웹 온톨로지 저장소의 질의 처리 성능에 대한 비교 평가

정동원 [O1] 최명회 [1] 정영식 [2] 한성국 [2]
[1]군산대학교 정보통계학과
{djeong, cmh775}@kunsan.ac.kr
[2]원광대학교 컴퓨터공학과
{ysjeong, skhan}@wku.ac.kr

Comparative Evaluation on Query Processing Performance of Web Ontology Storages

Dongwon Jeong [O1] Myounghoi Choi[1] Young-Sik Jeong[2] Sung-Kook Han[2]
[1]Dept. of Informatics & Statistics, Kunsan National University
[2]Dept. of Computer Engineering, Wonkwang University

요 약

이 논문에서는 관계형 데이터베이스 모델 기반의 OWL 웹 온톨로지 모델을 보이고 이에 대한 실험 및 비교 평가 결과에 대하여 기술한다. OWL은 W3C에 의해 2004년 12월에 권고안으로 채택된 이후에 많은 연구가 진행되고 있다. 편집 도구 개발, 저장소 개발 및 OWL 기반의 추론 엔진까지 이와 관련된 다양한 연구가 진행 중이다. 특히 OWL 온톨로지의 영구적인 저장 및 관리를 위해 관계형 데이터베이스 모델을 이용한 많은 연구 결과들이 발표되고 있다. 이 논문에서는 널리 알려진 관계형 모델 기반의 저장소 보다 나은 성능을 제공하기 위해 제안한 모델에 대한 평가 결과에 대하여 기술한다. 기존 유사 연구의 경우, 비교 평가를 위한 평가 항목으로 온톨로지 로드 시간을 고려하기도 하지만 이 논문에서는 질의 응답 시간에만 초점을 둔다. 이는 매우 특수한 상황을 제외한 대부분의 상황에서 질의 처리 시간이 가장 중요한 요소이며 실질적인 활용성 측면에서 핵심적으로 다루어야 하는 평가 항목이기 때문이다. 실험을 위한 데이터로서는 많은 연구에서 활용하고 있는 LUBM 데이터 셋을 이용하며 실험 대상으로는 오픈 소스이며 널리 알려진 시스템인 Jena의 저장소와 Sesame를 이용한다. 실험 및 비교 평가 결과, 제안 시스템이 비교 대상 시스템들에 비해 나은 성능을 보임을 알 수 있다.

1. 서 론

OWL (OWL Web Ontology)은 W3C에 의해 권고안으로서 시맨틱 웹을 구현하기 위해 개발된 웹 온톨로지 서술 언어이다. 보다 지능적인 웹 개발을 위한 많은 기술들이 개발되었으며 이러한 기술들 중 하나가 웹 온톨로지 서술 언어이다 [1]. OWL 이전에 다양한 웹 온톨로지 서술 언어가 개발되었으며 대표적인 언어로는 RDF (Resource Description Framework), RDF-S (Resource Description Framework Schema), DAML (DARPA Agent Markup Language)+OIL (Ontology Interface Layer) 등을 들 수 있다 [2,3,4]. OWL은 이러한 언어들의 특징을 수용하고 기능을 통합한 언어라 할 수 있다. 따라서 기존 다른 언어에 비해 보다 강한 표현력과 함께 온톨로지를 이용한 추론 기능까지 지원한다 [5].

2004년 12월에 권고안으로 채택된 이후에 OWL을 기반으로 한 많은 연구가 진행되고 있으며 다양한 온톨로지 에디터, 추론 엔진, 온톨로지 저장 시스템 등이 개발되었다 [6, 7, 8, 9, 10, 11, 12, 13].

온톨로지 관리 및 활용을 위해 다양한 기능 개발이 요구되며, 특히 효율적으로 저장 및 관리할 수 있는 저장소 개발은 매우 중요한 기능 중 하나이다. 온톨로지 저장소는 그래프 모델을 파일 시스템을 이용하여 저장하는 접근 방법과 기존 데이터베이스 관리 시스템을 이용하여 저장하는 접근 방법으로 분류할 수 있다. 데이터베이스 모델 중 최근 가장 활발하게 활용되고 있는 방법은 관계형 모델을 저장소로 이용하는 방법로서 이는 관계형 데이터베이스의 다양한 장점을 활용할 수 있다는 장점을 지닌다. 또한 대다수의 데이터가 관계형 데이터베이스에 저장 및 관리되고 있다는 현실적인 상황에 보다 적합한 접근 방법이다.

관계형 데이터베이스 모델을 이용하여 OWL 웹 온톨로지를 저장 및 관리하기 위해 개발된 대표적인

저장소로는 Jena 데이터베이스 저장소, Sesame, DLDB, OWLJessKB 등을 예로 들 수 있다. Jena는 HP에서 개발한 API로서 자동으로 관계형 데이터베이스 기반의 저장 모델을 생성해 준다 [8]. 이 외에도 Jena는 추론 시스템 개발 등을 위한 다양한 오픈 API를 제공해 준다.

Sesame는 웹을 기반으로 한 온톨로지 저장 관리 및 추론 시스템 개발 기능을 제공한다 [9, 10]. DLDB는 SWAT 프로젝트의 일환으로 개발된 시스템으로서 현재까지 연구 결과, 가장 나은 성능을 제공하는 것으로 보이며 대용량 온톨로지 생성 도구를 개발하여 평가를 위한 대용량 온톨로지 셋을 생성에 이용하였다 [11].

기존의 저장 시스템은 각각 다양한 장점을 지니는 반면 여전히 여러 가지 문제를 내포한다. Jena의 경우에는 비정규화 된 저장 구조를 제공하며 단일 테이블에 모든 온톨로지 정보를 트리플 형태로 저장한다. 이는 조인 연산 처리시 조인 연산에 이용되는 테이블의 사이즈를 증가시켜 전체적인 성능 저하를 가져오게 된다. 또한 조인 연산이 아닌 특정 클래스 정보 검색과 같은 단순 검색의 경우에도 성능이 저하되는 문제점을 지닌다. Sesame의 경우, 저장 구조가 복잡하고 여러 연구 결과에서 성능이 매우 저하되는 문제점을 지님을 알 수 있다. DLDB의 경우, 성능이 가장 우수한 것으로 관련 연구 논문에서 언급하고 있지만 시스템 구조나 관련 소스 혹은 API가 공개되지 않아 그 정확한 성능을 평가하는데 어려움이 따른다. OWLJessKB는 자바로 개발했던 Jess 추론 시스템을 확장한 것으로 OWL 기반 추론 엔진을 탑재한 시스템이며 영구 저장소 모델을 제공한다. 그러나 앞서 언급한 시스템들 중에서 가장 낮은 성능을 보인다.

이러한 문제점을 해결하기 위해 [14]에서는 새로운 OWL 웹 온톨로지 저장소 모델을 제안하였다. 이 논문에서는 저장 시스템들의 로딩 시간에 대한 실험만을 수행하였으며 또한 간접적인 성능 비교 결과를 보였다. 활용 측면에서 무엇보다 중요한 평가 항목은 질의 응답 시간, 검색 연산 처리 성능이다. 그러나 [14]에서는 이에 대한 실험 및 비교 평가 결과를 수행하지 않았다. 또한 질의 처리 시간과 함께 질의에 대한 정확성 또한 매우 중요한 요소이다.

이 논문에서는 Sesame, Jena를 대상 시스템으로 하여 질의 처리 성능에 대한 실험 비교 평가를 수행하고 이에 대한 결과에 대하여 기술한다. 실험 온톨로지 셋은 SWAT 프로젝트에서 이용한 온톨로지 생성 도구인 UBA (Univ-Bench Artificial Data Generator)를 이용하여 생성한다.

이 논문의 구성은 다음과 같다. 제 2장에서는 [14]에서 제안한 시스템에 대하여 소개하고 제 3장에서는 평가 방법론에 대하여 기술한다. 제 4장에서는 실험 결과에 대하여 기술하고 제 5장에서는 결론 및 향후 연구 과제에 대하여 기술한다.

2. 웹 온톨로지 저장소 모델

이 장에서는 [14]에서 제안한 웹 온톨로지 저장소에 대하여 소개한다.

2.1 프레임워크

그림 1은 OWL 온톨로지를 저장하기 위한 전체적인 프레임워크를 보여준다. 그림 1의 프레임워크는 대부분의 저장 시스템에서 보여주는 일반적인 구조이다. OWL 문서가 입력으로 주어지면 파싱을 통해 OWL 문서에 대한 유효성 검사가 수행되고 OWL의 동일한 데이터 별로 분류된다. 저장소에 따라 그래프 모델을 생성한 후에 저장 연산이 이루어지기도 하지만, 이 논문에서의 제안하는 구조는 스트리밍 처리를 통해 동일한 데이터 형태별로 분류하여 메모리 상에 로드한다.

메모리 상에 로드되는 데이터는 단순히 순차적으로 벡터 형태로 추가되지 않고 사전에 정의된 메모리 데이터 구조에 맞게 사상되어 저장된다. 이 과정을 변환이라고 정의하며 변환되어 메모리에 저장된 데이터는 저장 모듈을 통해 특정 관계형 데이터베이스 관리 시스템에 저장된다.

저장된 온톨로지에 대한 질의 연산은 다양한 형태의 질의 언어를 통해 작성될 수 있다. 그러나 이 논문에서 기술하는 저장소는 SQL로 작성된 질의만을 처리하도록 제한된다. 그러나 다양한 질의어 처리를 위해 질의 처리 모듈을 탑재함으로써 SPARQL, KIF, CL 등과 같은 다양한 질의 언어를 처리할 수 있다.

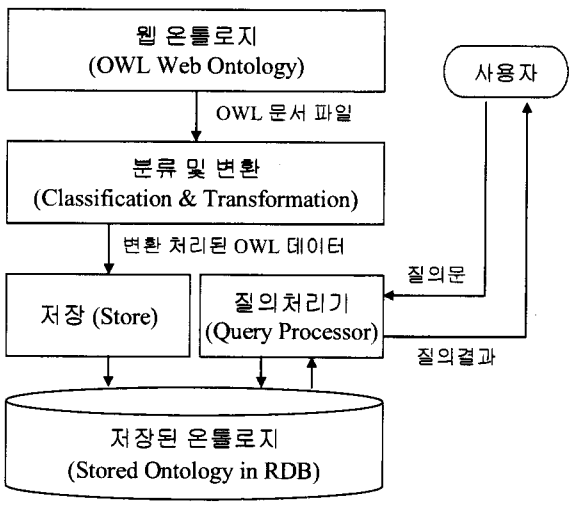

그림 1. 프레임워크

2.2 시스템 구조

그림 2는 제안 저장소 모델을 위한 시스템 구조를 보여준다. 파일 리더는 단순히 선택된 OWL 문서

파일을 판독하여 스트림으로 변환하는 기능을 수행한다. 요소 분류기는 OWL 문서로부터 데이터 분류에 따라 데이터를 식별하여 추출하는 역할을 수행한다. 즉 클래스, 인스턴스 등을 식별하여 분류하는 역할을 수행한다.

요소 분류기의 정보는 매핑 관리자에 전달되고 매핑 관리자는 전달된 OWL 데이터를 해당 메모리 저장소에 보관하게 된다. 메모리 상의 자료 구조는 OWL의 데이터를 클래스나 인스턴스 (Individual)과 같이 요소별로 관리할 수 있도록 정의된다. 이 단계까지를 메모리 로딩 단계로 정의할 수 있다. [15]에서는 이 단계에서의 평가 결과에 대하여 기술하고 있다.

입력된 OWL 데이터에 대한 메모리 로딩 작업이 완료되면 저장소 관리자에 의해 명시된 관계형 데이터베이스에 분류, 추출된 OWL 데이터를 저장하게 된다. 인터페이스 관리자는 사용자 질의에 대한 처리 인터페이스 모듈로서 사용자 질의를 데이터베이스 연결 관리자를 통해 데이터베이스 관리 시스템에 전달하고 이에 대한 결과를 사용자에게 반환해 주는 역할을 담당한다.

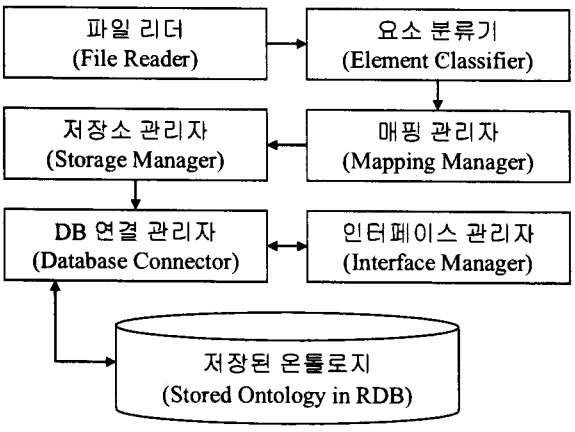

그림 2. 저장 시스템 구조

2.3 저장소 모델 및 저장 알고리즘

그림 3은 저장소를 위한 메타모델을 보여준다. 그림에서 개념(Concept)과 인스턴스(Instance)는 각각 OWL에서의 Class와 Individual로서 정의되어 있다. 제안 모델에서는 개념과 인스턴스를 분리하여 저장하며 관련 제약조건 등과 같은 정보들 또한 분리 저장된다.

Concept_Definiton 테이블은 OWL Class에 대한 정의 내용을 포함하며 Instance_Definition 테이블은 인스턴스 정보를 지닌다. 하나의 개념을 위한 여러 개의 인스턴스가 생성될 수 있기 때문에 Concept_Definition 테이블과 Instance_Definition 테이블 간 1:N의 다중성 (Multiplicity) 관계가 성립한다.

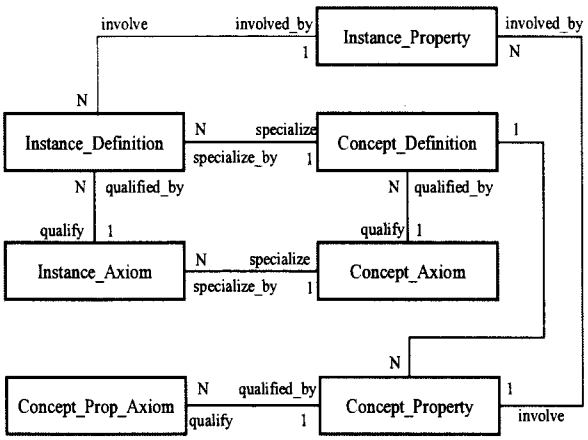

그림 3. 저장소를 위한 메타모델

다음은 그림 3에 표현되어 있는 저장소 모델을 통해 정의된 데이터베이스에 OWL 온톨로지를 로딩하는 전체적인 알고리즘을 보여준다. 알고리즘의 입력은 OWL 문서 (OWL_doc)이며 그 결과는 온톨로지를 저장하고 있는 릴레이션의 집합이다.

알고리즘에서 OWL 문서 파일을 입력 받으며 이에 대한 스트림 객체를 생성한다. 생성된 스트림 객체는 유효성 검사를 거쳐 분류 및 사상 과정에 필요한 연산을 수행한다. 마지막으로, 메모리 상에 저장되어 있는 OWL 데이터들이 지정된 영구 저장소로 저장된다.

```
Algorithm_Loading (Input: OWL_doc)
{
  //OWL document를 읽고 스트림 객체 생성
  owlStream = read(OWL_doc);
  //유효성 검사
  boolean vr = validator.check(owlStream);
  if(vr != true) {
    ExceptionHandler.show(INVALID_ERROR);
    return;
  }
  //OWL 데이터의 각 요소 추출 및 분류
  EClassifier classifier new EClassifier(owlStream);
  Mapper mapper = classifier.createMapper();
  //영구 저장소에 저장
  persistentManager.load(memManager);
}
```

3. 평가 방법론

이 장에서는 실험 환경 및 비교 항목 등의 평가 방법론에 대하여 기술한다.

3.1 비교 평가 항목 및 대상 시스템

이 논문에서는 질의 응답 시간에 대한 실험 및 비교 평가를 수행한다. 유사 연구 결과에서는 메모리 로딩 시간이나 데이터베이스 혹은 파일 시스템과 같은 영구 저장소로의 로딩 시간에 대해서도 평가하였다. 비록 로딩 시간이 중요한 평가 항목일지라도 몇몇 특수한 상황하에서의 경우이며, 일반적인 경우 오프라인 상에서 데이터 로딩이 수행된다. 이러한 관점에서 보다 중요하게 고려되어야 할 평가 항목은 질의 처리 시간, 즉 질의 응답 시간이라 할 수 있다. 따라서 이 논문에서는 실용성을 고려한 평가를 위해 질의 처리 성능에 대하여 평가한다.

평가 대상 후보 시스템으로는 많은 시스템들이 선택될 수 있다. 앞서 언급하였듯이, 현재 널리 알려진 시스템들은 Sesame, Jena, OWLJessKB, DLDB 등이다. 이 시스템들 중에서 Sesame와 Jena를 평가 대상 시스템 선택하고 오픈되지 않거나 낮은 성능으로 평가 의미가 없는 다른 시스템은 비교 평가 대상에서 제외한다.

3.2 실험 데이터 셋 및 시스템 환경

이 논문에서는 UBA (Univ-Bench Artificial Data Generator) 온톨로지 생성 도구를 통해 생성한 데이터 셋을 이용하여 실험을 실시한다. UBA는 DLDB 시스템에 대한 실험 및 타 시스템과의 비교 평가를 위해 개발된 도구로서 현재 많은 연구에서 실험을 위해 UBA 도구를 이용하고 있다. 생성되는 온톨로지 도메인은 대학으로서 학과, 교수, 학생 등의 주요 개념을 중심으로 계층 구조까지 생성해 준다 [16].

실험에 이용한 온톨로지 셋은 LUBM(1,0), LUBM(2,0), LUBM(3,0)이며 각 온톨로지 셋의 크기와 인스턴스 수는 표 1과 같다. 보다 방대한 크기의 온톨로지를 이용하여 실험을 수행할 수 있으나 표 1의 데이터 크기만으로도 시스템 간 성능 차이를 명확하게 알 수 있기 때문에 주어진 온톨로지 셋만을 이용한다.

표 1. 실험을 위한 온톨로지 셋

항목 / 데이터 셋	인스턴스 개수	온톨로지 크기
LUBM(1,0)	103,074	8.02 MB
LUBM(2,0)	237,142	18.40 MB
LUBM(3,0)	348,006	27.10 MB

실험을 위한 시스템 환경은 다음과 같다.

- CPU : Pentium D 3.0GHz
- 메모리 : 2GB
- 하드 디스크: 250GB
- 운영체제 : Windows XP Professional
- 개발 언어 : JAVA SDK 1.6.0
- DBMS : MySQL 4.1

3.3 실험을 위한 질의 정의

이 논문은 질의 응답 시간에 대한 평가 결과에 초점을 둔다. 이를 위해 네 개의 질의 패턴을 이용한다. 즉, 네 개의 질의 패턴은 클래스 정보를 검색하는 질의, 특정 클래스의 인스턴스를 검색하는 질의, 특정 클래스의 모든 하위 클래스를 검색하는 질의 및 특정 클래스의 하위 클래스에 해당하는 모든 인스턴스를 검색하는 질의이다. 정의한 질의 패턴에 대한 질의 예제는 표 2와 같다.

표 2. 실험을 위한 질의 예제

질의	질의 설명
Query1	모든 클래스를 검색하시오.
Query2	모든 대학원생을 검색하시오.
Query3	모든 교수 클래스를 검색하시오.
Query4	모든 교수들을 검색하시오.

4. 실험 및 평가

이 장에서는 3장에서 기술한 실험 환경 및 평가 방법에 따른 수행한 실험 결과 및 평가 결과에 대하여 기술한다.

그림 4~7은 각각 Query1, Query2, Query3 및 Query4에 대한 실험 결과를 보여준다. 각 실험 결과에서 제안 시스템이 나은 성능을 보임을 알 수 있다. 각 실험 결과에 대한 분석 결과는 다음과 같다.

먼저 그림 4에서, Query1에 대한 질의시 제안 시스템은 클래스 정보만을 비교하기 때문에 다른 시스템 비해 낮은 처리 비용이 요구되며 인스턴스 크기와는 무관하게 일정한 질의 처리 시간을 요구한다. 반면 Jena의 경우 인스턴스와 클래스 정보가 하나의 테이블에 저장되므로 인스턴스 크기가 증가함에 따라 클래스 검색 시간도 증가하게 된다.

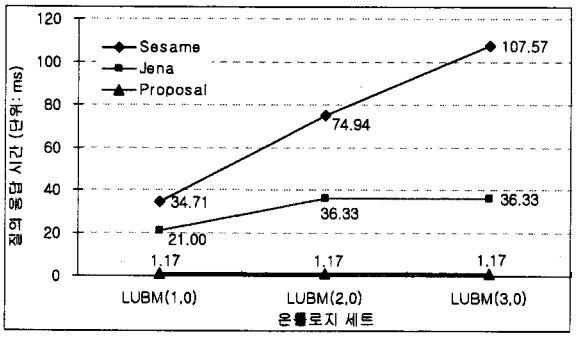

그림 4. Query1에 대한 실험 결과

그림 5는 Query2, 즉 특정 클래스에 해당하는 모든 인스턴스를 검색하는 질의에 대한 실험 결과로서 Jena 저장 시스템이 가장 낮은 성능을 보인다. 이는 Jena의 구조적인 특성에서 기인한 것으로, 동일한 테이블에서 클래스를 검색하고 다시 검색된 클래스를 동일한 크기의 테이블과 비교하기 때문이다. 그러나 제안한 시스템은 클래스와 인스턴스 정의 테이블이 각각 정의되어 있기 때문에 클래스 검색과 검색 클래스에 해당하는 인스턴스를 검색하는데 요구되는 비교 횟수가 가장 적기 때문에 빠른 성능을 제공한다.

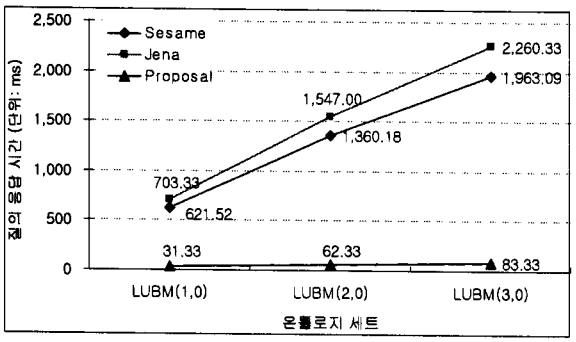

그림 5. Query2에 대한 실험 결과

그림 6은 클래스 계층 구조를 고려한 질의에 대한 실험 결과로서 특정 클래스의 모든 하위 클래스를 검색하는데 요구되는 질의 처리 시간을 보여준다. Query1에서와 같이 비교 시스템은 클래스 정보 검색을 위해 불필요한 다른 정보에 대한 비교 연산이 이루어진다. 이는 추가적인 오버헤드를 요구하며 높은 질의 처리 비용을 요구한다. 반면, 제안 시스템은 클래스 정보에 대한 비교 연산만이 이루어지며 클래스의 수는 온톨로지 크기와 무관하게 일정하므로 온톨로지 크기가 증가하더라도 동일한 연산 시간을 요구하게 된다.

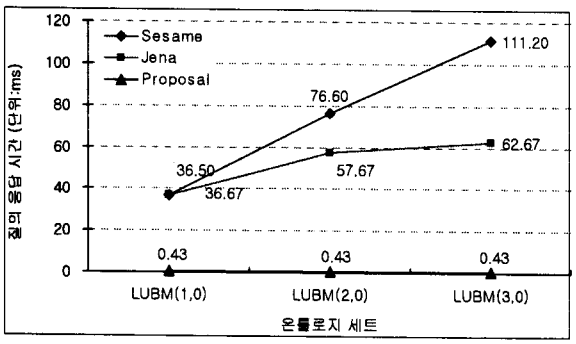

그림 6. Query3에 대한 실험 결과

그림 7은 특정 클래스의 하위 클래스를 검색하고 각 클래스에 해당하는 인스턴스를 검색하는 질의에 대한 실험 결과로서 제안 시스템이 가장 낮은 처리 시간을 요구함을 알 수 있다.

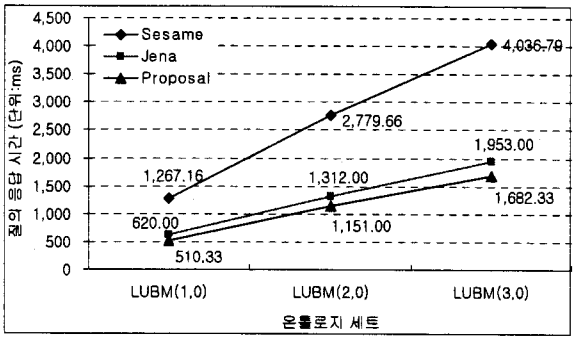

그림 7. Query4에 대한 실험 결과

주어진 실험 환경과 방법론 하에서, 제안 시스템이 정의된 모든 질의에 대해 나은 성능을 보였다. 그러나 보다 복잡하고 다양한 질의나 웹 온톨로지의 크기 혹은 특성에 따라 성능 비교 결과가 달라질 수 있을 것이다. 따라서 향후 이에 대한 연구가 추가적으로 이루어져야 할 것이다.

5. 결론

이 논문에서는 관계형 모델 기반의 웹 온톨로지 저장 시스템에 대한 실험 및 비교 평가 결과에 대하여 기술하였다. 비교 평가를 위한 평가 항목은 질의 처리 시간이며, 먼저 제안 시스템에 대하여 소개하고 실험을 위한 방법 및 실험 환경에 대하여 정의하였다. 또한 질의 응답 시간에 대한 평가를 위해 다양한 형태의 질의 패턴을 정의하였다.

실험 결과에서, 제안 시스템이 정의한 모든 질의 패턴에 대해 나은 성능을 보였다. 특히 단순 클래스 검색이나 클래스 계층을 고려한 클래스 정보 검색에 있어서는 동일한 검색 시간만을 요구하였다. 이는 클래스 정보만을 관리하는 저장소 모델의 특성에서 비롯된 것으로, 이러한 특성은 특정 클래스의 해당 인스턴스 검색을 위한 질의 처리 시간에도 많은 영향을 준다.

향후 연구로서, 이 논문에서는 네 개의 질의 패턴만을 정의하여 실험을 수행하였으나 보다 다양한 질의 패턴에 대한 실험 및 평가가 요구된다. 또한 질의 결과에 대한 정확성과 완전성 문제에 대한 평가 또한 이루어져야 할 것이다. 정확성과 완전성은 제안된 저장소가 정보에 대한 손실 여부, 질의 결과에 대한 안정성을 보여주는 중요한 평가 항목들이다.

참고문헌

[1] Tim Berners-Lee, James Hendler, and Ora Lassila, "The Semantic Web," Scientific American, May 2001.

[2] W3C, RDF/XML Syntax Specification, http://www.w3.org/TR/rdf-syntax-grammar, Feb. 2004.

[3] W3C, RDF Vocabulary Description Language 1.0: RDF Schema, http://www.w3.org/TR/rdf-schema, Feb. 2004.

[4] DAML+OIL Reference Description W3C Note, http://www.w3.org/TR/daml+oil-reference, Dec. 2001.

[5] W3C, OWL (Web Ontology Language), http://www.w3.org/2004/OWL/, 2007.

[6] Protege, http://protege.stanford.edu/, 2007.

[7] JENA2, http://jena.sourceforge.net/, 2007.

[8] SourceForge.net, "JENA2 Database Interface: Database Layout," Nov. 2004.

[9] SESAME, http://www.openrdf.org/, 2007.

[10] Jeen Broekstra, Arjohn Kampman, Frank van Harmelen, "SESAME: A Generic Architecture for Storing and Querying RDF and RDF Schema," Lecture Notes in Computer Science (LNCS), Vol. 2342, pp. 54-68, Jun. 2002.

[11] Pan, Z. and Heflin, J., "DLDB: Extending Relational Databases to Support Semantic Web Queries," In Workshop on Practical and Scaleable Semantic Web Systems, The 2nd International Semantic Web conference (ISWC2003), 2003.

[12] AKT (Advanced Knowledge Technologies), 3store, http://www.aktors.org/technologies/3store/, 2007.

[13] OWLJessKB: A Semantic Web Reasoning Tool, http://edge.cs.drexel.edu/assemblies/software/owljesskb/, 2007.

[14] Dongwon Jeong, Myounghoi Choi, Yang-Seung Jeon, Youn-Hee Han, Laurence T. Yang, Young-Sik Jeong, and Sung-Kook Han, "Persistent Storage System for Efficient Management of OWL Web Ontology," Springer-Verlag, Lecture Notes in Computer Science (LNCS),Vol. LNCS 4611, pp. 1089-1097, Jul. 2007.

[15] Dongwon Jeong, Myounghoi Choi, Yang-Seung Jeon, Youn-Hee Han, Young-Sik Jeong, and Sung-Kook Han, "A Novel Memory-Oriented OWL Storage System," Springer-Verlag, Lecture Notes in Computer Science (LNCS), Vol. LNCS 4331, pp. 542-549, Dec. 2006.

[16] SWAT Projects - the Lehigh University Benchmark (LUBM), Data Generator (UBA), http://swat.cse.lehigh.edu/projects/lubm/, 2007.