OWL 인식 관계형 모델에서 SQL 기반의 시맨틱 질의 처리

(SQL-based Semantic Query Processing in the OWL-aware Relational Model)

김 학 수 † 손 진 현 ††
(Hak Soo Kim) (Jin Hyun Son)

요 약 온톨로지 기반의 애플리케이션에 대한 사용이 증가함에 따라 시맨틱 정보를 효율적으로 저장 및 처리하는 것이 중요하게 다루어지고 있다. 비록 몇몇의 관련된 시스템들이 개발되어왔지만, 이들은 시맨틱 데이타의 크기, 시맨틱 질의 처리의 성능, 시맨틱 데이타 유지관리의 관점에서 몇몇 제한을 가지고 있다. 본 논문에서는 온톨로지 관리 시스템을 위한 OWL 인지 관계형 모델을 제안하고 이를 이용하는 SQL 기반의 시맨틱 질의 처리 메커니즘을 제안한다. 또한 질의 처리 성능에 대한 검증을 위해 Sesame와 비교를 통해서 좀 더 효율적임을 보여준다.

키워드 : 시맨틱 정보, 시맨틱 질의, OWL, SQL, 데이타베이스

Abstract According to the widespread use of ontology-based applications, it is critical to efficiently store and process semantic information. Even though several related systems have been developed, they have some limitations in perspectives of the volume of target semantic data, the performance of semantic query processing, and the semantic data maintenance. In this paper we propose the OWL-aware relational model for the ontology management system and SQL-based semantic query processing mechanism. Also, to verify the query processing performance, we show that the proposed query processing mechanism is more efficient than sesame.

Key words : Semantic information, Semantic query, OWL, SQL, Database

1. 서 론

오늘날 온톨로지의 개념은 시맨틱 웹 애플리케이션, 홈 네트워킹, 텔레매틱스, 지능형 로봇과 같은 효율적인 상황(Context) 정보 처리를 요구하는 분야에서 널리 사용되고 있다. 본 논문에서 논의하는 주제는 W3C에서 제안한 웹 기반의 온톨로지 언어인 RDF/S[1,2]와 OWL [3]에 근간을 두고 있다. RDF는 웹에 있는 임의의 자원을 트리플(주어, 서술어, 목적어)의 형태로 표현하는 프레임워크로서 제안되었다. 트리플로 표현된 RDF 문서를 기계가 쉽게 처리할 수 있도록 RDF 그래프로 표현할 수 있는데 주어와 목적어는 그래프 노드로, 서술어는 에지(Edge)로 표현될 수 있다. 결국, 웹 상에 존재하는 모든 자원들은 RDF로 기술된 그래프 모델로 표현할 수 있다. 한편, RDF는 단순히 웹 상에 존재하는 자원을 표현하는 언어로 스키마 정보, 자원 사이의 관계 정보, 제약 정보 등에 대한 표현이 미약하다. 이에 W3C에서는 좀 더 표현력이 풍부한 RDF Schema와 OWL을 추가적으로 제정하였다. 본 논문은 DL(Description Logic)을 기반으로 한 OWL에 초점을 둔다.

상황 인지(Context-awareness) 시스템들은 기본적으로 시맨틱 웹 기술[4]에 의해 지원되기 때문에, 상황 이

· 이 연구(논문)는 산업자원부 지원으로 수행하는 21세기 프론티어 연구개발사업(인간기능 생활지원 지능로봇 기술개발사업)의 일환으로 수행되었습니다.
· 이 논문은 2007년도 정부(과학기술부)의 재원으로 한국과학재단의 지원을 받아 수행된 연구임(No. R01-2007-000-20135-0).

† 학생회원 : 한양대학교 컴퓨터공학과
hagsoo@cse.hanyang.ac.kr
†† 종신회원 : 한양대학교 컴퓨터공학과 교수
jhson@cse.hanyang.ac.kr

논문접수 : 2007년 4월 27일
심사완료 : 2007년 11월 14일

해 서비스들을 제공하길 원하는 대부분의 애플리케이션들은 웹 관련 국제 표준들을 채택하고 있다. 국제 표준으로는 웹 온톨로지 언어인 OWL, 소프트웨어 모듈들 사이의 상호작용을 위한 웹 서비스가 표준으로 제공되고 있다. 이러한 표준의 채택으로 많은 관련 애플리케이션들이 개발됨에 따라, OWL 기반의 온톨로지 문서 크기의 증가로 인한 효율적인 온톨로지 관리 시스템에 대한 연구가 활발히 진행되고 있다. 효율적인 상황 정보 처리를 위해, RDF/S를 기반으로 한 온톨로지 문서들을 저장하고 관리하는 방법은 [5]의 논문에서 처음으로 논의 되었다. 본 논문에서, 우리는 [5]의 논문과 비슷한 연구 방향으로서 OWL 문서들을 저장할 수 있는 OWL 인지 관계형 모델을 제안한다.

또한 온톨로지 데이타를 질의할 수 있는 시맨틱 질의 언어로 지금까지 약 20여 개가 제안되었으며, 시맨틱 질의 언어의 분류 체계는 그림 1과 같다. 이 중에서 현재 W3C에 의해 표준화 과정에 있는 시맨틱 질의 언어로는 RDQL과 SPARQL이 있는데 모두 SQL-유사 질의 언어이며 본 논문에서는 RDQL을 대상으로 하는 시맨틱 질의 처리 메커니즘에 초점을 둔다.

논문의 구성은 다음과 같다. 시맨틱 질의 언어인 RDQL에 대한 소개를 하고 Jena, Sesame의 RDQL 질의 처리 엔진에 대한 관련연구를 2장에서 소개한다. 3장에서는 본 논문에서 제안한 OWL 인지 관계형 모델을 기술하며 4장에서는 RDQL 질의 언어를 SQL로 변환하는 메커니즘을 소개한다. 마지막으로 5장에서는 성능 평가를, 6장에서 결론으로 논문을 마무리 짓는다.

2. 관련 연구 및 동기 부여

2.1 RDQL

RDQL은 W3C에서 표준으로 제정한 RDF를 위한 시맨틱 질의 언어로서 SQL과 유사한 구문을 제공한다.

표 1 RDQL 문법 요약

SELECT	결과 값으로 반환되어야 하는 변수를 기술한다.
FROM	질의 대상이 되는 문서를 URI로 지정한다.
WHERE	질의의 대상이 되는 트리플들을 기술한다. 여기에서 변수가 사용될 수 있다. 트리플들은 다음과 같은 트리플 패턴을 가질 수 있으며 다중 트리플 패턴을 기술할 수 있다. ({주어|?변수} {서술어|?변수} {목적어|?변수})
AND	불리언 연산을 표현한다.
USING	URI에 대한 접두사(Prefix)를 지정한다.

SELECT 절에서는 트리플 패턴에서 사용된 변수를 기술함으로써 변수의 결과 값을 SQL의 테이블과 같은 형태로 결과 값을 얻도록 하고, WHERE 절에서는 트리플 패턴을 기술하여 RDF 그래프 모델상에 있는 특정한 트리플들을 검색할 수 있게 한다. 표 1은 RDQL이 가지고 있는 문법을 간단하게 요약한 것이다. 이에 대한 RDQL 예제는 그림 2와 같다.

그림 2에서 보는 것처럼 RDF 문서 예제에 대한 두 개의 RDQL 질의 예를 보여주고 있다. 첫 번째 RDQL 질의는 RDF 문서상에서 "vcard-rdf:FN"의 값으로 "Matt Jones"인 주어를 찾는 예제이고 두 번째 예제는 "vcard-rdf:Family"의 값이 "Jones"인 "vcard-rdf:N"의 주어를 찾는 예제이다.

2.2 기존의 RDF/S 저장소

지금까지 개발된 두드러진 시맨틱 정보 관리 시스템은 Jena[6,7], Sesame[8], Parka[9], TAP[10] 등이 있다. 특히 Jena와 Sesame는 본 논문에서 제안한 시스템과 비교할 수 있는 온톨로지 정보 관리 시스템으로 주목할 만하다. Jena는 HP Laps에서 시맨틱 웹 연구의 일환으로 연구 중에 있는 오픈 소스로서 시맨틱 웹 애플리케이션을 구축하기 위한 자바 프레임워크이다. 특징으로는 RDF, RDFS, DAML+OIL[11], OWL에 대한

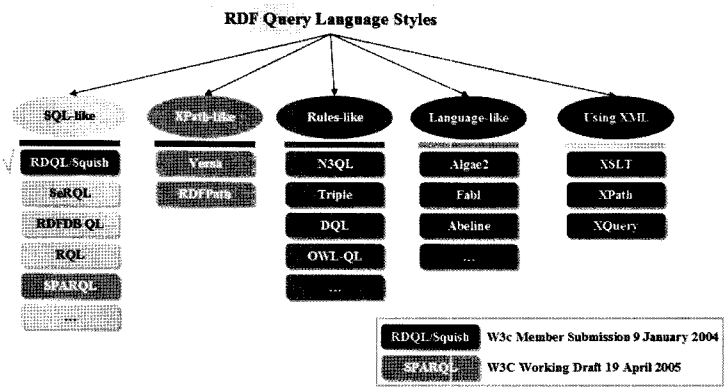
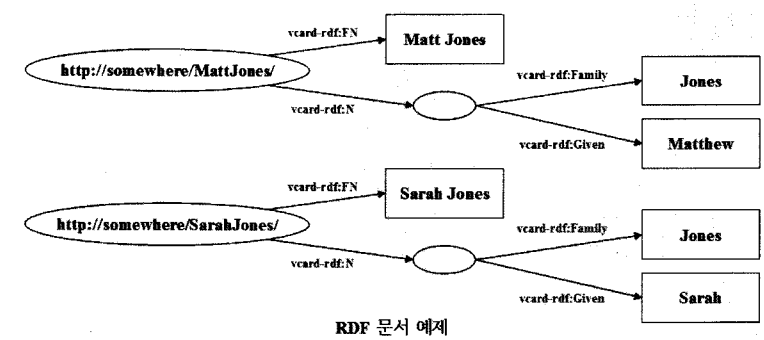


그림 1 시맨틱 정보 질의 언어의 분류



RDF 문서 예제

단일 트리플 패턴을 가지는 RDQL 질의

```
SELECT ?x
WHERE (?x  vcard-rdf:FN  "Matt Jones")
USING vcard-rdf FOR <http://www.w3.org/2001/vcard-rdf/3.0#>
```

결과

```
?x
<http://somewhere/MattJones/>
```

다중 트리플 패턴을 가지는 RDQL 질의

```
SELECT ?y
WHERE (?x  vcard-rdf:Family  "Jones"),
      (?y  vcard-rdf:N       ?x)
USING vcard-rdf FOR <http://www.w3.org/2001/vcard-rdf/3.0#>
```

결과

```
?y
<http://somewhere/MattJones/>
<http://somewhere/SarahJones/>
```

그림 2 RDQL 예제

문서 저장 및 관리를 위한 API를 제공하며 온톨로지 데이타에 대한 질의 언어로서 RDQL를 지원한다. Aduna (NLnet Foundation와 OntoText에서 공동연구)에서 연구 중에 있는 Sesame는 RDF와 RDFS를 저장, 질의, 추론하기 위한 오픈 소스 자바 프레임워크이다. 주요 특징으로는 RDF와 RDFS로 기술된 문서를 저장 관리하는 데이타베이스 시스템으로서 사용된다. 자바 라이브러리를 제공하기 때문에 RDF/S를 처리해야 할 필요가 있는 애플리케이션에서도 사용될 수 있다. 또한 BOR[12] 패키지를 설치함으로써 OWL 및 DAML+OIL을 저장 및 관리할 수 있도록 하고 있다.

Jena는 메모리 기반의 시맨틱 모델을 사용해서 MySQL, PostgreSQL, Oracle 9i 등과 같은 데이타베이스에 온톨로지 문서들을 저장한다. Sesame의 경우에는 문서 저장소로서 MySQL, PostgreSQL, Oracle 9i를 사용할 수 있으며 웹 인터페이스를 통해서 쉽게 온톨로지 문서를 저장 및 삭제할 수 있도록 한다. Sesame와 Jena는 저장 시스템을 빌딩할 때 RDF의 주요 속성들을 그대로 반영하도록 설계되어 있다. 시맨틱 웹에서 자원은 트리플 데이타 구조인 (주어, 서술어, 목적어)에 의한 단일 문장(Statement)으로 표현된다. 다시 말해서, Sesame와 Jena는 트리플 구조에 따라 설계된 데이타 스키마 위에서 RDF 문서를 저장하고 관리한다. [13]에서 제안한 데이타 모델은 Sesame와 Jena와 비슷한 구조를 가진다. 이와 비슷하게 [14]는 RDFS 문서의 기본 구성체인 클래스와 프로퍼티를 사용하도록 한다. 비록 소규모의 문서 관리 시스템에서는 매우 효율적이지만 대용량의 온톨로지 문서에 대해서는 효율성이 감소되는 문제점을 가지고 있다. 위에서 언급한 시스템들은 기존의 RDBMS를 이용하여 설계된 것에 비해 [10]은 RDF 문서들에 대한 자신의 저장 시스템을 개발하였다. 이러한 이유 때문에 이 시스템은 RDF 트리플에 의해 표현된 자원들을 관리할 때 가볍고 효율적이라고 볼 수 있다. 그러나 DAML+OIL 및 OWL 기반의 문서들을 관리를 위한 시스템으로 쉽게 확장할 수 없다.

2.3 동기 부여

일반적으로 대용량의 온톨로지 문서들을 안정되게 저장하고 효율적으로 RDQL 질의를 처리하며 저장된 시맨틱 정보를 모순없이 유지 관리할 수 있는 온톨로지 관리 시스템을 위해 다음과 같은 요구사항을 필요로 한다.

- 첫 번째, 시맨틱 정보를 저장하고 관리하기 위한 안정된 메커니즘에 대한 필요성
- 두 번째, 효율적인 RDQL 질의 처리 메커니즘에 대한 필요성
- 세 번째, 저장된 시맨틱 정보의 모순 없는 유지 관리를 위한 메커니즘의 필요성

이러한 요구사항들을 위해 본 논문의 접근은 다음과 같이 간단하게 요약할 수 있다.

- 첫 번째 요구사항에 대해서 우리는 차별화된 관계형 데이타베이스 시스템을 개발하기 위해 OWL 인지 관계형 모델을 설계할 것이다. 이를 통해서 우리는 검증된 기존의 관계형 데이타베이스 시스템의 기능들을 이용할 수 있다.
- 두 번째 요구사항에 대해서 우리는 RDQL 질의에 대

응하는 단일 SQL 질의로의 변환을 통해서 데이타베이스 접근 횟수를 최소화하여 질의 성능을 향상시킬 것이다. 일반적으로 RDQL 질의 처리의 성능은 데이타베이스(혹은 디스크) 접근 횟수와 밀접한 관계가 있기 때문에, 가능한 한 데이타베이스(혹은 디스크) 접근 횟수를 최소화시키면서 RDQL 질의를 처리할 수 있어야 한다. 기존 Sesame 시스템의 RDQL 질의 처리기는 RDQL의 WHERE절에 있는 각각의 트리플 패턴에 대해 데이타베이스 질의 언어인 SQL로 변환하고, 각각의 SQL 질의의 중간 결과를 메모리에서 조인하는 인메모리 조인(In-Memory Join) 알고리즘을 사용한다. 이러한 방법은 데이타베이스 접근 횟수가 트리플 패턴의 개수에 비례하며, 또한 트리플 패턴에 대한 중간 질의 결과에 대한 메모리 사용량도 증가한다.

- 세 번째 요구사항은 온톨로지의 성장 및 발전에 관련되기 때문에 본 논문의 주제와 벗어난다.

3. OWL 인지 관계형 모델

Jena 시스템은 인메모리 기반의 Jena 모델이라는 구조를 가지고 있기 때문에 대규모의 시맨틱 문서 저장 시스템으로는 부적당하다. Jena는 Sesame+BOR와 달리 사용자가 온톨로지 관리 시스템안에 저장된 모든 문서의 시맨틱 정보에 대한 질의 요청을 어렵게 하는 구조를 가지고 있다. 왜냐하면 Jena는 각각의 URI 기반의 문서에 대해서 Jena 모델을 생성하고 모델 단위로만 질의 처리를 할 수 있기 때문이다. 만약 사용자가 여러 문서에 질의 요청을 하게 된다면, Jena는 여러 문서를 하나의 모델로 병합하여 질의 요청을 해야만 한다. 이에 반해서 Sesame+BOR는 온톨로지 관리 시스템 안에 저장된 모든 문서들에 대해 시맨틱 정보에 대한 질의 요청을 할 수 있다.

본 논문의 목적은 OWL 온톨로지 문서들을 효율적으로 저장할 수 있고 저장된 문서들을 RDQL로 질의할 때 효율적으로 검색할 수 있는 데이타베이스 스키마를 제안하는 것이다. 대체로 Jena는 단일 문서 기반의 저장 시스템으로 설계되었다. 다시 말해서 우리가 시맨틱 문서를 Jena에 저장할 때 마다 jena_*gntn*_reif, jena_*gntn*_stmt와 같은 두 개의 테이블이 다른 기본 테이블들과 공유되면서 새롭게 생성된다. 여기서 *n*은 각각의 문서에 의존하는 그래프 ID이다. Jena에서 질의 처리를 하는 경우에, 우리는 문서를 Jena 모델로 불리는 메모리 모델로 로드한 다음에 각각의 문서에 대해서만 질의할 수 있다. 이와 달리 Sesame+BOR는 저장된 모든 문서들이 공유된 공통의 관계형 스키마를 제공하기 때문에 질의는 Sesame+BOR 시스템에 저장된 모든 문서들을 대상으로 한다. 하지만 원본 문서로부터 추론에 의해 더 많은 추가적인 정보를 생성하기 때문에 Sesame+BOR의 테이블들은 대체적으로 공간 복잡도가 높음을 주목하자. 이러한 관점에서 본 논문에서 제안하는 온톨로지 관리 시스템은 Sesame+BOR와 비슷한 데이타베이스 스키마 구조를 가지고 있다. 하지만 Sesame+BOR가 14개의 테이블을 통해서 OWL 문서를 저장하지만 본 논문에서는 7개의 테이블(그림 3)을 사용하고 있다. Sesame+BOR와의 차이점은 클래스 계층, 프라퍼티 계층, 인스턴스 테이블을 제거함으로써 최소의 데이타가 저장되도록 하였다. Sesame+BOR 같은 경우에 rdfs:subClassOf와 rdfs:subPropertyOf에 대한 질의의 효과적인 질의 처리를 위하여 별도의 테이블을 두었지만 이에 대한 정보는 B+-tree

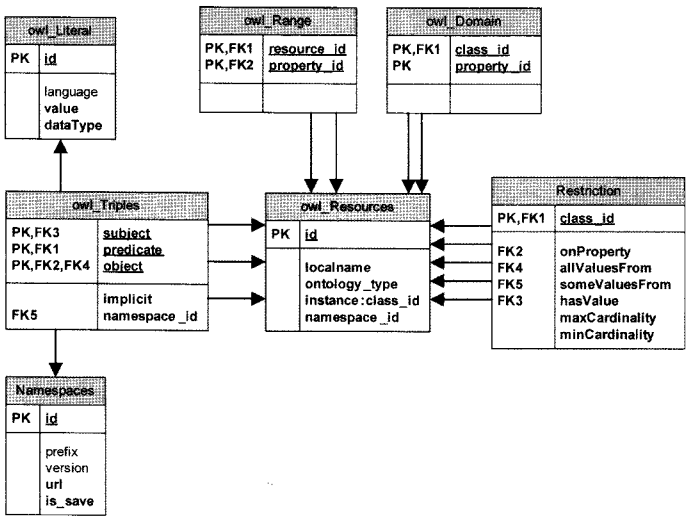


그림 3 OWL 관계형 데이타베이스 스키마

인덱스로 구축된 트리플을 저장하는 테이블을 둠으로써 성능의 향상을 크게 못보기 때문이다. 이에 본 논문에서는 임베디드 시스템과 같은 스토리지가 제한적인 환경에서 질의 성능을 최대화 시키도록 하는 데이타베이스 스키마를 제안하였다. 본 논문에서는 원본 문서들과 비교해서 정보의 손실 없이 가능한 적은 데이타를 포함하도록 공간 복잡도를 최소화 시키는데 초점을 두었다. 이러한 특징 때문에 본 논문에서 제안한 시스템은 홈 네트워킹, 텔레매틱스, 지능형 로봇과 같은 제한된 자원을 가지는 임베디드 시스템에 적당하다고 볼 수 있다.

OWL은 RDF/S를 기반으로 확장되었기 때문에 OWL 문서의 구성체는 RDF/S의 일곱 개의 주요 구성체인 rdfs:Resource, rdfs:Class, rdfs:subClassOf, rdf:Property, rdfs:domain, rdfs:subPropertyOf, rdfs:range 로부터 상속되었다. 그 중에서 우리는 그림 3에서처럼 세 개의 주요 구성체에 기반으로 한 관계형 데이타 스키마를 가지는 저장 시스템을 설계하고 구현하였다. 기본적으로 본 논문에서 제안한 스키마는 다음과 같은 세 개의 정책을 기반으로 한다. 첫째, 문서를 저장하는 동안 정보 손실이 없어야 한다. 둘째, OWL 정보의 저장 및 검색에 대한 처리의 효율성이 중요하게 반영되어졌다. 온톨로지 관리 시스템의 효율성은 RDQL 질의 처리에 대한 비용에 의해 주로 평가되어 진다. 일반적으로 효율성은 DBMS에서의 데이타 처리 비용과 추론 엔진에서의 질의 처리 비용과 관련되어 진다. 따라서 그림 3의 데이타베이스 스키마는 효율성을 극대화하기 위해 테이블의 수와 저장된 데이타의 양을 최소화하도록 설계되었으며 질의 처리 비용을 감소시키기 위해 RDQL 질의에 대응하는 SQL 질의 변환 기법을 제안할 것이다. 마지막으로 제안된 온톨로지 관리 시스템은 대용량의 데이타 저장에 대해 안정적이어야 한다. 이것은 시스템이 미래에 기술되는 새로운 온톨로지를 커버할 수 있어야 됨을 의미한다. 본 논문에서 제안한 데이타베이스 스키마는 owl_Triples라는 테이블을 통해서 트리플 기반의 그래프 모델을 지원하기 때문에 새롭게 기술되어 지는 온톨로지를 효과적으로 저장 및 유지관리 할 수 있다. 그림 3에서 보는 것처럼 7개의 테이블을 가지고 있으며 아래와 같은 특징을 가지도록 설계되었다.

- 리터럴 문자열을 처리하기 위한 텍스트 검색 지원 : owl_Literal 테이블
- URI에 대한 효율적인 지원 : Namespaces 테이블
- XML 스키마 데이타타입에 대한 지원 : owl_Literal 테이블의 datatype 필드
- OWL 프로퍼티 제약에 대한 지원 : Restriction 테이블

owl_Literal 테이블은 리터럴 스트링에 대한 정보를 관리함으로써 특정 문자열을 참조하는 자원을 빠르게 검색할 수 있도록 한다. Namespaces 테이블은 URI를 참조하는 다른 테이블에 대한 유지 관리가 쉬워지며, OWL 버전 정보를 유지함으로써 OWL 온톨로지 관리 시스템의 온톨로지 일관성을 보장하는 목적으로 사용된다. owl_Literal 테이블에서 datatype 필드는 다양한 XML 스키마 데이타 타입을 유지할 수 있도록 한다. 마지막으로 OWL은 프로퍼티 제약 정보를 owl:onProperty, owl:allValuesFrom, owl:someValuesFrom, owl:hasValue, owl:maxCardinality, owl:minCardinality를 이용하여 표현할 수 있기 때문에 효율적인 제약 정보를 질의하기 위해 설계된 테이블이다.

4. SQL 기반의 RDQL 질의 처리

OWL 문서에 대한 시맨틱 질의 처리에서 추론은 W3C에 의해 제안한 OWL 스펙에 정의된 OWL 시맨틱을 기반으로 하고 있다. 이러한 추론의 주된 목적은 새로운 트리플들이 더 이상 생성되지 않을 때까지 새로운 트리플을 찾는 것이다. 본 논문에서 제안한 온톨로지 관리 시스템의 OWL 데이타에 대한 추론 방법은 다음과 같다. 첫째, OWL 인지 관계형 모델에 OWL 문서가 저장될 때 OWL 스펙에 정의된 OWL 시맨틱을 적용하여 추론된 트리플들을 생성한다. 둘째, RDQL 질의가 처리될 때 데이타베이스 질의 언어인 SQL을 이용하여 효율적으로 처리한다. 첫째에 대해서는 OWL 스펙을 따르기 때문에 본 논문에서는 논의하지 않으며 두 번째에 대해서 우리는 다음과 같은 RDQL 질의 처리 엔진을 통해 목적을 달성할 수 있다. 본 논문에서 설계한 RDQL 질의 처리 엔진은 그림 4와 같이 3단계로 구성된다.

- 단계1: RDQL 파서는 RDQL 질의를 입력으로 하여 RDQL의 트리플 패턴을 RDF 그래프 모델로 변환한다. 이 단계에서는 기존의 RDQL 파서 모듈을 본 연구 환경에 적합하도록 재개발하였다.
- 단계2: RDQL2SQL은 RDQL 파서를 통해서 생성된 RDF 그래프 모델을 입력으로 하여 이에 대응하는 SQL 질의 언어를 생성한다.

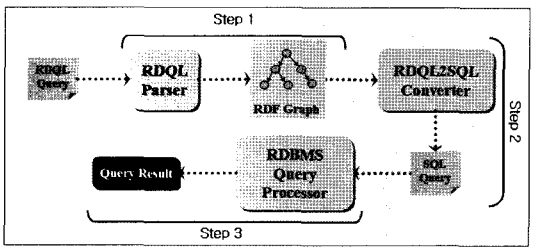


그림 4 온톨로지 관리 시스템의 전체 스텝

• 단계3: RDQL2SQL을 통해서 생성된 SQL 질의를 기존의 관계형 데이타베이스 질의 처리 엔진을 이용하여 처리 결과를 얻는다.

4.1 단계 1: RDQL 파서

RDQL 파서는 입력으로 들어오는 RDQL 질의를 RDF 그래프로 변환한다. 이때 RDF 그래프는 트리구조를 가진다. 이 과정에서 RDQL 파서는 그림 5와 같이 RDQL 질의의 WHERE절에 있는 각각의 트리플 패턴을 하나의 RDF 그래프로 변환한다. 트리플 패턴에 있는 주어와 목적어는 그래프의 노드로 변환하고, 서술어는 방향성 에지로 변환된다. 그리고 이러한 트리플 패턴은 RDF 그래프는 가상의 "root" 노드에 연결된다. 이와 같은 방법으로 모든 트리플 패턴들을 그래프 노드 및 에지로 변환한 후에 동일한 노드를 서로 연결하여 궁극적으로 하나의 RDF 그래프를 생성한다. 그러나 RDF 그래프는 루프가 발생될 수 있기 때문에 루프를 제거하기 위해서 RDQL의 트리플 패턴을 RDF 그래프로 변환할 때 루프가 만들어지는 브리지 노드를 복사하여 루프가 생기지 않도록 한다. 트리플 패턴을 RDF 그래프로 변환하는 다양한 예는 5장 실험에 있는 그림 10과 같다.

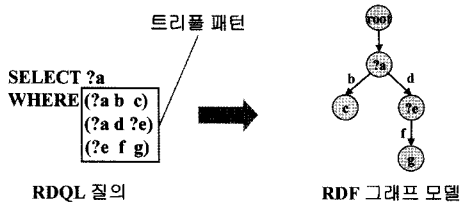


그림 5 RDQL 트리플 패턴으로부터 RDF 그래프 모델로의 변환

4.2 단계 2: RDQL2SQL 변환기

RDQL2SQL 변환기는 RDQL 질의 처리 엔진에서 핵심이 되는 모듈로서 RDF 그래프 모델에 대응하는 SQL 질의를 생성한다. 이 모듈은 아래에서 설명하는 변환 알고리즘에 의해 구현된다. 변환 알고리즘은 정리 1, 2를 기반으로 RDF 그래프 모델의 깊이 우선 탐색을 통해서 이루어진다.

정리 1. RDF 그래프를 이루는 트리플의 집합을 T = {t_1, t_2, …, t_n}라고 할 때 T의 각각의 요소는 그림 6과 같이 하나의 SQL 질의로 변환된다. 즉, $t_1 \rightarrow s_1$, $t_2 \rightarrow s_2$, … $t_n \rightarrow s_n$으로 변환된다. 단, SQL 질의의 집합을 S = {s_1, s_2, …, s_n}로 정의한다.

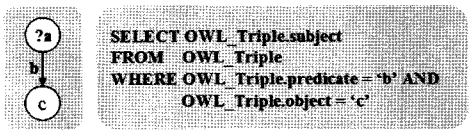


그림 6 단일 트리플로부터 SQL 질의로의 변환

정리 2. 2개의 RDF 트리플이 주어 또는 목적어를 공유할 때 아래의 패턴을 따른다.

- 패턴 1: 2개의 트리플이 동일한 주어를 가지는 경우 (그림 7의 패턴 1)
  트리플 ①과 ②의 SQL질의는 ①.subject = ②.subject의 조건을 가지는 내부조인(Inner Join)으로 조인된다. 정의 1에 의해 트리플 ①, ②로부터 변환된 SQL 질의를 각각 SQL(①), SQL(②)라고 할 때, 다음과 같이 정의한다.
  SQL(①) JOIN SQL(②) ON SQL(①).subject=SQL(②).subject
- 패턴 2: 2개의 트리플이 ①.object=②.subject일 경우 (그림 7의 패턴 2)
  트리플 ①과 ②의 SQL질의는 ①.object = ②.subject의 조건을 가지는 내부조인(Inner Join)으로 조인된다. 이를 다음과 같이 정의한다.
  SQL(①) JOIN SQL(②) ON SQL(①).object=SQL(②).subject
- 패턴 3: 2개의 트리플이 ①.object=②.object일 경우 (그림 7의 패턴 3)
  패턴 3의 경우 그림 7에서 보는 것처럼 루프가 생기지 않도록 변환되기 때문에 패턴 1과 패턴 2로 처리될 수 있다.

그림 6에서 보는 것처럼 정리 1은 RDF 그래프에서 (주어, 서술어, 목적어)를 이루는 트리플이 이에 대응하는 SQL 질의로 변환됨을 나타내며 정리 2는 RDF 그래프 모델이 SQL 질의로 변환될 때 RDF 그래프내에서 나타나는 모든 패턴에 따른 SQL 질의 변환 기법을 정리한 것이다. 그림 7에서 패턴 1은 SQL(①)과 SQL

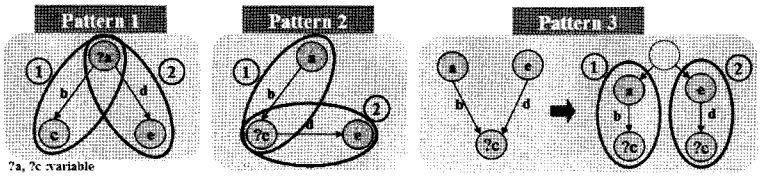


그림 7 RDF 그래프에서 세 가지 패턴

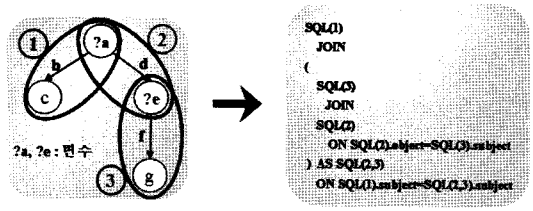


그림 8 RDF 그래프로부터 SQL 질의로 변환 예

(②)가 주어인 "?a"를 서로 공유하기 때문에 "SQL(①).subject = SQL(②).subject"의 조건을 가지는 내부조인으로 조인된다. 패턴 2도 마찬가지로 내부조인의 조건으로 "SQL(①).object = SQL(②).subject"를 가지며 SQL(①)과 SQL(②)의 내부조인으로 조인된다. 위와 같이 정리 1, 2를 통해서 RDF 그래프 모델에서 SQL질의로 변환할 수 있는 기법을 제공해 줄 수 있다. 정리 1과 2로부터 RDF 그래프 모델에 대응하는 SQL질의로 변환하는 예제는 그림 8과 같다.

지금까지 RDF 그래프에서 SQL질의로 변환하기 위한 이론적인 면을 봤다. 여기서부터는 정의 1, 2를 이용하여 SQL질의로 변환하는 알고리즘을 소개한다.

```
알고리즘 : RDQL2SQL
입력 : RDQL 그래프 모델
출력 : SQL 질의

(1) RDF 그래프 모델에서 깊이 우선 탐색을 통해서
    그래프의 단말노드 또는 SQL 질의로 변환된 트리플을
    탐색한다.
(2) RDF 그래프의 단말 노드에 도달하면 정의 1에 의해
    (주어, 서술어, 목적어)에 해당하는 트리플을
    SQL질의로 변환한다.
(3) 변환해야 될 대상이 단말 노드가 아니고 정리 2의
    2가지 패턴에 만족될 경우 두 개의 변환된 SQL질의를
    패턴 1, 패턴 2에 맞게 변환한다.
(4) RDF 그래프의 탐색이 끝날 때 까지 (1), (2), (3)을
    반복한다.
```

그림 9는 알고리즘 RDQL2SQL를 통해서 SQL 질의로 변환하는 과정을 보여준다. RDF 그래프 모델에서 깊이 우선 탐색이기 때문에, 먼저 ①이 정리 1에 의해서 SQL(①)로 변환되고 그 다음 ③이 SQL(③)으로 변환된다. 다음에 ②가 SQL(②)로 변환되고 정리 2의 패턴 2에 의해 SQL(②)와 SQL(③)이 조인된다. 마지막으로 그림 9의 5로 변환된다.

RDQL의 WHERE절이 가질 수 있는 트리플 패턴은 (a,b,c), (?a,b,c), (a,?b,c), (a,b,?c), (?a,?b,c), (?a,b,?c), (a,?b,?c), (?a,?b,?c)가 되며 트리플로부터 SQL질의로의 변환은 간단한 규칙에 의해서 변환이 된다. 첫째, 변수('?'로 시작하는 노드명)는 SQL질의의 "SELECT" 부분에 위치한다. 둘째, 변수가 아닌 노드는 SQL질의의 "WHERE"절에 위치하고 2개 이상의 노드일 경우 SQL질의의 "AND"에 묶여진다. 두 개의 규칙에 의해서 그림 6과 같이 SQL질의로 변환된다. 노드 "?a"는 변수이므로 SELECT에 위치한다. 즉 "?a"는 트리플의 주어에 해당되므로 "SELECT OWL_Triple.subject로 변환된다. 그리고 간선 "b", 노드 "c"는 변수가 아니므로 WHERE에 위치하여 그림 6과 같이 변환된다. 여기에서 OWL_Triple는 시맨틱 웹 온톨로지 문서를 저장하는 시스템에 의존하게 된다. 이 논문에서 OWL_Triple는 RDBMS의 테이블로서 OWL문서를 트리플(주어, 서술어, 목적어)로 저장하기 위한 것이다. 정의 및 설명을 간략하고 명확하게 하기 위해 이름공간은 고려하지 않았다. 이름공간을 고려하는 것은 간단하기 때문에 이 논문에서는 생략한다. 그래서 OWL_Triple는 속성으로서 "subject", "predicate", "object"를 가지는 테이블이다.

4.3 단계 3: 관계형 데이타베이스 질의 처리기

RDQL2SQL 변환기를 통해 생성된 SQL 질의는 기존의 관계형 데이타베이스 질의 처리기를 통하여 최종 질의 결과를 얻게 된다. 일반적으로 온톨로지 관리 시스템은 네트워크 기반의 다중 사용자를 위한 시스템으로 설계된다. 여기에서 질의 성능에 큰 영향을 미치는 요소 중에 하나는 네트워크의 상태라고 볼 수 있다. 일반적으로 Sesame와 Jena는 온톨로지 저장소가 네트워크 상에 존재하게 되는데 이러한 구조에서는 질의에 대한 네트워크 오버헤드를 최소로 할 필요가 있다. 기존의 온톨로지 관리 시스템에서 RDQL의 질의 처리 방식은 Sesame에서 처럼 RDQL의 각 트리플 패턴에 대응하는 여러 개의 SQL문을 생성하여 각 SQL문에 대응하는 질의 결과를 중간 결과로 메모리에 저장한다. 그런 다음에 모

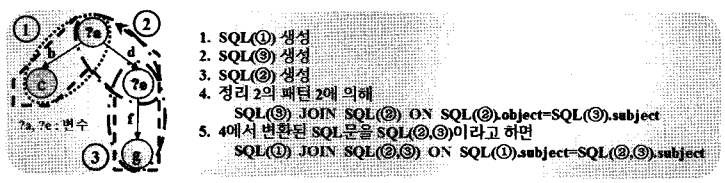


그림 9 RDQL2SQL 알고리즘의 예

든 SQL문이 처리되면 메모리에 저장된 중간 결과의 조인을 통해서 최종 결과를 얻을 수 있다. 문제점은 SQL문을 트리플 패턴 수만큼 개별적으로 처리되는데 있다. 즉 SQL문을 처리하기 위한 네트워크 통신횟수의 증가로 성능이 떨어지는 문제점이 발생한다. 또 하나의 문제점은 각 SQL문의 결과를 중간결과로 메모리상에 유지하는 비용과 이들을 조인하는 비용이 발생하는데 있다. 하지만 본 논문에서 제안하는 SQL 기반의 RDQL 처리 방식은 이러한 문제를 해결하는 효율적인 방법이라고 할 수 있다.

5. 실 험

Sesame는 두드러진 온톨로지 관리 시스템일 뿐만 아니라 본 논문과 비슷하게 접근하고 있기 때문에 우리는 제안된 시스템을 Sesame와 비교한다. RDQL 시맨틱 질의 처리에 대한 효율성 측면에서 두 시스템을 비교하기 위해서 우리는 그림 10에서처럼 이전 장에서 언급한 5개의 서로 다른 질의 패턴에 대해서 질의 처리 시간을 측정하였다.

- query 1 : RDQL의 WHERE절에 단일 트리플 패턴이 기술된 형태로서 서술어 "p"와 목적어 "o"를 만족하는 주어인 변수 "?s"를 결과 값으로 리턴한다.
- query 2 : 정리 2의 패턴 1에 대한 RDQL과 이를 RDF 그래프로 변환한 것을 보여준다. 서술어 "p1"과 "p2", 목적어 "o1"과 "o2"를 만족하는 주어 "?s"를 결과 값으로 리턴한다.
- query 3 : 정리 2의 패턴 2에 대한 RDQL과 이를 RDF 그래프로 변환한 것을 보여준다. 주어 "s"와 서술어 "p1"을 만족하는 "?o1"중에서 서술어 "p2"와 목적어 "o2"를 만족하는 "?o1"을 결과 값으로 리턴한다.
- query 4 : 정리 2의 패턴 3에 대한 RDQL과 이를 RDF 그래프로 변환한 것이다. 패턴 3은 목적어 "?o"에 의해서 루프가 생기기 때문에 이를 처리하기 위한 RDF 그래프로 변환된 모습을 보여준다. 이렇게 함으로써 패턴 3은 패턴 1과 패턴 2로 처리될 수 있다. 주어 "s1"과 서술어 "p1", 주어 "s2"와 서술어 "p2"를 만족하는 "?o"를 결과 값으로 리턴한다.
- query 5 : 패턴 1과 패턴 2가 같이 포함된 질의 예이다. 실제로 패턴 3이 포함되어 있지만 변환되는 RDF 그래프는 패턴 3은 패턴 1과 2로 처리됨을 인지하자. 이때 점선으로 된 원은 "root" 노드를 나타낸다. 질의의 의미는 주어 "s1"과 서술어 "p1", 주어 "s2"와 서술어 "p2"를 만족하는 "?o"중에서 "?o"의 서술어 "p3"와 목적어 "o3", 서술어 "p4", 목적어 "o4"를 만족하는 자원들을 결과 값으로 리턴한다.

위에서 설명한 다섯 개의 질의에 대한 성능 결과를 분석하기 위해서 본 논문에서는 로봇 온톨로지에서의 사용을 목적으로 한 온톨로지를 사용하였다. 온톨로지의 구성은 Object, Environment, Element, Shape 등으로 구성되어져 있으며 표 2와 같이 클래스, 프라퍼티, 인스턴스로 구성되어져 있다.

표 2와 같이 구성된 온톨로지에 대한 질의 성능을 분

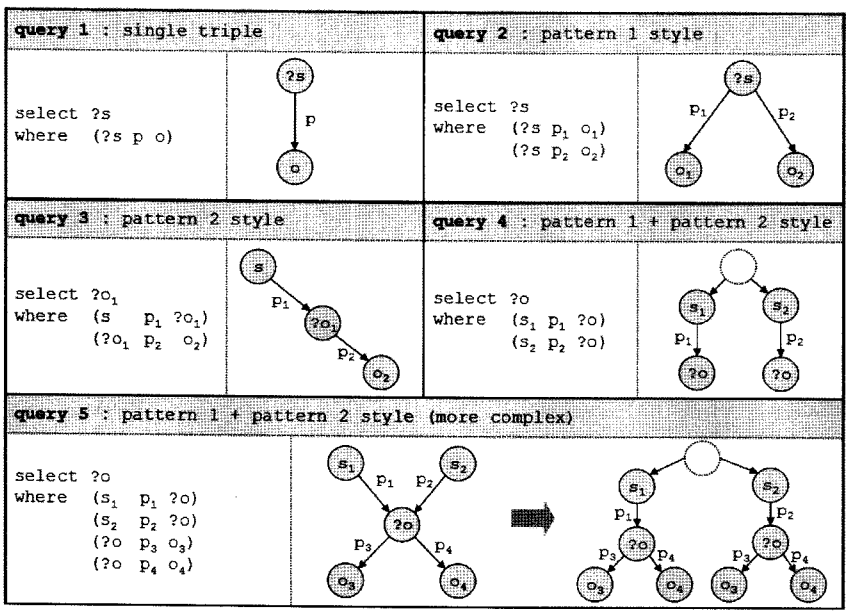


그림 10 다섯 개의 다른 RDQL 질의 패턴

표 2 로봇 온톨로지 구성

	System Ontologies	Robot Ontologies	Total
Classes	55	592	647
Properties	187	908	1095
Instances	44	188	232

석하기 위해서 Sesame, Jena, 본 논문에서 제안하는 데이타베이스 스키마로 구성된 질의 엔진을 서로 비교하였다. 비교를 위해서 MySQL 5.0.18를 사용하여 온톨로지 저장소를 네트워크상에 두고 질의 엔진을 클라이언트에 둠으로써 서로 질의 성능을 비교하였다. 하지만 Jena의 실험 결과는 제외시켰다. 왜냐하면 질의 처리 성능을 단 시간에 확인할 수 없을 정도로 나쁘게 나왔기 때문이다. 본 논문에서는 Jena의 실험 결과를 보기 위해서 Jena 모델을 추론된 모델과 추론하지 않은 모델 두 개로 분류해서 실험을 하였다. 추론하지 않은 모델은 질의 결과의 값이 0개가 나와서 제대로된 질의로서 판단이 안되기 때문에 제외시켰을 뿐만 아니라 추론된 모델에서는 질의 결과를 볼 수가 없었다. 예를 들어 "query_1"에 대해서 자바 힙 메모리를 1024Mbyte로 잡았을 경우에 메모리 오브플로우가 났음을 확인하였다.

우리는 본 논문에서 제안한 시스템의 최종 질의 결과들이 그림 11(b)에서 보는 것처럼 서로 같은 상황에서 그림 11(a)와 같이 Sesame보다 더 좋은 성능을 가짐을 볼 수 있었다. 그림 11(a)에서 "query_1"의 경우 단일 트리플 패턴에 대한 질의이기 때문에 성능에 별 차이가 없었지만 다중 트리플 패턴에 대해서는 좀더 나은 성능을 보임을 알 수 있다. 결과적으로 제안된 SQL 기반의 시맨틱 정보 관리 시스템이 복잡하게 표현된 RDQL 질의에 대해서 더 효율적임을 알 수 있다.

6. 결 론

본 논문은 OWL 관계형 모델에서 SQL 기반의 시맨틱 정보 관리 시스템을 기술하였다. 우리는 시맨틱 데이타의 크기, 시맨틱 질의 처리의 성능, 시맨틱 데이타 유지 관리의 관점에서 온톨로지 관리 시스템의 일반적인 요구사항을 논의했다. 이러한 요구사항에 대해 우리는 구별된 관계형 데이타베이스 기능성 및 SQL 기반 질의 처리 메커니즘을 이용하기 위한 OWL 관계형 모델을 제안했다.

앞으로의 연구 방향으로 우리는 OWL 스펙에서 언급된 모든 OWL 시맨틱을 완전히 지원하기 위해 다양한 시맨틱 질의 언어(특히, OWL-QL)의 확장에 초점을 맞출 계획이다.

참 고 문 헌

[1] Dave beckett and et al.: RDF/XML Syntax Specification (Revised), W3C Recommendation 10 February 2004. See http://www.w3.org/TR/rdfsyntax-grammar/.

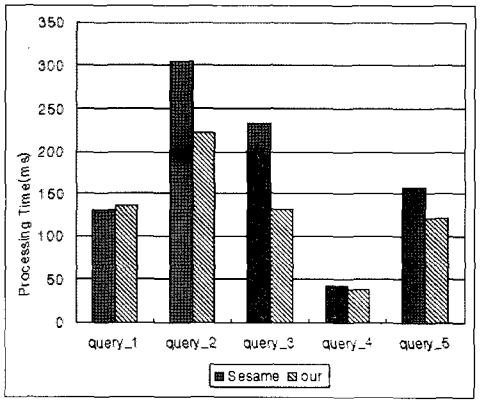
[2] Dan Brickely, R.V. Guha and et al.: RDF Vocabulary Description 1.0: RDF Schema, W3C Recommendation 10 February 2004. See http://www.w3.org/TR/rdf-schema/.

[3] Deborah L. McGuinness, Frank van Harmelen: OWL Web Ontology Language Overview W3C Recommendation. Feb 2004. See http://www.w3.org/TR/owlfeatures/

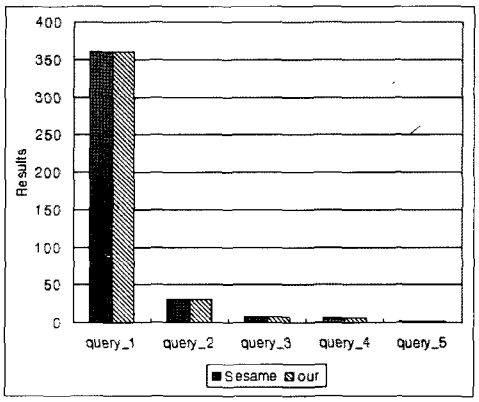
[4] T. Berners-Lee, J.Hendler, and O. Lassila : The semantic Web. Scientific American, Vol.284, No.5, pp. 34-43, 2001.

[5] Z. X. Pan and J. Heflin: DLDB: Extending relational databases to support semantic web queries, Workshop on Practical and Scalable Web Systems, pp. 109-113, 2003.

[6] Kevin Wilkinson, Craig Sayers, Harumi Kuno, Dave Reynolds: Efficient RDF Storage and Retri-



(a) 질의 처리 시간



(b) 질의 결과

그림 11 성능 평가

eval in Jena2. SWDB 2003, pp. 131-150, 2003.
[7] Jeremy Carrol, Brian McBride: The Jena Semantic Web Toolkit. HP-Labs, Bristor, 2001. See http://jena.sourceforge.net/
[8] J. Broekstra, A. Kampman, F. van Harmelen: Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema. In The Semantic Web-ISWC 2002, volume 2342 of Lecture Notes in Computer Science, pp. 54-68, 2002. See http://openrdf.org/
[9] Kilian Stoffel, Merwyn Taylor, James Hendler: Efficient management of very large ontologies. Proc. 14th Nat' l Conference AAAI, AAAI press, pp. 442-447, 1997.
[10] Ramanathan V. Guha, Rob McCool: TAP: a Semantic Web platform. Computer Networks, Vol.42, No.5, pp. 557-577, 2003.
[11] Ian Horrocks, Frank van Harmelen and et al.: DAML+OIL (March 2001). See http://www.daml.org/2001/03/daml+oil-index.html.
[12] Kiril Simov, Stanislav Jordanov: BOR: a Pragmatic DAML+OIL Reasoner. On-To-Knowledge project, 2002.
[13] Harris, S. and Gibbins, N: 3store: Efficient Bulk RDF Storage. In Proceedings of 1st International Workshop on Practical and Scalable Semantic Systems-PSSS 2003, pp. 1-15, 2003.
[14] Alberto Reggiori, Dirk-Willem van Gulik, Zavisa Bjelogrlic: Indexing and retrieving Semantic Web resources: the RDFStore model. Europe Workshop on Semantic Web Storage and Retrieval. SWAD 2003, 2003.

김 학 수

2004년 한양대학교 전자 컴퓨터 공학과(학사). 2006년 한양대학교 컴퓨터공학과(석사). 2006년~현재 한양대학교 컴퓨터공학과(박사과정). 관심분야는 데이타베이스, 시멘틱 마이닝, e-비즈니스

손 진 현

1996년 서강대학교 전산학과 학사. 1998년 한국과학기술원 전산학과 석사. 2001년 한국과학기술원 전자전산학과 박사. 2001년 9월~2002년 8월 한국과학기술원 전자전산학과 박사후 연구원. 2002년 9월~현재 한양대학교 컴퓨터공학과 조교수. 관심분야는 데이타베이스, e-비즈니스, 유비쿼터스 컴퓨팅, 임베디드 시스템