연산자 집합은 순서 또는 선택 연산자노드의 자식 노드로 all 연산자 노드를 갖는 경우가 있다. 이는 순서 또는 선택 연산자 노드의 내용모델이 all 연산자 노드를 직접적으로 포함하는 것이 아니라, 순서 또는 선택 연산자 노드의 내용모델이 그룹 (group)을 포함하고, 이 그룹이 내용 모델로 all 연산자 노드를 갖는 경우를 의미한다.

관련 연구 중에서 연산자 통합 방법을 제시하고 있는 논문은 Mello와 Heuser [3], Mello 등 [10], Passi 등 [11], 그리고 Zhang과 Liu [12] 의 방법이다. 제안된 연산자 통합 규칙 I 은 Passi 등 [11] 의 방법과 Zhang과 Liu [12] 의 방법과 유사하지만, 순서 연산자 노드와 선택 연산자 노드의 통합 방법은 기존 방법들과 다르다. 소스 및 타겟 연산자 노드의 자식 연산자 간에 유의어 및 상하위어 관계를 갖는 노드들은 통합된 순서 연산자 노드의 자식 노드들이 되고, 그 노드들의 빈도 지시자의 최소값은 영이 된다.

그러므로 Zhang과 Liu [12] 의 방법은 선택 연산자 노드가 갖던 의미는 보존하지만, 순서 연산자 노드가 갖고 있던 자식 노드들 간의 순서의 의미를 완전히 상실한다. 반면, 제안된 방법은 선택 연산자 노드가 갖고 있던 XML 문서 내에서 자식 노드들 중 하나만이 출현한다는 의미는 상실하지만, 자식 노드들의 빈도 지시자 최소값을 0으로 수정함으로써 XML 문서 내에서 각 자식 노드들이 출현하지 않을 수 있다는 의미는 보존한다. 그리고 순서 연산자 노드가 갖던 의미를 보존한다.

그림 2(a)의 노드 *Publisher*의 자식 연산자 집합과 그림 2(b)의 노드 *Publisher*의 자식 연산자 집합을 통합하는 예는 다음과 같다. 그림 2(a)의 연산자 집합은 하나의 순서 연산자 노드로, 그림 2(b)의 연산자 집합은 하나의 all 연산자 노드로 이루어져 있으므로

표 2. 연산자 통합 규칙 I

소스 노드	타겟 노드	통합 조건	통합된 연산자 집합
&*	&		all
|	|		choice
,	,	소스 연산자의 자식 노드들이 타겟 연산자의 자식 노드들과 유의어 또는 상하위어 관계를 갖지 않을 경우	소스 및 타켓 sequence 연산자를 자식으로 갖는 choice 연산자 노드 (단, 소스 및 타겟 연산자의 자식 노드들은 각각 통합된 소스 및 타켓 sequence의 자식 노드들이 된다.)
		위의 통합 조건을 만족하지 않는 경우	sequence (longest common subsequence approach ([18]))
&	|	타겟 연산자 노드의 자식 노드들이 모두 엘리먼트이고 빈도수가 1이하인 경우	all
		위의 통합 조건을 만족하지 않는 경우	all을 자식으로 갖는 choice (단, 소스 및 타겟 연산자 노드의 자식 노드들은 각각 통합된 all 및 choice의 자식 노드들이 되고, 소스 및 타겟 연산자 노드의 자식 노드들 간에 유의어 및 상하위어 관계를 갖는 노드들은 통합된 choice의 자식 노드들이 된다.)
&	,	타겟 연산자 노드의 자식 노드들이 모두 엘리먼트이고 빈도수가 1이하인 경우	all
		위의 통합 조건을 만족하지 않는 경우	all 및 sequence를 자식으로 갖는 choice (단, 소스 및 타겟 연산자의 자식 노드들은 각각 통합된 all 및 sequence의 자식 노드들이 된다.)
,	|		choice를 자식으로 갖는 sequence (단, 소스 및 타겟 연산자 노드의 자식 노드들은 각각 통합된 sequence 및 choice의 자식 노드들이 되고, 소스 및 타겟 연산자 노드의 자식 노드들 간에 유의어 및 상하위어 관계를 갖는 노드들은 통합된 sequence의 자식 노드들이 된다.)

* '&', '|', ','는 각각 all, choice, sequence 연산자를 의미함.

두 연산자 집합을 표 2의 통합 규칙을 이용하여 통합한다. 그림 2(a)의 순서 연산자 노드의 유일한 자식은 엘리먼트이고, 빈도수는 생략되어 있으므로 최대값과 최소값 모두 1이다. 그러므로 통합된 연산자 집합은 all 연산자 노드가 된다.

Case 2 : 두 연산자 집합 모두 하나 이상의 연산자 노드로 이루어졌으며 적어도 한 개의 연산자 집합이 두개 이상의 연산자 노드로 이루어진 경우

두 연산자 집합 모두 하나 이상의 연산자 노드로 이루어졌으며 적어도 한 개의 연산자 집합이 두개 이상의 연산자 노드로 이루어진 경우에는 그림 4의 연산자 통합 규칙II를 이용해 통합한다. 그림 4의 연산자 통합 규칙II를 이용하여 통합하는 과정에서 소스 순서 연산자 노드와 타겟 선택 연산자 노드를 일대일 통합할 경우 타겟 선택 연산자가 타겟 순서 연산자 노드의 자식이라면, 통합된 순서 연산자 노드의 자식들은 소스 순서 연산자의 노드 자식들 간의 순서뿐 아니라 타겟 순서 연산자 노드의 자식들 간의 순서와도 일치하는 순서를 갖는다.

소스 및 타겟 연산자 집합의 노드들을 깊이가 가장 깊은 연산자 노드들부터 깊이가 가장 얕은 연산자 노드들까지 일대일로 [표 2]의 연산자 통합 규칙 I 을 이용해 통합하고 이렇게 통합된 연산자 집합을 C_I라고 한다. 이러한 일대일 연산자 통합 후, 소스 및 타겟 연산자 집합에 통합되지 않은 연산자 노드가 없을 때까지 다음을 반복한다. 소스 및 타겟 연산자 집합에 더 이상 통합할 연산자 노드가 없으면, C_I가 소스 및 타겟 연산자 집합의 통합된 연산자 집합이 된다. 소스 및 타겟 연산자 집합의 현재 통합 대상 연산자 노드를 각각 C_S 및 C_T라고 하면,

1. 만약 C_S는 존재하고 C_T는 존재하지 않으며, C_S와 C_I의 깊이가 가장 얕은 연산자 노드의 종류가 다른 경우, 소스 연산자 집합에서 C_S의 자식 연산자 집합을 C_I로 대체한 연산자 집합이 소스 연산자 집합이자 C_I가 된다.
2. 만약 C_S는 존재하고 C_T는 존재하지 않으며, C_S와 C_I의 깊이가 가장 얕은 연산자 노드의 종류가 같은 경우, 소스 연산자 집합에서 C_S의 자식 연산자 집합을 삭제하고 C_S와 C_I의 깊이가 가장 얕은 연산자 노드를 [표 2]의 연산자 통합 규칙 I 을 이용해 통합한 연산자 집합이 소스 연산자 집합이자 C_I가 된다.

그림 4. 연산자 통합 규칙II

그림 2(a)의 노드 *Reference*의 자식 연산자 집합과 그림 2(b)의 노드 *Publication-Reference*의 자식 연산자 집합을 통합하는 예는 다음과 같다. 그림 2(a)의 연산자 집합은 선택 연산자 노드를 자식 노드로 갖는 순서 연산자 노드이고, 그림 2(b)의 연산자 집합은 하나의 순서 연산자 노드로 이루어져 있으므로 두 연산자 집합을 그림 4의 통합 규칙을 이용하여 통합한다. 우선, 그림 2(a)의 연산자 집합의 가장 깊은 연산자 노드인 선택 연산자 노드와 그림 2(b)의 유일한 연산자 노드인 순서 연산자 노드를 표 2의 통합 규칙을 이용해 통합한다. 이 때 통합된 연산자 집합은 노드 *Book-Reference*, 선택 연산자 노드, 노드 *Conference-Proceedings-Reference*, 노드 *Article-Reference*, 그리고 노드 *Technical-Report-Reference*를 자식 노드들로 갖는 순서 연산자 노드가 된다. 이 통합된 연산자 집합 내의 선택 연산자 노드는 노드 *Workshop-Proceedings-Reference*를 자식 노드로 갖는다.

그 다음으로, 그림 2(a)의 선택 연산자 노드가 부모 연산자 노드를 갖고 있으며, 이 부모 연산자 노드와 이제까지 통합된 연산자 집합의 깊이가 가장 얕은 연산자 노드의 종류가 순서 연산자 노드로 같으므로, 이 두 순서 연산자 노드를 표 2의 통합 규칙을 이용해 통합한다. 이 때 통합된 연산자 집합은 노드 *Book-Section-Reference*, 노드 *Book-Reference*, 선택 연산자 노드, 노드 *Conference-Proceedings-Reference*, 노드 *Article-Reference*, 노드 *Thesis-Reference*, 그리고 노드 *Technical-Report-Reference*들을 자식 노드들로 갖는 순서 연산자 노드가 된다. 이 통합된 연산자 집합 내의 선택 연산자 노드는 노드 *Workshop-Proceedings-Reference*를 자식 노드로 갖는다. 두 연산자 집합의 깊이가 가장 깊은 연산자 노드들이 통합되었으므로, 이 연산자 집합이 두 연산자 집합의 통합된 연산자 집합이 된다.

Case 3 : 하나의 연산자 집합은 연산자 노드를 갖지만 다른 연산자 집합은 연산자 노드를 갖지 않을 경우

하나의 연산자 집합은 연산자 노드를 갖지만 다른 연산자 집합은 연산자 노드를 갖지 않을 경우에는 표 3의 연산자 통합 규칙III을 이용해 통합한다. 만약 연산자 노드를 갖는 연산자 집합이 하나 이상의 연산

표 3. 연산자 통합 규칙III

깊이가 가장 얕은 소스 연산자 노드	통합 조건	통합된 연산자 노드
all	타겟 연산자 집합의 자식 노드 (애트리뷰트)의 빈도수가 1이하인 경우	all
	위의 통합 조건을 만족하지 않고, 소스 연산자 집합의 자식 노드들이 모두 단말 노드일 경우	존재하지 않음
	위의 통합 조건들을 모두 만족하지 않는 경우	통합되지 않음
choice	소스 연산자 집합의 자식 노드들 중 타겟 연산자 집합의 자식 노드들과 유의어 및 상하위어 관계를 갖는 노드가 둘 이상 없는 경우	choice
	위의 통합 조건을 만족하지 않고, 소스 연산자 집합의 자식 노드들이 모두 단말 노드이고 빈도수가 1이하인 경우	all
	위의 통합 조건들을 모두 만족하지 않고, 소스 연산자 집합의 자식 노드들이 모두 단말 노드인 경우	존재하지 않음
	위의 통합 조건들을 만족하지 않는 경우	통합되지 않음
sequence	소스 연산자 집합의 자식 노드들 중 타겟 연산자 집합의 자식 노드들과 유의어 및 상하위어 관계를 갖는 노드가 둘 이상 없는 경우	sequence
	위의 통합 조건을 만족하지 않고, 소스 연산자 집합의 자식 노드들이 모두 단말 노드이고 빈도수가 1이하인 경우	all
	위의 통합 조건들을 모두 만족하지 않고, 소스 연산자 집합의 자식 노드들이 모두 단말 노드인 경우	존재하지 않음
	위의 통합 조건들을 모두 만족하지 않는 경우	통합되지 않음

자 노드로 이루어져 있다면, 가장 깊이가 얕은 연산자 노드 하나만이 통합 대상 연산자 노드가 된다. 본 논문은 하나의 연산자 집합은 연산자 노드를 갖고, 다른 연산자 집합은 연산자 노드를 갖지 않을 경우의 정교한 연산자 통합 규칙을 제안한다.

그림 2(a)의 노드 *Non-Profit-Organization*의 자식 연산자 집합과 그림 2(b)의 노드 *Non-Profit-Organization*의 자식 연산자 집합을 통합하는 예는 다음과 같다. 그림 2(b)의 연산자 집합은 연산자 노드를 갖지만 그림 2(a)의 연산자 집합은 연산자 노드를 갖지 않기 때문에 두 연산자 집합을 표 3의 연산자 통합 규칙III을 이용하여 통합한다. 그림 2(b)의 연산자 집합의 유일한 연산자 노드는 선택 연산자 노드이고, 그림 2(a)의 연산자 집합의 자식 노드들 중 그림 2(b)의 연산자 집합의 자식 노드들과 유의어 관계를 갖는 노드가 두개 존재한다. 그리고 그림 2(b)의 연산자 집합의 자식 노드들은 모두 단말 노드이고 빈도수는 생략되어 있으므로 최대값과 최소값 모두 1이다. 그러므로 통합된 연산자 집합은 all 연산자 노드가 된다.

3.2.2 유의어 노드 통합

유의어 노드 통합 방법은 통합될 노드들의 종류에 따라 다음의 세 가지로 분류된다. 첫째, 단말 노드 간 통합에서 통합된 노드는 단말 노드로 두 노드 간 이름, 데이터 타입, 빈도 지시자 및 enumeration의 충돌을 해결하여 생성된다.

두 노드 간 이름이 서로 다른 경우, WordNet과 도메인 온톨로지를 이용해 광의의 이름을 통합 노드의 이름으로 선택한다. 두 노드 간 데이터 타입이 서로 다른 경우, 정보 손실 없이 변환을 통해 다른 타입을 포함할 수 있는 타입을 선택한다. 두 타입 간 변환이 불가능할 경우, 통합 노드의 데이터 타입은 문자열(string)이 된다. 두 노드 간 빈도 지시지가 다를 경우, 통합 노드는 두 최소값 중 작은 값과 두 최대값 중 큰 값을 각각 최소값과 최대값으로 갖는다.

단말 노드와 비 단말 노드 간 통합에서 통합된 노드는 섞인 내용 모델의 비 단말 노드로 생성된다. 통합된 노드는 비 단말 노드의 하위 구조를 포함하며 단말 노드 간 통합과 같은 방법으로 노드 간 이름 및 빈도수 충돌을 해결한다.

두 노드가 모두 비 단말 노드일 경우 통합된 노드는 비 단말 노드로써, 두 노드의 하위 구조를 포함한다. 비 단말 노드 간 통합은 단말 노드 간 통합과 같은 방법으로 노드의 이름 및 빈도수 속성 및 enumeration의 충돌을 해결한다.

3.2.3 상하위어 노드 통합

두 스키마 간 상하위어 노드를 통합할 경우 하위어 엘리먼트 (또는 애트리뷰트) 노드를 상위어 노드의 가장 가까운 하위 엘리먼트 (또는 애트리뷰트) 노드로 통합한다. 만약 하위어 노드가 상위어 노드의 가장 가까운 하위 엘리먼트 또는 애트리뷰트 노드 중 하나와 유의어 관계에 있다면, 이 두 노드들은 유의어 노드 통합 방법에 의해 통합된다.

두 스키마 간 상하위어 노드를 통합하는 예는 다음과 같다. 그림 2(a)의 노드 *Serial-Publication*은 그림 2(b)의 노드 *Journal*의 상위어로 도메인 온톨로지 그림 2(c) 상에서 정의되어 있다. 그러므로 하위어 노드 *Journal*은 상위어 노드 *Serial-Publication*의 가장 가까운 하위 엘리먼트로 통합된다. 또한 하위어 노드 *Journal*은 상위어 노드 *Serial-Publication*의 가장 가까운 하위 엘리먼트 노드 *Journal*의 유의어이므로, 두 노드는 유의어 노드 통합 방법에 의해 하나의 노드로 통합 된다.

3.3 최적화

이전 단계에서 생성된 스키마를 그림 5와 같이 몇 가지 자동화된 휴리스틱을 통해 최적화하여 최종적인 통합 스키마를 생성한다. 그림 5의 규칙 1~3번은 본 논문에서 제안하는 연산자 최적화 규칙이다.

규칙 1. 현재 연산자 노드의 종류가 현재 연산자 노드의 자식 연산자 노드의 종류와 같다면, 현재 연산자 노드의 자식 연산자 노드를 제거한다. 현재 연산자 노드의 자식 연산자 노드의 하위 구조는 현재 연산자 노드의 하위 구조가 된다.

규칙 2. 현재 연산자 노드의 자식 연산자 노드는 형제 노드를 갖지 않고, 현재 연산자 노드와 현재 연산자 노드의 자식 연산자 노드의 종류가 다르면, 현재 연산자 노드를 제거한다. 현재 연산자 노드의 하위 구조는 현재 연산자 노드의 부모 노드의 하위 구조가 된다.

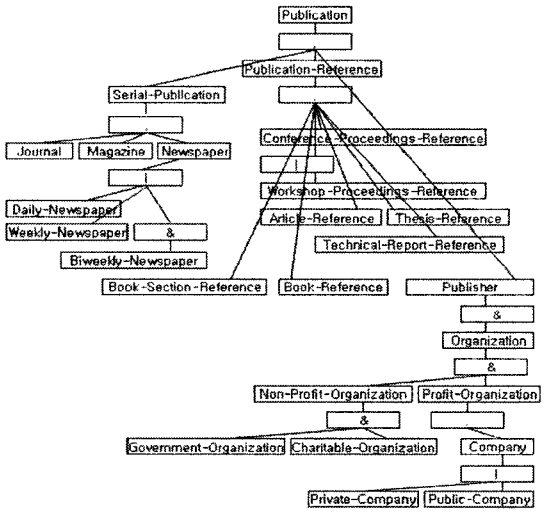
규칙 3. 현재 연산자 노드의 자식 연산자 노드가 자식 노드를 하나만 갖고 있다면, 현재 연산자 노드의 자식 연산자 노드를 삭제한다. 현재 연산자 노드의 자식 연산자 노드의 하위 구조는 현재 연산자 노드의 하위 구조가 된다.

규칙 4. 통합 스키마 내의 사이클을 제거한다.

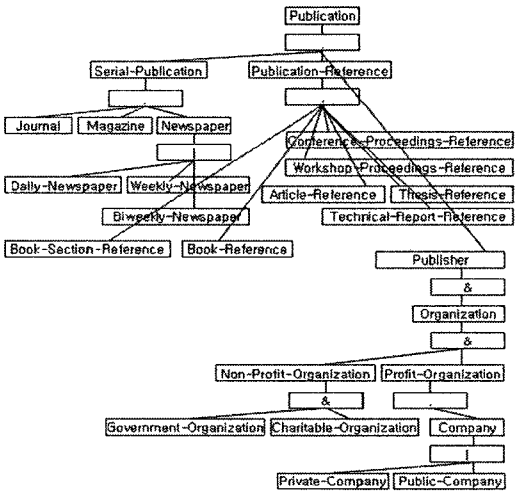
규칙 5. 중복되는 빈도 지시자를 제거한다.

그림 5. 스키마 최적화 규칙

그림 5의 규칙 1은 현재 연산자 노드와 현재 연산자 노드의 자식 연산자 노드의 종류가 같을 경우, 자식 연산자는 전체 스키마에 어떤 의미도 추가하지 않기 때문에 성립한다.



(a) 최적화 규칙 적용 전



(b) 최적화 규칙 적용 후

그림 6. 통합 스키마 트리의 예

표 4. 실험 데이터

도메인	스키마수	스키마당 평균 엘리먼트 및 애트리뷰트 노드수	스키마당 평균 단말노드수	도메인 온톨로지 출처
Publication	5	23	16	http://www.schemaweb.info/
Movie	5	35	29	http://www.schemaweb.info/
Science	5	33	22	http://www.schemaweb.info/

또한, 규칙 2와 3은 다음과 같은 이유 때문에 성립한다. 만약 순서 연산자의 자식 노드가 하나 밖에 없는 경우, 순서 연산자가 갖는 자식 노드들 간의 순서의 의미가 상실된다. 또한, 선택 연산자의 자식 노드가 하나 밖에 없는 경우, 선택 연산자가 갖는 자식 노드들 중 하나만이 출현할 수 있다는 의미가 상실된다. 위의 두 경우에, 자식 노드를 하나만 갖는 연산자의 부모 노드 또는 자식 노드가 연산자 노드라면, 자식 노드를 하나만 갖는 연산자 노드는 전체 스키마에 일반적인 연산자 노드로써의 의미를 추가하지도 않고, 종류별 연산자 노드가 갖는 특수한 의미를 추가하지도 않기 때문에 삭제되어도 전체 스키마의 의미는 변하지 않는다.

그림 2의 두 스키마의 최적화 규칙 적용 전과 후의 통합 스키마는 그림 6과 같다. 그림 6(a)의 내의 노드 *Publication-Reference*의 자식 연산자 집합을 최적화 하는 예는 다음과 같다. 노드 *Publication-Reference*의 자식 연산자 집합은 선택 연산자 노드를 자식으로 갖는 순서 연산자 노드이다. 그리고 이 선택 연산자 노드는 노드 *Workshop-Proceedings-Reference*를 유일한 자식 노드로 갖는다. 그러므로 이 선택 연산자를 제거하고 노드 *Workshop-Proceedings-Reference*를 그 선택 연산자의 부모 노드였던 순서 연산자의 자식 노드로 최적화한다.

4. 실험 결과

본 절에서는 제안된 방법의 성능을 평가하기 위해 수행한 실험 결과를 기술한다. 실험 데이터는 표 4와 같이 Publication, Movie, 그리고 Science의 3개 도메인에서 사용 중인 15개의 XML 스키마와 해당 도메인 온톨로지로 구성된다.

4.1 평가 기준

알고리즘의 성능 분석을 위한 기준은 표 5와 같이 Meo 등 [4] 의 기준인 정확률, 재현율, *F*-Measure, Overall, 그리고 *RSS* (Relative Schema Size)와 본 논문에서 제안하는 *RCN* (Relative Compositor Number)을 사용하였다. 정확률과 재현율은 정보 추출 기술의 대표적인 기준이고 [19], *RSS*와 *RCN*은 복잡도의 기준이다.

특히 본 논문에서는 정확률과 재현율을 계산 시, 유의어 및 상하위어 간의 매칭을 고려한다. Overall은 매칭 이후의 노력을 평가하기 위한 기준이며 정확률, 재현율, *F*-Measure, 그리고 Overall은 값이 클수록 알고리즘이 우수하다는 것을 의미한다.

*RSS*는 소스 및 타겟 스키마의 *ConstructSet*의 크기에 대한 통합 스키마의 *ConstructSet*의 크기의 비

표 5. 성능 분석을 위한 기준

기 준	정 의
정 확 률	$\frac{\lvert A \cap C \rvert}{\lvert C \rvert}$, A는 전문가들이 추출한 통합 스키마와 입력 스키마들 사이에 존재하는 매칭들의 세트. C는 제안된 알고리즘을 바탕으로 구현한 시스템이 추출한 통합 스키마와 입력 스키마들 사이에 존재하는 매칭들의 세트. A∩C는 전문가들이 추출한 매칭들과 제안된 알고리즘을 바탕으로 구현한 시스템이 추출한 매칭들 중 공통된 매칭들의 세트
재 현 율	$\frac{\lvert A \cap C \rvert}{\lvert A \rvert}$
F-Measure	$2^* \frac{\text{정확률}^* \text{재현율}}{\text{정확률} + \text{재현율}}$
Overall	$\text{재현율}^*(2 - \frac{1}{\text{정확률}})$
RSS (Relative Schema Size)	$\frac{\lvert ConstructSet(S_G) \rvert}{\lvert ConstructSet(S_1) \rvert + \lvert ConstructSet(S_2) \rvert}$, *ConstructSet*은 해당 스키마의 모든 엘리먼트 및 애트리뷰트들의 세트
RCN (Relative Compositor Number)	$\frac{\lvert CompositorSet(S_G) \rvert}{\lvert CompositorSet(S_1) \rvert + \lvert CompositorSet(S_2) \rvert}$ *CompositorSet*은 해당 스키마의 모든 연산자 노드들의 세트

율을 의미한다. *RCN*은 본 논문에서 제안하는 연산자 복잡도에 대한 기준으로, 소스 및 타겟 스키마의 *CompositorSet*의 크기에 대한 통합 스키마의 *CompositorSet*의 크기의 비율을 의미한다. *RSS*는 0.5와 1 사이의 값을 갖고, *RCN*은 0과 ∞사이의 값을 갖는다. *RSS*와 *RCN*은 값이 작을수록 통합 스키마가 덜 복잡하다는 것을 의미한다.

4.2 성능 분석

본 논문은 제안된 알고리즘을 바탕으로 구현한 시스템의 성능 평가 결과를 도메인별로 정리하였다. 각 도메인의 입력 스키마 수에 대한 모든 조합을 이루는 스키마 쌍에 대해 통합을 수행한 후, 평균값을 계산하였다.

표 6~8은 제안된 방법으로 실험한 결과와, 제안된 방법에서 한 가지 조건만을 달리한 방법으로 실험한 결과들을 비교한 것이고, 그림 7~9는 이를 그래프로 나타낸 것이다. 다시 말하면, 표 6~8과 그림 7~9는 제안된 방법으로 실험한 결과와 도메인 온톨로지 또는 구조 가중치 또는 어휘 간 상하위어 유추 조건을 이용하지 않은 방법으로 실험한 결과를 각각 표와 그래프를 이용해 비교한 것이다. 어휘 간 상하위어 유추 조건을 이용하지 않은 방법은 두 어휘 간 상하위어 관계가 WordNet 또는 도메인 온톨로지에 직접적으로 기술되어 있어야만 두 어휘 간 상하위어 관계를 추출한다.

표 6과 그림 7을 보면 Publication 스키마의 경우, 제안된 방법은 어휘 간 상하위어 유추 조건을 이용하지 않은 방법보다 높은 *RSS*를 보였다. 그 이유는 제안된 방법은 어휘 간 상하위어 유추 조건을 이용하지 않은 방법보다 상하위어 매칭을 많이 찾았으나, 상위

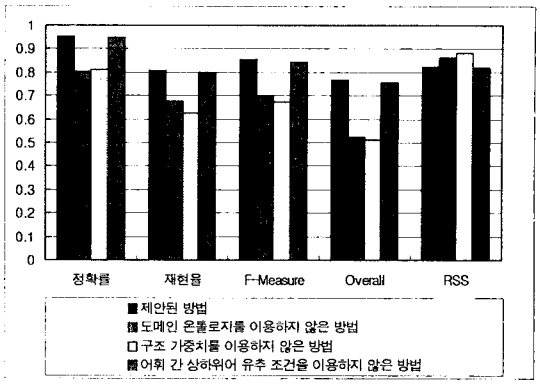


그림 7. Publication 스키마의 실험 결과

어 노드의 가장 가까운 하위 엘리먼트 노드들과 하위어 노드 간에 유의어 관계가 존재하지 않았으므로 통합 스키마의 노드 수가 감소하지 않았기 때문이다. 그리고 어휘 간 상하위어 유추 조건을 이용하지 않은 방법은 제안된 방법이 찾지 않은, 전문가가 찾은 매칭과 공통되지 않은 유의어 관계 매칭을 하나 찾았기 때문이다.

표 7과 그림 8을 보면 Movie 스키마의 경우, 제안된 방법은 도메인 온톨로지를 이용하지 않은 방법과 같은 정확률을 보였다. 그 이유는 Movie 도메인 온톨로지에 있는 정보가 도메인 온톨로지를 이용하지 않은 방법이 추출한 잘못된 매칭을 수정하기에 불충분했기 때문이다.

표 8과 그림 9를 보면 Science 스키마의 경우, 제안된 방법, 도메인 온톨로지를 이용하지 않은 방법, 그리고 어휘 간 상하위어 유추 조건을 이용하지 않은 방법은 같은 정확률을 보였다. 그 이유는 위에서 설명한 Movie 스키마에서 제안된 방법과 도메인 온톨로지를 이용하지 않은 방법이 같은 정확률을 보인

표 6. Publication 스키마의 실험 결과

실험 방법 / 통합 결과	제안된 방법	도메인 온톨로지를 이용 하지 않은 방법	구조 가중치를 이용 하지 않은 방법	어휘 간 상하위어 유추 조건을 이용 하지 않은 방법
정확률	0.952	0.804	0.811	0.947
재현율	0.808	0.679	0.626	0.798
F-Measure	0.853	0.696	0.674	0.843
Overall	0.766	0.525	0.512	0.756
RSS	0.8228	0.8605	0.8801	0.8207

표 7. Movie 스키마의 실험 결과

실험 방법 / 통합 결과	제안된 방법	도메인 온톨로지를 이용 하지 않은 방법	구조 가중치를 이용 하지 않은 방법	어휘 간 상하위어 유추 조건을 이용 하지 않은 방법
정확률	0.984	0.984	0.913	0.982
재현율	0.812	0.553	0.548	0.766
F-Measure	0.815	0.597	0.577	0.789
Overall	0.794	0.535	0.514	0.748
RSS	0.8548	0.9084	0.9121	0.8663

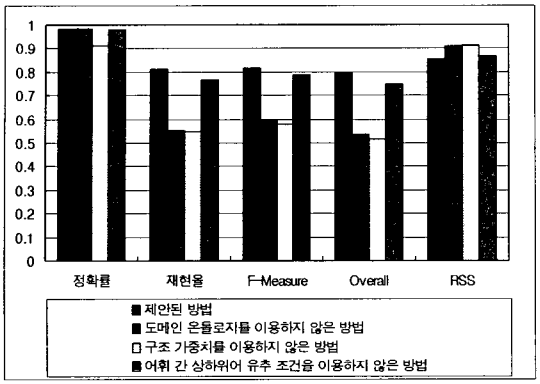


그림 8. Movie 스키마의 실험 결과

표 8. Science 스키마의 실험 결과

실험 방법 / 통합 결과	제안된 방법	도메인 온톨로지를 이용 하지 않은 방법	구조 가중치를 이용 하지 않은 방법	어휘 간 상하위어 유추 조건을 이용 하지 않은 방법
정확률	0.991	0.991	0.874	0.991
재현율	0.870	0.709	0.732	0.852
F-Measure	0.912	0.794	0.784	0.902
Overall	0.860	0.699	0.620	0.842
RSS	0.8418	0.8647	0.8664	0.8418

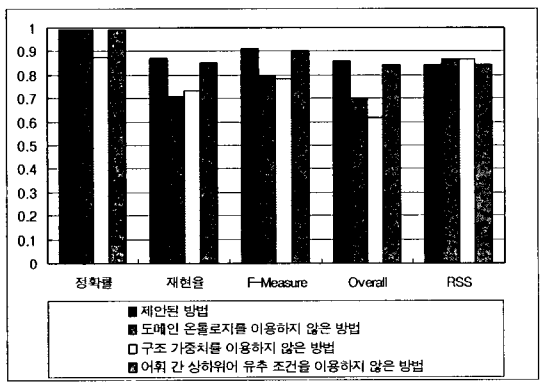


그림 9. Science 스키마의 실험 결과

이유와 같다.

또한 Science 스키마의 경우, 제안된 방법은 어휘 간 상하위어 유추 조건을 이용하지 않은 방법과 같은 *RSS*를 보였다. 그 이유는 제안된 방법은 어휘 간 상하위어 유추 조건을 이용하지 않은 방법보다 상하위어 매칭을 많이 찾았으나, 상위어 노드의 가장 가까운 하위 엘리먼트 노드들과 하위어 노드 간에 유의어 관계가 존재하지 않았으므로 통합 스키마의 노드 수가 감소하지 않았기 때문이다.

표 9. 다양한 도메인의 스키마의 RCN

실험 방법 / 도메인	제안된 방법	연산자 최적화 적용 하지 않음
Publication	0.7750	0.8102
Movie	0.7517	0.7979
Science	0.8176	0.8455
평균	0.78143	0.81787

또한 표 9는 제안된 방법과 연산자 최적화 규칙을 적용하지 않은 경우에 대하여 *RCN*를 비교한 것이다. 한편 실험에 사용된 임계값은 사전에 실험을 통해 결정하였다. 어휘 간 유사도 임계값 TH_{sim}은 0.7로 설정하였고, 토큰 간 유사도 임계값 $TH_{tokensim}$은 0.8로 설정하였다.

제안된 방법은 도메인 온톨로지 또는 구조 가중치 또는 어휘 간 상하위어 유추 조건을 이용하지 않은 방법보다 개선된 정확률, 재현율, *F*-Measure, Overall 그리고 *RSS*를 보였다. 또한 제안된 방법은 연산자 최적화 규칙을 이용하지 않은 방법보다 낮은 *RCN*을 보였다.

이와 같이, 도메인 온톨로지를 이용한 제안된 방법이 개선된 정확률, 재현율 *F*-Measure, Overall 그리고 *RSS*를 보인 이유는 도메인 온톨로지에 있는 정보가 전문가가 추출한 어휘 간 유의어 및 상하위어 관계 정보와 매우 유사했기 때문이다. 예를 들어, Publication 도메인에서 제안된 방법은 도메인 온톨로지를 이용하여, Publication 도메인의 첫 번째 스키마의 어휘 *Serial-Publication*이 Publication 도메인의 두 번째 스키마의 어휘 *Journal*의 상위어라는 정보를 추출하여, 이 두 어휘를 갖는 노드를 상하위어 노드 통합 방법으로 통합한다. 또한, 제안된 방법은 Publication 도메인의 첫 번째 스키마의 노드 *Serial-Publication*의 가장 가까운 하위 엘리먼트 노드 *Journal*과, Publication 도메인의 두 번째 스키마의 노드 *Journal*이 유의어 관계를 가지므로 유의어 노드 통합 방법으로 통합한다. 그러나 도메인 온톨로지를 이용하지 않은 방법은 이 상하위어 관계를 추출하지 못했기 때문에 제안된 방법보다 낮은 재현율, *F*-Measure, Overall과 높은 *RSS*를 보였다.

그리고 제안된 방법은 구조 가중치를 이용하여 보다 정확하게 유의어 및 상하위어 관계를 추출할 수

있었기 때문에 개선된 정확률, 재현율 *F*-Measure, Overall 그리고 *RSS*를 보였다. 예를 들어, Movie 도메인에서 제안된 방법은 구조 가중치를 이용하여, Movie 도메인의 첫 번째 스키마의 어휘 *Genre*와 Movie 도메인의 두 번째 스키마의 노드 Movie의 자식 노드의 어휘 *MovieGenre* 간의 유의어 관계를 추출한다. 그러므로 두 노드는 유의어 노드 통합 방법으로 통합된다. 그러나 구조 가중치를 이용하지 않은 방법은 이 두 어휘 간 유의어 관계를 추출하지 못했기 때문에 제안된 방법보다 낮은 재현율, *F*-Measure, Overall과 높은 *RSS*를 보였다.

또한, 제안된 방법은 어휘 간 상하위어 유추 조건을 이용하여 보다 정확하게 상하위어 관계를 추출할 수 있었기 때문에 개선된 정확률, 재현율 *F*-Measure, Overall 그리고 *RSS*를 보였다. 그 예로, Movie 도메인에서 제안된 방법은 어휘 간 상하위어 유추 조건을 이용하여 Movie 도메인의 첫 번째 스키마의 어휘 *Festival*이 Movie 도메인의 두 번째 스키마의 어휘 *CannesFilmFestival*의 상위어라는 정보를 추출하여 두 노드를 상위어 노드 통합 방법으로 통합한다. 그러나 어휘 간 상하위어 유추 조건을 이용하지 않은 방법은 두 어휘 간의 상하위어 관계를 추출하지 못하므로 제안된 방법보다 낮은 재현율, *F*-Measure, Overall을 보였다.

또한 제안된 방법은 연산자 최적화 규칙을 이용하여 의미적으로 중복되는 연산자 노드를 삭제함으로써, 기존의 연산자가 갖는 의미를 유지하면서 개선된 *RCN*을 보였다. 예를 들어, Science 도메인에서 연산자 최적화 규칙을 이용하지 않은 방법으로 Science 도메인의 첫 번째 스키마와 Science 도메인의 세 번째 스키마를 통합한 스키마에서 노드 *Scientist*의 자식 연산자 집합은 선택 연산자 노드와 엘리먼트 노드들을 자식 노드로 갖는 순서 연산자이고, 그 선택 연산자 노드는 단 하나의 자식 노드 *Geophysicist*를 갖는다. 그러나 그림 5의 규칙 3인 연산자 최적화 규칙을 이용한 제안된 방법으로 통합한 스키마에서 노드 *Scientist*는 자식 노드로 순서 연산자 노드를 갖고, 그 순서 연산자는 노드 *Geophysicist*와 다른 엘리먼트 노드들을 자식 노드로 갖는다.

4.3 기존 연구와의 비교

스키마 통합에 관한 기존 연구와의 비교는 표 10과 같다. 노드 이름 토큰별 중요도 추출은 제안된 방법의 구조 가중치처럼 각 노드 이름 토큰이 갖는 중요도를 추출하여 스키마 통합에 활용하는지 여부를 가리킨다.

표 10. 스키마 통합에 관한 기존 연구와의 비교

	Huma 등 [2]	Mello와 Heuser [3]	Meo 등 [4]	Cruz 등 [5]	제안된 알고리즘
개념 모델	XML 온톨로지	ORM /NIAM	*XS -Graph*	RDF 온톨로지	XML schema
온톨로지 이용	○	×	×	○	○
도메인 온톨로지에서 정보추출	○	×	×	×	○
노드 이름 토큰별 중요도 추출	×	×	×	×	○
연산자 최적화	×	×	×	×	○
이름 통합	○	○	○	○	○
타입 통합	○	×	○	×	○
노드 순서 고려	×	×	×	×	○
구조 간 통합	○	○	○	○	○
최적화	×	×	○	×	○
자동화	×	×	○	×	○

5. 결론 및 향후연구

본 논문은 동일한 도메인에 속하는 XML 스키마 통합 방법을 제안한다. 제안된 방법은 다음과 같은 세 단계로 구성된다. 첫 번째로, 제안된 방법은 유의어 및 상하위어 관계 정보를 일반적인 사전과 도메인 온톨로지에서 찾음으로써 일반적인 정보 뿐 아니라 해당 도메인의 특수한 정보를 활용한다. 그리고 제안된 방법은 구조 가중치와 어휘 간 상하위어 유추 조건을 이용하여 두 어휘 간 유의어 및 상하위어 관계를 보다 정확하게 검사한다. 두 번째로, 제안된 방법은 정교한 연산자 노드 통합 방법을 이용하여 연산자 노드를 통합하고, 유의어 및 상하위어 노드들을 통합한다. 마지막으로 연산자 최적화 규칙을 적용하여 의미적으로 중복된 노드를 삭제한다. 제안된 방법은 XML 스키마들을 다른 개념 모델들로 변환시키지 않으므로 변환을 필요로 하는 방법보다 입력 스키마들

의 의미를 더 많이 보존할 수 있다. 또한 제안된 방법의 모든 과정은 사용자의 개입을 필요로 하지 않는다.

실험 결과, 도메인 온톨로지 또는 구조 가중치 또는 어휘 간 상하위어 유추 조건을 이용한 경우, 정확률, 재현율, F-Measure, Overall, 그리고 복잡도가 개선되었음을 확인할 수 있었다. 또한 제안된 연산자 최적화 규칙을 이용함으로써, 입력 스키마의 연산자가 지닌 의미를 잃지 않으면서 입력 스키마들의 연산자 노드 수에 대한 통합 스키마의 연산자 노드 수의 비율을 감소시켰다.

제안된 방법은 기존 XML 스키마 통합 방법과 마찬가지로 전문가들이 추출하는 수준의 의미적으로 정확한 통합 스키마를 생성하지 못한다. 이러한 한계점을 극복하기 위해서는 소스 스키마들이 갖는 구조와 노드들의 의미를 보다 정확하게 추출할 수 있어야 할 것이다. 또한 XML 스키마 통합 방법들의 시간 복잡도를 비교 분석하고, 이를 감소시킬 수 있는 알고리즘의 개발도 필요하다.

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] Z. Huma, M.J. Rehman, and N. Iftikhar, "An Ontology-Based Framework for Semi-Automatic Schema Integration," *Journal of Computer Science and Technology*, Vol.20, No.6, pp. 788-796, 2005.

[3] R.D.S. Mello and C.A. Heuser, "BInXS: A Process for Integration of XML Schemata," *Proc. 17th Int'l Conf. CAiSE* (LNCS 3520), pp. 151-166, 2005.

[4] P.D. Meo, G. Quattrone, G. Terracina, and D. Ursino, "Integration of XML Schemas at Various Severity Levels," *Information Systems*, Vol.31, No.6, pp. 397-434, 2006.

[5] I.F. Cruz, H. Xiao, and F. Hsu, "An Ontology-Based Framework for XML Semantic Integration," *Proc. Int'l Defence Exhibition & Seminar*, pp. 217-226, 2004.

[6] E. Jeong and C.-H. Hsu, "View Inference for Heterogeneous XML Information Integration," *Journal of Intelligent Information Systems*, Vol.20, No.1, pp. 81-99, 2003.

[7] X. Yang, M.L. Lee, and T.W. Ling, "Resolving Structural Conflicts in the Integration of XML Schemas: A Semantic Approach," *Proc. 22nd Int'l Conf. on Conceptual Modeling* (LNCS 2813), pp. 520-533, 2003.

[8] S. Castano, A. Ferrara, G.S.K. Ottathycal, and V.D. Antonellis, "A Disciplined Approach for the Integration of Heterogeneous XML Datasources," *Proc. Int'l Workshop Web Semantics*, pp. 103-110, 2002.

[9] K.-H. Lee, M.-H. Kim, K.-C. Lee, B.-S. Kim, and M.-Y. Lee, "Conflict Classification and Resolution in Heterogeneous Information Integration Based on XML Schema," *Proc. IEEE Conf. Computers, Communications, Control and Power Engineering*, pp. 93-96, 2002.

[10] R.D.S. Mello, S. Castano, and C.A. Heuser, "A Method for the Unification of XML Schemata," *Information & Software Technology*, Vol.44, No.4, pp. 241-249, 2002.

[11] K. Passi, L. Lane, S.K. Madria, B.C. Sakamuri, M.K. Mohania, and S.S. Bhowmick, "A Model for XML Schema Integration," *Proc. Third Int'l Conf. on E-Commerce and Web Technologies*, pp. 193-202, 2002.

[12] Y.-F. Zhang and W.-Y. Liu, "Semantic Integration of XML Schema," *Proc. Int'l Conf. Machine Learning and Cybernetics*, pp. 1058-1061, 2001.

[13] N.F. Noy and M.A. Musen. "PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment," *Proc. 17th National Conf. on Artificial Intelligence and 12th Conf. on Innovative Applications of Artificial Intelligence*, pp. 450-455, 2000.

[14] S. Castano, V.D. Antonellis, and S.D.C.D. Vimercati, "An XML-Based Interorganizatio-

nal Knowledge Mediation Scheme to Support B2B Solutions," *Proc. 9th IFIP 2.6 Working Conf. on Database Semantics*, pp. 121-135, 2001.

[15] T.-W. Rhim and K.-H. Lee, "Clustering of XML Schemas for Information Integration," *Journal of Computer Information Systems*, Vol.46, No.2, pp. 3-13, 2006.

[16] G.A. Miller, "WordNet: A Lexical Database for English," *Communications of the ACM*, Vol.38, No.11, pp. 39-41, 1995.

[17] P.D. Meo, G. Quattrone, G. Terracina, and D. Ursion, "Extraction of Synonymies, Hyponymies, Overlappings and Homonymies from XML Schemas at Various Severity Levels," *Proc. Int'l Database Engineering and Applications Symposium*, pp. 389-394, 2004.

[18] C.-H. Moh, E.-P. Lim, and W.-K. Ng, "Re-Engineering Structures from Web Documents," *Proc. 5th ACM Conf. Digital Libraries*, pp. 67-76, 2000.

[19] C.J.V. Rijsbergen, *Information Retrieval*, University of Glasgow, Scotland, U.K., 1979.

강 혜 란

2003년 8월 상명대학교 전자계산학과 졸업(학사)
2007년 2월 연세대학교 컴퓨터과학과 졸업(석사)
2007년 3월~현재 연세대학교 컴퓨터과학과 박사과정

관심분야 : XML Schema, Schema Integration, Ontology Integration

이 경 호

1995년 2월 연세대학교 전산과학과 졸업(학사)
1997년 2월 연세대학교 컴퓨터과학과 졸업(석사)
2001년 2월 연세대학교 컴퓨터과학과 졸업(박사)
2001년 3월 National Institute of Standard and Technology (NIST) 객원연구원
2002년 9월~현재 연세대학교 컴퓨터과학과 부교수
관심분야 : Internet Computing, Service-Oriented Computing, Multimedia Document Engineering