한국지능시스템학회 논문지 2008, Vol. 18, No. 1, pp. 127-132

A Natural Language Query Framework for the Semantic Web

Jin Sung Kim

School of Business Administration, Jeonju University
1200 Hyoja-Dong 3Ga, Wansan-Ku, Jeonju, Jeonbuk 560-759, Korea

Abstract

This study proposes a Natural Language Query Framework (NLQF) for the semantic web. It supports an intelligent inference at a semantic level. Most of previous researches focused on the knowledge representation on the semantic web. However, to revitalize the intelligent e-business on the semantic web, there is a need for semantic level inference to the web information. To satisfy the need, we will review the knowledge/resource representation on the semantic web such as RDF, Ontology and Conceptual Graph (CG), and then discuss about the natural language (NL) inference. The result of this research could support a natural interface for the semantic web. Furthermore, we expect that the NLQF can be used in the semantic web-based business communications.

Keywords: Conceptual graph, Intelligent agent, Natural language, Ontology, RDF, Semantic web

1. Introduction

The web-based knowledge representation and gathering is increasingly moving to the semantic web. According to the improvement of the technology in knowledge description on the semantic web, a great number of web information resources are also available. During the development of web-based intelligent knowledge gathering, the most important work is a description of the information resources [3]. Where the information is annotated with concepts from sharing ontology, thus the semantics of web information can be understood and consumed by agents [1]. The agents include the human agents and intelligent agents (IAs). To help them effective annotation mechanisms were developed by previous researchers. Among them, RDF and OWL-related languages are well known to semantic web developers. However, most of previous researches still have focused on how to annotate the HTML information efficiently.

In contrast with these improvements, there were few researches in semantic web-based knowledge inference. First of all, to infer the knowledge on the semantic web, it is need to construct and translate a RDF or OWL web document. To construct a RDF or OWL-oriented web document, an ontology and schema should be ready to describe the web resources explicitly. Where the use of a sharable ontology and schema for the explication of implicit knowledge is a possible approach to overcome the limitations came out from semantic heterogeneity. Therefore, with respect to the description of information resources on the semantic web, they can be used for the identification and association of semantically corresponding information concepts [2]. However, a barrier still remain which protect the intelligent inference on the semantic web.

To overcome the limitations, in this study, we introduce the RDF, OWL, NL-oriented knowledge transformation, and a semantic web-based NL inference framework. The rest of the paper is organized as follows. Section 2 reviews the related works on RDF, Ontology, CG, and OWL. In Section 3, we propose a NLQF. The concluding remarks are presented in Section 4.

접수일자 : 2007 년 4 월 25 일
완료일자 : 2007 년 12 월 10 일

2. Research Background

There are several approaches to explain the semantics on the web such as Ontology, DTD, RDF, XML Schema, and etc. To clarify the purpose of this study, only the RDF, Ontology, CG, and OWL will be introduced.

2.1. RDF

RDF developed by W3C organizes information in a *Resource-Property-Value* (RPV) or *Subject-Verb-Object* (SVO) triples form, thus the RDF files can be processed semantically on the semantic web [1, 4].

- A *Resource* is an entity accessible by an URI on the Semantic Web (e.g. an XML document). So the *Resources* are the elements described by RDF statements.
- A *Property* defines a binary relation between *resources* and/or its atomic values. A *property* enables us to attach detailed information to *resources*, and provide descriptions for *resources*.

- A *Value* can be either a character string or a *resource*. Reification of *resources* using *property* and *values* enables us to transform the tripe into a *resource*.

RDF is a general purpose knowledge representation framework and RDF-Schema (RDFS) is the RDF vocabulary description language to be used to provide a further description mechanism to define classes or groups of related resources and the relationships between the resources [3]. Some primitives are defined in RDFS and there are several successors of RDFS such as Simple HTML Ontology Extensions (SHOE), DARPA Agent Markup Language (DAML), DAML-ONT, Ontology Inference Layer (OIL), DAML+OIL, and OWL. The OWL has three increasingly-expressive sublanguages OWL Lite, OWL Description Logic (DL) and OWL Full [8]. The RDF structure is embedded in these languages [1]. Therefore, most of semantic web applications are focused on the RDF and RDFS-oriented annotations.

2.2. Ontology

As one of researches on the semantic web ontology, Li & Liang's (2006) ontology hierarchy will be introduced in this section. The contribution of their research is that a genetic model for organizing several ontologies. The proposed genetic model could organize several different ontologies on the semantic web. Especially, the genetic model has four operators *Inheritance*, *Block*, *Atavism* and *Mutation*. Each one of ontologies, in Figure 1, could reuse the primitives of ontology languages e.g. RDF, RDFS and OWL based on the *Inheritance* operator. And lower level ontology reuse the concepts of higher level ontologies. For example, Employee Ontology inherits Person Ontology directly. However, the concept home_phone of Person Ontology is blocked by Employee Ontology since the home phone of an employee should not be public. The concept contact_number of Employee Ontology (refers to office_phone) is a mutation of the contact_number in Person Ontology (refers to home_phone).

2.3. CG

CG is a method of knowledge representation developed by Sowa [5] based on Charles Peirce's Existential Graphs and Semantic Networks of AI [7]. CG has a direct mapping to and from NL and a graphic notation designed for human readability. Therefore, CG could express the meanings in a form that is logically precise, humanly readable, and computationally tractable. Many popular graphic notations and structures ranging from type hierarchies to entity-relationship or state transition diagrams can be viewed as special cases of CGs [7].

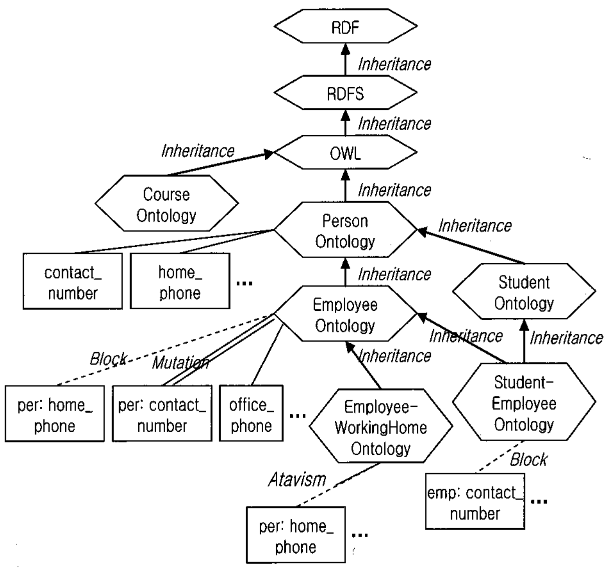


Figure 1 Ontology hierarchy (Li & Ling, 2005)

On the CG, the knowledge is divided into two parts *Terminological Knowledge* and *Assertional Knowledge*. Fist, the *Terminological Knowledge* (*Support*) which contains the 'ground' vocabulary and the *Assertional Knowledge* which consist of a set of CGs built by means of the *Terminological Knowledge* [6]. The *Support* provides the ground vocabulary used to build the domain knowledge base: the type of *Concepts* used, the *Instances* of these types, and the types of *Relations* (*Conceptual Relationships*) linking the concepts. The concepts can be linked by means of relations and the support contains the set of *Conceptual Relation Types*. Kayed & Colomb (2005) used CG formalism to build up some ontologies. They explained that the using CGs facilitates deploying XMLs capabilities and advantages by two ways [7]:

- By embedding CGs in Web-XML documents to represent knowledge.
- By building a browser-based XML like language that can communicate with a collection of ontological components that use Conceptual Graph Interchange Format (CGIF) as their native language.

It means that the CG and CGIF could represent a logical knowledge and their relationship on the semantic web. Then the knowledge could be delivered to the remote knowledge user whom he need that knowledge since the CG and CGIF could support the communication with ontological components. Therefore, it could be used in the retrieving, delivering and inference of knowledge on the semantic web.

2.4. OWL

A set of XML statements by itself does not allow the web users to reach a certain conclusion about any other XML statements. In order to employ XML to generate the conclusions, we need a knowledge embedded in some procedural code [8]. In this case, a set of OWL statements by itself can allow us to reach a conclusion about another OWL statement [8]. OWL facilitates greater machine readability of web content than that supported by XML, RDF, and RDFS by providing additional vocabulary along with a formal semantics. An OWL statement looks very much like XML and RDF documents in terms of *elements*, *tags*, and *namespaces*. OWL statement starts with a *header* and then defines *properties* and *classes*. The first part of OWL statement is the outer OWL block, delimited by *owl:Ontology* containing version information *owl:versionInfo*, and an import section *owl:imports*. Table 1 shows an example OWL header.

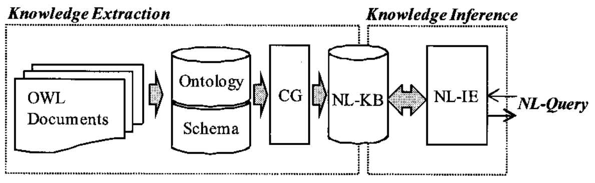
Table 1 An Example of OWL header

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF
  xmlns:owl = "http://www.w3.org/2002/07/owl#"
  xmlns:rdf = "http://www.w3.org/1999/02/22-rdf-syntax-
ns#"
xmlns:rdfs =" http://www.w3.org/2000/01/rdf-schema#"
xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:xsd = "http://www.w3.org/2000/1/XMLSchema#">
  <owl:Ontology rdf:about = " ">
    <owl:versionInfo>1.0 </owl:versionInfo>
    <dc:title>e-Book: Introducing Semantic Web
Ontology</dc:title>
    <dc:author>Jin S. Kim</dc:author>
    <dc:date>Sep. 19 2007</dc:date>
  </owl:Ontology>
</rdf:RDF>
```

The namespaces shown above are similar with standards as well as the outer rdf:RDF opening and elements. However, a schema for the existing RDF vocabulary was not defined. Rather, the RDF and XML were used and new ontology was constructed. Where the Dublin Core element <dc> was designed to document metadata about published resources. Then <dc> elements were included in the header of OWL document to define title, author, date, and other information since the ontology was a published resource [8].

3. Research Methodology

The proposed NLQF is consists of two phases *Knowledge Extraction* and *Knowledge Inference*. Figure 2 shows the whole structure of the NLQF. Where, Schema and Ontology are used to explicit the knowledge and mapping with other knowledge represented on the semantic web. The NL-KB is used to store and retrieve the knowledge extracted from the semantic web.



NL-IE: NL Inference Engine
NL-KB: NL Knowledge Base

Figure 2 The NLQF

3.1. Knowledge Extraction

Web information represented by OWL contains several kind of knowledge. Traditional XML Schema, XML and RDFS messages focused on the specific data but it is a knowledge representation rather than a message format. First, in the NLQF procedure, the OWL-based web documents (information) are translated into the CG. After the translation, the knowledge and their relationships stored in CG should be transformed as a form of knowledge which could be used in NL inference. The NL-KB has the knowledge. For example PROLOG+CG Artificial Intelligence (AI) language can be adapted directly to that procedure. Table 2 shows the OWL sentences on the web.

Table 2 OWL ontology for wine

```
<owl:Class rdf:ID="Wine">
  <rdfs:subClassOf rdf:resource="http://www.w3.org/TR/2003/CR-
owl-guide-20030818/food#PotableLiquid" />
<rdfs:subClassOf>
  <owl:Restriction>
  <owl:onProperty rdf:resource="#hasMaker" />
                                        <owl:cardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInte
ger">1</owl:cardinality>
  </owl:Restriction>
  </rdfs:subClassOf>
<rdfs:subClassOf>
  <owl:Restriction>
  <owl:onProperty rdf:resource="#hasMaker" />
  <owl:allValuesFrom rdf:resource="#Winery" />
  </owl:Restriction>
</rdfs:subClassOf>
...
<owl:ObjectProperty rdf:ID="hasColor">
```

```
                                                    <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty" />
  <rdfs:subPropertyOf rdf:resource="#hasWineDescriptor" />
  <rdfs:domain rdf:resource="#Wine" />
  <rdfs:range rdf:resource="#WineColor" />
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="hasMaker">
                                                    <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty" />
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="producesWine">
  <owl:inverseOf rdf:resource="#hasMaker" />
</owl:ObjectProperty>
...
<owl:Class rdf:ID="WhiteLoire">
<owl:intersectionOf rdf:parseType="Collection">
   <owl:Class rdf:about="#Loire" />
   <owl:Class rdf:about="#WhiteWine" />
   </owl:intersectionOf>
   </owl:Class>
<owl:Class rdf:about="#WhiteLoire">
<rdfs:subClassOf>
<owl:Restriction>
<owl:onProperty rdf:resource="#madeFromGrape" />
<owl:allValuesFrom>
<owl:Class>
<owl:oneOf rdf:parseType="Collection">
   <owl:Thing rdf:about="#CheninBlancGrape" />
   <owl:Thing rdf:about="#PinotBlancGrape" />
   <owl:Thing rdf:about="#SauvignonBlancGrape" />
   </owl:oneOf>
   </owl:Class>
   </owl:allValuesFrom>
   </owl:Restriction>
   </rdfs:subClassOf>
   </owl:Class>
<owl:Class rdf:ID="WhiteBurgundy">
<owl:intersectionOf rdf:parseType="Collection">
   <owl:Class rdf:about="#Burgundy" />
   <owl:Class rdf:about="#WhiteWine" />
   </owl:intersectionOf>
   </owl:Class>
<owl:Class rdf:about="#WhiteBurgundy">
<rdfs:subClassOf>
<owl:Restriction>
<owl:onProperty rdf:resource="#madeFromGrape" />
<owl:hasValue rdf:resource="#ChardonnayGrape" />
   </owl:Restriction>
   </rdfs:subClassOf>
<rdfs:subClassOf>
<owl:Restriction>
<owl:onProperty rdf:resource="#madeFromGrape" />
   <owl:maxCardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInte
ger">1</owl:maxCardinality>
   </owl:Restriction>
   </rdfs:subClassOf>
</owl:Class>
...
```

The above web ontology is transformed into a form of CG. In this study we used the PROLOG+CG tools to manage the CG and NL inference. Table 3 shows the example of CG transformed.

Table 3 The example of transformed web ontology

```
Universal > Wine, hasBody, hasColor, hasFlavor, hasMaker, hasSugar,
Property.
Attribute > WineBody, WineColor, WineFlavor, Winery, WineSugar.
hasBody > WineBody.
hasColor > WineColor.
hasFlavor > WineFlavor.
hasMaker > Winery.
hasSugar > WineSugar.
...
Wine > WhiteWine, RedWine, TableWine.
TableWine > WhiteTableWine, RedTableWine.
WhiteWine > WhiteLoire, WhiteBurgundy, WhiteBordeaux,
SauvignonBlanc.
RedWine > RedBurgundy, RedBordeaux.
...
WhiteBurgundy = CortonMontrachetWhiteBurgundy,
PulignyMontrachetWhiteBurgundy.
SauvignonBlanc = CorbansSauvignonBlanc,
CorbansPrivateBinSauvignonBlanc.
...
RedTableWine = MariettaOldVinesRed, MariettaPetiteSyrah,
MariettaZinfandel.
WineBody = Light, Medium, Full.
WineColor = White, Rose, Red.
WineFlavor = Delicate, Moderate, Strong.
Winery = Bancroft, Beringer, ChateauChevalBlanc, ChateauDYchem,
ChateauDeMeursault.
WineSugar = Sweet, OffDry, Dry.
...
[WhiteWine : x]<-MEMB-[hasBody : Light] :-
  [Wine : x]<-AGNT-[Property]-hasBody->[WineBody: Light].
[WhiteWine : x]<-MEMB-[hasBody : Medium] :-
  [Wine : x]<-AGNT-[Property]-hasBody->[WineBody: Medium].
[WhiteWine : x]<-MEMB-[hasBody : Full] :-
  [Wine : x]<-AGNT-[Property]-hasBody->[WineBody: Full].
```

```
[WhiteWine : x]<-MEMB-[hasFlavor : Strong] :-
  [Wine : x]<-AGNT-[Property]-hasFlavor->[WineFlavor: Strong].

[RedWine : x]<-MEMB-[hasBody : Light] :-
  [Wine : x]<-AGNT-[Property]-hasBody->[WineBody: Light].
[RedWine : x]<-MEMB-[hasBody : Medium] :-
  [Wine : x]<-AGNT-[Property]-hasBody->[WineBody: Medium].
[RedWine : x]<-MEMB-[hasBody : Full] :-
  [Wine : x]<-AGNT-[Property]-hasBody->[WineBody: Full].
...
[Wine : x] -
  - hasBody->[WineBody],
  - hasColor->[WineColor],
  - hasFlavor->[WineFlavor],
  - hasMaker->[Winery],
  - hasSugar->[WineSugar].
...
[Wine: CortonMontrachetWhiteBurgundy]<-AGNT-[Property]-
hasBody->[WineBody: Light].

[Wine: PulignyMontrachetWhiteBurgundy]<-AGNT-[Property]-
hasBody->[WineBody: Medium].

[Wine: CorbansPrivateBinSauvignonBlanc]<-AGNT-[Property]-
hasFlavor->[WineFlavor: Strong].

[Wine: CortonMontrachetWhiteBurgundy]<-AGNT-[Property]-
hasFlavor->[WineFlavor: Strong].
```

3.2. Knowledge Inference

NL was used as an efficient tool to manage the complicate human knowledge manipulation and inference. At the same purpose, semantic network was also used to represent the expert knowledge and their relationships logically. Therefore, we can adapt the NL inference if we could extract and transform the knowledge stored on semantic web into a NL-KB form. The revised versions of NL tools which have a NL-IE are applicable to this procedure. Then the results inferred by NL-IE should be represented as a human readable format on the semantic web. Furthermore, the agents on the semantic web will comprehend the information and can react to that. In this procedure, the knowledge interpreted by NL-IE is compared with the user's input NL Query. The capability of knowledge inference is rely on the NL-IE. Table 4 shows the NL queries and their inference results.

Table 4 The NL queries and their inference result

Query Description:
Show me the "White Wines (WhiteWine: x)" which have 'Strong' flavor (hasFlavor: Strong).

```
?- [WhiteWine: x]<-MEMB-[hasFlavor: Strong].

{x = CorbansPrivateBinSauvignonBlanc}
{x = CortonMontrachetWhiteBurgundy}
```

Answer Description: *The names of "White Wine" which have 'Strong' flavor are 'CorbansPrivateBinSauvignonBlanc' and 'CortonMontrachetWhiteBurgundy'*

4. Concluding Remarks

As a natural inference mechanism using on the semantic web, in this study, we proposed a NLQF. The proposed NLQF is consists of two phases *Knowledge Extraction* and *Knowledge Inference* to extract and infer the knowledge stored in OWL document. To infer the knowledge on the semantic web, schema and ontology are adapted firstly. Then the extracted knowledge is transformed into CGs which can manipulate expert knowledge and their conceptual relationships. The knowledge represented by CGs is stored in NL-KB as a form of NL. Then the NL-KB was used to answer for the user's request composed by NL query. The important contributions of this study could be summarized as follows.

- First, the NLQF support a knowledge extraction from the semantic web documents.
- Second, the combination of NL inference and OWL documents could support a natural inference on the semantic web.

However, the implementation process showed a simple prototype for NLQF. Therefore, it has many limitations to adapt it to the real-world problem. To cover the limitations, in the further research, the detailed and efficient inference procedures should be developed.

References

[1] C. Li and T. W. Ling, From XML to Semantic Web, *Database Systems for Advanced Applications*, Berlin: Springer, *The 10th International Conference on Database Systems for Advanced Applications* (DASFAA 2005), LNCS 3453, pp. 582-587, Beijing, China, 17-20 April 2005.
[2] J. Gu, B. Hu, and Y. Zhou, Semantic Query Planning Mechanism on XML Based Web Information Systems, *The 7th International Conference on Web Information Systems Engineering* (WISE 2006), LNCS 4256, pp. 194-205, Wuhan, China, 2006.
[3] H. Yao and L. Etzkorn, L., Automated conversion between

different knowledge representation formats, *Knowledge-Based Systems*, 19, pp. 404-412, 2006.

[4] O. Corby, R. Dieng, and C. Hébert, A conceptual graph model for W3C resource description framework, *Proceedings of the 8th International Conference on Conceptual Structures (ICCS '2000)*, Berlin, August 2000.

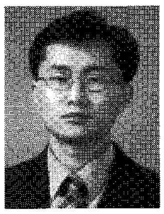
[5] J.F. Sowa, *Conceptual structures: Information processing in minds and machines*, Addison-Wesley, Reading, MA, 1984.

[6] J. Dibie-Barthélemy, O. Haemmerlé, and E. Salvat, A semantic validation of conceptual graphs, *Knowledge-Based Systems*, 19, pp. 489-510, 2006.

[7] A. Kayed and R.M. Colomb, Using BWW model to evaluate building ontologies in CGs formalism, *Information Systems*, 30, pp. 379-398, 2005.

[8] H. Peter Alesso and Craig F. Smith, *Developing Semantic Web Services*, AK Peters Ltd., Natick, Massachusetts, 2005.

저 자 소 개



김진성(Kim, Jin Sung)
김진성 (金珍成)은 현재, 전주대학교 경영학부 조교수로 재직 중이다. 주요 관심분야는 퍼지이론과 인공지능 기법을 이용한 지능형의사결정지원과 시멘틱 웹 서비스 등이다.

TEL : 063-220-2932
FAX : 063-220-2787
E-mail : kimjs@jj.ac.kr