sparQL을 이용한 온톨로지 검색 구현

박재훈, 최종옥, 전양승, 정석태, 정영식, 한성국
원광대학교 컴퓨터공학과
{pjh98, cjomail, globaljeon, stjoung, ysjeong,
skhan}@wonkwang.ac.kr

Implementation of Ontology Search using sparQL

Jae-Hun Park, Jong-Ok Choi, Yang-Seung Jeon,
Suck-Tae Joung, Young-Sik Jeong, Sung-Kook Han
Dept of Computer Engineering, Wonkwang University

요 약

시맨틱 웹에서 지능형 검색을 위해 최적의 온톨로지 구축은 필수적이다. 온톨로지 언어인 OWL은 웹 온톨로지 언어로써 특히, OWL Lite의 경우 웹 응용에 많이 사용된다. OWL Lite로 구축된 온톨로지의 인디비절 검색은 sparQL이라는 쿼리 언어를 이용해 XML 형태의 결과로 반환해 활용의 폭을 넓혔다.

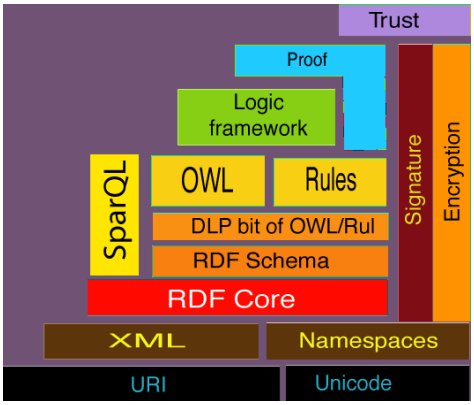
1. 서론

현재의 웹은 사람이 보고 잘 이해할 수 있도록 하기 위한 브라우저의 디스플레이 또는 레이아웃 기술에 초점을 맞추고 있다. HTML을 이용한 이러한 표현방식은 문서의 내용과 의미를 나타내는 시맨틱 정보를 표현하기가 어려우며, 따라서 사람이 아닌 프로그램 또는 소프트웨어 에이전트가 자동으로 문서로부터 의미를 추출하기가 어렵다.

이러한 문제점을 해결하기 위해서 1990년대 말에 W3C(World Wide Web Consortium)에서 시맨틱 웹(Semantic Web)을 제안하였다.

Tim Berners Lee에 의해 제안된 시맨틱 웹은 컴퓨터가 웹상의 정보를 이해하고, 정보를 창출할 수 있는 웹 환경으로서, 정보의 탐색과 의사결정이 인간이 아닌 컴퓨터가 할 수 있도록 만들어진 Web 환경이다. 즉, 메타데이터의 개념을 통하여 웹 문서에 시맨틱 정보를 덧붙이고, 이를 이용하여 에이전트가 의미 정보를 자동으로 추출할 수 있는 패러다임을 조성하는 것이다. 부수적으로 의미 정보의 자동 추출뿐 아니라 정보의 확장이나 공유 등도 가능하다.

시맨틱 웹의 목적은 웹에 있는 정보를 컴퓨터가 쉽게 이해 할 수 있도록 도와주는 표준과 기술을 개발하여 시맨틱 검색, 데이터 통합, 네비게이션, 업무의 자동화 등을 지원하는 것이다. 시맨틱 웹에서 이러한 기능을 지원하기 위해서는 컴퓨터의 지능적인 정보처리가 가능토록 웹 문서 내에 지식 표현을 위한 온톨로지를 삽입하고, 지식 간의 관계를 설정하며 추론 규칙을 포함 시켜야한다. 이를 통해서 사용자가 원하는 정보를 정확하게 전달해 줄 수가 있다.



(그림 1) 2005년판 시맨틱 웹 구조[6]

2. 온톨로지

시맨틱 웹에서의 검색을 위해 온톨로지는 가장 중요한 요소이다. 온톨로지의 개념을 이해하고 다양한 온톨로지 언어를 이용해 온톨로지의 설계 및 구축 작업이 선행되어야 시맨틱 웹 검색이 가능하다[4].

2.1 온톨로지 개념

온톨로지는 간단히 표현하면 단어와 관계들로 구성된 사전으로서 어느 특정 도메인에 관련된 단어들을 계층적 구조로 표현하고 추가적으로 이를 확장할 수 있는 추론 규칙을 포함한다. 그리고 웹 기반의 지식 처리나 응용 프로그램 사이의 지식 공유, 재사용을 가능하게 하는 중요한 요소이다.

온톨로지에는 계층분류와 추론규칙에 대한 정의가 포함된다. 계층분류는 객체이 클래스와 서브클래스, 그들간의 관계를 정의한다. 그리고 온톨로지를 표현하기 위해 스키마와 구문구조 등을 정의한 언어가 온톨로지 언어이다[2][4][5].

2.2 온톨로지 언어

▪ RDF[1]

특정 자원에 대한 메타 데이터를 기술하는 XML 기반의 프레임워크이다. RDF는 레코드(record)를 하나의 기술 단위로 취급해온 기존의 방식과는 달리 자원, 속성, 속성 값을 하나의 단위로 취급하는 이른바 “Triple” 개념이 그 핵심이다. 자원 속성 표현의 세분화로 인해 자원에 대한 좀 더 정교한 기술이 가능해지고, 자원들 간의 관계 설정이 속성(Predicate)를 통해 무한으로 가능하게 되어진다. 각각의 자원들은 URI를 통해 고유 식별자를 가지게 된다. 그리고 자원을 기술하는 속성 명 또한 고유한 URI를 통해 XML Namespace에 정의되어진 속성을 사용함으로써 상호간 의미 충돌을 막는다. 속성의 값으로는 다른 URI가 지정될 수 있으며, 속성 값으로 지정된 자원 역시 다시 기술의 대상이 되기 때문에 그 자원에 대한 속성과 속성 값을 다시 부과할 수 있다.

Subject (Resource)	http://www.w3.org/Home/Lassila
Predicate (Property)	Creator
Object (literal)	"Ora Lassila"

(그림 2) RDF의 Triple 구조

▪ TopicMap

TopicMap은 ISO를 중심으로 한 Semantic Web 기술로 ISO/IEC 13250 표준으로 지식 표현 기술(Knowledge Representation)의 표준이다. RDF와 마찬가지로 XML 기반의 표준 기술 언어인 XTM(XML Topic Maps)라는 언어를 사용하여 정보와 지식의 분산 관리를 지원한다. 이는 정보 자원의 구성, 추출, 네비게이션에 관계하는 새로운 패러다임으로, 정보와 지식 관리를 위해 최적화된 표현 양식을 제공하고 있다. Topic Maps는 지식층과 정보층의 이중 구조를 나타내는데, 지식층은 상위 계층으로 토픽(topic)과 토픽간의 연계(Association)로 구성된다. 토픽은 특정 주제를 나타내는 표현이고, 연계는 주제들 간의 관계를 나타내는 표현이다. 정보층은 디지털 컨텐츠를 나타내며, 이들 지식층과 정보층은 어커런스(Occurrence)를 통해 상호 연결이 이루어진다.

▪ OWL

OWL은 표현력에서는 우수하지만 처음 접근하는 이용자, 개발자 및 개발도구 지원 등에 용이하지 않은 측면이 있어 이를 개선하여 보편적인 이용자를 확보라기 위해 좀더 간결하고 사용하기 쉬운 언어가 필요하게 되었다. 이러한 목적으로 개발된 DAML+OIL의 새로운 개정 언어이다. 또한 OWL의 서브셋으로 제안된 OWL Lite는 OWL과 DAML+OIL의 공통적이고 유용한 부분들을 간추려 만들어졌다. 기능적인 면에서 볼 때 웹 응용프로그램을 지원하기 위해 간단하면서도 RDFS에 비해 풍부한 표현력을 가지는 언어이다.

OWL에서 제안하고 있는 유형은 OWL Lite, OWL DL, OWL Full이 있는데 OWL Lite는 시소러스에 접근이 용이하고 단순한 면을 강조하여 웹 응용에 강점을 갖고 있으며 OWL DL은 Lite 보다 좀 더 논리적인 표현을 위한 온톨로지이다. 마지막으로 OWL Full은 표현력에 있어서 가장 풍부하며 RDF의 자유로운 구문을 모두 허용하고 있다.

즉, OWL Full은 DL과 Lite의 모든 기능을 포함하는 관계이며 유효성 및 호환성에 있어 가장 완벽하지만 웹 온톨로지 언어로써 이용하기에 Lite가 용이하며 시소러스의 개념에 접근하기 쉽다는 점에서 웹 응용에 많이 사용된다.

```
<owl:Ontology rdf:about=""/>
<owl:Class rdf:ID="Family">
 <rdfs:subClassOf>
  <owl:Class rdf:ID="Movie"/>
 </rdfs:subClassOf>
 <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
 >가족 영화에 대한 정보를 표현한다.</rdfs:comment>
</owl:Class>
<owl:Class rdf:ID="Animation">
 <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
 >애니메이션에 대한 정보를 표현한다.</rdfs:comment>
 <rdfs:subClassOf>
  <owl:Class rdf:about="#Movie"/>
 </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="OST">
 <rdfs:subClassOf>
  <owl:Class rdf:ID="Music"/>
 </rdfs:subClassOf>
 <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
```

(그림 3) OWL 문서

2.3 온톨로지 구축

온톨로지의 구축 단계를 간략하게 설명하면, 특정한 목적과 영역을 고려한 다음, 개념을 자동·반자동 추출하거나 어휘 사전을 확보하여, 확보된 개념들을 정의하고 조직화해야 한다. 조직화한다는 것은 개념들 간의 관계를 설정함과 동시에 개념이 가지는 특수한 속성을 추출하여 체계화시키는 것을 의미하는 것으로서, 이론적 체계와 더불어 실질적인 구축 원리를 마련해야 한다. 다음 단계로 온톨로지를 표현할 웹 온톨로지 언어나 기계 가독형 표현 언어(machine readable representation language)를 설정하여 형식화하고 실질적으로 구현하여야 하며, 구축 중인 온톨로지와의 다른 온톨로지와의 통합 문제와 기존에 존재하는 많은 자원(resources)을 어떻게 활용할 것인가를 모색해야 한다. 그리고 마지막으로 구축된 온톨로지를 대상 애플리케이션(target application)에서 실험하거나 사용 패턴(usage patterns)을 분석하여 평가해야 한다. 또한 이러한 평가 결과를 바탕으로 유지·보수를 해야 하며, 조금 더 발전적인 온톨로지로 개선해야 한다.

이러한 일련의 구축 단계에서 가장 어려운 문제는 바로 온톨로지를 실질적으로 구축하는 이론적 체계와 원리가 아직까지 마련되지 않았다는 것이다. 기존의 구축 사례들을 살펴보면, 온톨로지의 실질적인 구축 측면보다도 기존의 시소러스나 의미망, 분류체계 등을 이용한 온톨로지 구축이나 기구축된 온톨로지를 이용한 애플리케이션 개발이 대부분을 차지한다. 이것은 온톨로지 구축에 있어서 국내외적으로 가지는 공통적인 문제라 할 수 있다. 이를 위해서는 WordNet, UMLS와 같이 관련 학문에 대한 이론 습득과 더불어 자연언어처리 기법의 활용을 통한 언어 습득 및 이해 처리 등과 같은 부수적인 연구가 뒤따라야 할 것이다.

3. 온톨로지 기반 검색

온톨로지 기반 정보검색 기술은 중요한 정보가 있는 자원을 빠르게 찾아 사용할 수 있다는 점과 자원을 찾는 정확도를 향상시킬 수 있다는 점에서 중요한 기술로 자리 잡아 가고 있다. 또한 검색엔진이 온톨로지에 정의된 개념과 규칙을 활용하면서, 온톨로지의 검색 향상을 위해 추론 규칙을 이용하기 때문에, 단순히 사용자의 질의와 일치되는 문서만 보여주는 것이 아니라 사용자의 질의의 의미를 분석하여 그와 관련된 정보를 온톨로지에 표현된 관계에 따라 다시 질의를 적절하게 바꿀 수도 있게 한다[3].

3.1 온톨로지 검색 쿼리 sparQL

sparQL(SPARQL Protocol And RDF Query Language)은 RDF 그래프로부터 정보를 얻을 수 있는 쿼리 언어로 그래프 패턴 매칭을 기반으로 한다. 가장 단순한 패턴은 Triple 패턴으로 RDF triple을 예로 들 수 있다. RDF triple이 아니더라도 주어(Subject), 서술어(Predicate), 목적어(Object)로 구성된 유효한 값만 있다면 가능하다.

```
PREFIX base: <http://www.owl-ontologies.com/unnamed.owl#>
SELECT ?birthCountry ?age
FROM <http://www.owl-ontologies.com/unnamed.owl#>
WHERE
{       ?Actor base:birthCountry ?birthCountry.
        ?Actor base:age ?age.           }
```

(그림 4) sparQL Syntax

그림 4는 sparQL 예제로 PREFIX의 base에 해당 온톨로지의 네임스페이스를 정의한다. 쿼리 문법은 다른 쿼리 언어와 비슷한 문법이지만 FROM 절에는 테이블 명 대신에 온톨로지의 네임스페이스를 입력한다. 그리고 WHERE 절에는 클래스의 인디비절을 조건으로 입력해 SELECT에 해당하는 값을 조회한다. 조건문의 ?Actor는 클래스를 의미하고 base:age는 base라는 네임스페이스의 온톨로지에서 age라는 인디비절을 의미한다. 그리고 ?age는 SELECT 절의 ?age와 매칭되어야 한다.

4. 온톨로지 검색 구현

온톨로지 검색의 구현을 위해 JENA API를 사용했다. 기본적으로 JENA API에는 sparQL 처리를 위한 라이브러리가 포함되어 있지 않으므로 ARQ API를 별도로 추가해야 한다. 온톨로지의 구축 또한 JENA API를 이용해 데이터베이스에 임포트하는 방법을 사용했다.

온톨로지를 데이터베이스에 임포트하는 이유는 대

용량의 온톨로지가 파일로 존재할 경우 사용자가 파일에 직접 접근해 인디비절을 검색할 경우 상당히 퍼포먼스가 떨어진다. 이를 해결하기 위해 JENA API는 온톨로지 파일의 데이터베이스 임포트를 지원해 온톨로지 접근 및 검색에 있어 뛰어난 퍼포먼스를 제공한다.

```
    private static String queryString =
        "   PREFIX base: <http://www.owl-ontologies.com/unnamed.owl#> " +
        "   SELECT ?birthCountry ?age       " +
        "   FROM <http://www.owl-ontologies.com/unnamed.owl#> " +
        "   WHERE " +
        "   (    ?Actor base:birthCountry ?birthCountry. " +
        "        ?Actor base:age ?age.) " ;

    public static void main(String[] args) {
        Sparql inferenceinterface = new Sparql();

        // 추론 interface의 getSPARQLResult 메소드 호출
        ResultSet resultset = inferenceinterface.getSPARQLResult(queryString);

        // ResultSetFormatter에서 필요
        Query query = QueryFactory.create(queryString);

/*  ###########################################################################
        @ return type = xml
        @ note : xml 출력시 사용 예
       ###########################################################################*/
       ByteArrayOutputStream out = new ByteArrayOutputStream();
       ResultSetFormatter rsFmt = new ResultSetFormatter(resultset, query);
       rsFmt.outputAsXML(out);
       String xml = out.toString();
       System.out.println(xml);


       /*   ###########################################################################
          @ return type = text
          @ note : text 출력시 사용 예
       ###########################################################################*/
//          ResultSetFormatter.out(System.out, resultset, query);

       //     QueryExecution objects should be closed to free any system resources
       inferenceinterface.qeClose();
    }
```

(그림 5) sparQL 쿼리 처리 소스

그림 5는 sparQL 쿼리를 이용해 온톨로지에서 쿼리 결과를 XML 형태로 출력해주는 프로그램이다. sparQL 쿼리를 저장하고 있는 queryString 변수는 JENA API에서 제공하는 ResultSet과 Query 타입의 클래스로 변환되어 쿼리 요청자에게 XML 형태의 파일로 결과를 반환한다.

- sparQL Sample Query

sparQL 문장

```
PREFIX base: <http://www.owl-ontologies.com/unnamed.owl#>
SELECT ?birthCountry ?age
FROM <http://www.owl-ontologies.com/unnamed.owl#>
WHERE
{   ?Actor base:birthCountry ?birthCountry.
    ?Actor base:age ?age.}
```

- sparQL Sample Query Result

sparQL 결과

```
<?xml version="1.0"?>
<sparql
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
  xmlns="http://www.w3.org/2001/sw/DataAccess/rf1/result" >
  <head>
    <variable name="birthCountry"/>
    <variable name="age"/>
  </head>
  <results>
    <result>
      <birthCountry
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">????</birthCountry>
      <age rdf:datatype="http://www.w3.org/2001/XMLSchema#int">34</age>
    </result>
    <result>
      <birthCountry
rdf:datatype="http://www.w3.org/2001/XMLSchema#string">????</birthCountry>
      <age rdf:datatype="http://www.w3.org/2001/XMLSchema#int">26</age>
    </result>
```

(그림 6) XML 형태의 sparQL 실행 결과

5. 결론

시맨틱 웹의 가장 큰 이슈는 지능적이라는 것이다. 기존의 HTML 웹은 단순한 링크로 이루어진 반면 시맨틱 웹은 모든 정보들이 지능적으로 관계를 맺고 있어 보다 정확한 검색이 가능하다. 이를 위해 온톨로지의 구축은 아주 중요하다. 웹 기반의 응용 프로그램의 경우 OWL Lite가 적합하다. 시소러스의 개념에 접근하기 쉽고 단순하기 때문이다. OWL Lite로 작성된 온톨로지는 퍼포먼스 극대화를 위해 JENA API를 이용해 데이터베이스로 임포트 한 후 sparQL 쿼리를 이용해 온톨로지의 인디비절을 검색한다. 하지만 현재 구현된 프로그램은 sparQL 쿼리의 문법을 이해한 후 쿼리를 직접 작성해야 하는 단점이 있어 제한적이다. 그리고 JSP 페이지로 웹 상에서 구동되고 있지만 웹 서비스를 적용하면 활용도가 더욱 높아질 것으로 전망된다.

감사의 글

본 연구는 학술진흥재단 기초과학연구사업 중 지방 연구중심대학 육성사업인 헬스케어 기술개발사업단의 지원에 의해 수행되었으며 이에 감사드립니다.

6. 참고문헌

[1] 이재호, “시맨틱 웹의 온톨로지 언어”, 정보과학회지, 제21권, 제3호, pp. 18-27, 2003

[2] 양정진, “시맨틱 웹에서의 온톨로지 공학”, 정보과학회지, 제21권, 제3호, pp.28-35, 2003

[3] 장명길 외, “의미기반 정보검색”, 정보과학회지, 제19권, 제10호, pp. 7-18, 2001

[4] 옥철영, “한국어정보처리와 온톨로지”, 2004 한국어정보처리연구회 동계 튜토리얼 자료집

[5] 최호섭, 옥철영, 김창환, 왕지현, 장명길, “질의응답시스템을 위한 백과사전 기반 지식베이스와 온톨로지”, 제15회 한글 및 한국어 정보처리학술대회 자료집, pp. 177-183, 2003

[7] 김현희, 안태경, “온톨로지를 이용한 인터넷웹 검색에 관한 실험적 연구”, 정보관리학회지, 제20권, 제1호, pp. 417-455, 2003

[8] 정도헌, “시맨틱웹을 위한 온톨로지 언어와 구현 사례 연구”, 정보관리연구, 제34권, 제3호, pp. 87-109, 2003

[6] Berners-Lee, T., Hendler, J., Lassila, O., "The Semantic Web", Scientific American, 2001