Proceedings of KFIS Autumn Conference 2006 Vol. 16, No. 2.

PROLOG와 개념 그래프를 이용한 협동 온톨로지의 설계

Design of Cooperation Ontology by using PROLOG and Conceptual Graph

김진성

전북 전주시 완산구 효자동3가 1200 전주대학교 경영학부
E-mail: kimjs@jj.ac.kr

Abstract

This study proposes an ontology design framework to support the cooperation among devices by using PROLOG, Conceptual Graph (CG), and Resource Description Framework (RDF). Quite a large number of representation languages for representing ontology on the Web have been established over the last decade. Most of these researches are focused on design of independent resources description. In Semantic Web, however, cooperation ontology will be needed. In this study, the CG could make an entire conceptual view of knowledge and RDF can represent that knowledge. Then the PROLOG could support the natural inference based on that knowledge. Therefore, our proposed ontology will be used in the designing of Semantic Web-based cooperation systems.

Key Words : Conceptual graph, Cooperation, Ontology, PROLOG, RDF, Semantic Web

1. Introduction

During the development of the Semantic Web-based applications, the most important is the description of the resources/knowledge [1]. The diversity of resource/knowledge representation forms results in knowledge system developers choosing on formalism over others for a specific application.

In this paper, we will discuss translating and cooperating knowledge representation between two well-known knowledge representation standards: the PROLOG-based Conceptual Graph (CG) model and the Resource Description Framework (RDF) model. PROLOG (PROgramming in LOGic) is a programming language that has been designedfirst for natural language processing and theorem proving. It has been used, since that time in many fields of Artificial Intelligence (AI) [3]. PROLOG+CG is a *conceptual* and an *object-oriented extension* of PROLOG.

Some former researches for CGs and RDF pointed out that CGs and RDF are closely associated in concepts, syntax, and semantics [1, 4]. CGs and RDF are designed to represent domain knowledge, or systematic semantics for knowledge sharing and reuse. They are both graph-oriented models depicted by *nodes* connected with *arcs*: in CGs, *concept* nodes are connected by *conceptual relationship*arcs; in RDF, *resource* nodes are connected with *property* arcs [1].

The purpose of this study is to knowledge sharing and cooperation between knowledge models expressed in different representations CGs and RDF. In this paper, therefore, we apply the PROLOG+CG programming paradigm and RDF/XML for developing cooperation ontology. PROLOG+CG will provide a useful basis and conceptual view to illustrate and develop our ideas due to its simple semantics and widespread familiarity [5].

2. Research Background

2.1. Conceptual Graph

A Conceptual Graph (CG) is composed of: (i) a set of *concept vertices* (noted in rectangles) which represent the entities, attribute, states, events; (ii) a set of *relation* *vertices*(noted in ovals) which express the nature of the relationship between concepts; (iii) a set of *edges* linking relation vertices to concept vertices; (iv) a *label* for each vertex or edge [6].

To make CGs machine-process-able, Conceptual Graph Interchange Form (CGIF) is employed for encoding knowledge represented by CGs into machine-readable character strings [1]. Another CG model representation is Linear Form (LF) which makes CGs more human-readable. The following is an example illustrating three CG forms to represent the meaning of "Jin is going to Seoul by train" [1]

(1) Display Form:

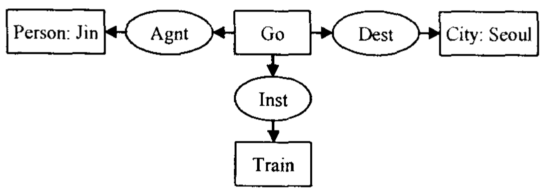

Fig. 1. A CG display form for "Jin is going to Seoul by train."

The linear form for CGs is intended as a more compact notation than Display Form, but with good human readability. It is exactly equivalent in expressive power to the abstract syntax and the Display Form. Following is the LF for Fig. 1:

(2) LF:
[Go] -
(Agnt)-> [Person: Jin]
(Dest) -> [City: Seoul]
(Inst) -> [Train].

For communication between machines, the CGIF has a simpler syntax and a more restricted character set. Following is the CGIF for Fig. 1:

(3) CGIF:
(Go [*x] (Agnt ?x [Person: Jin]) (Dest ?x [City: Seoul]) (Inst ?x [Train])

The PROLOG+CG has a similar syntax with LF. Following is the PROLOG+CG form for Fig. 1:

(4) PROLOG+CG:
[Go] -
- Agnt -> [Person: Jin],
- Dest -> [City: Seoul],
- Inst -> [Vehicle: Train].

2.2. Resource Description Framework

RDF is based on an underlying model with triples mad of *resource, property*, and *value* [4].

- A *resource* is an entity accessible by an URI on the Semantic Web (e.g. an XML document). Resources are the elements described by RDF statements.
- A *property* defines a binary relation between resources and/or its atomic values. Aproperty 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.

An RDF *statement* specifies a *value* for a *property* of a *resource*. The *statement* is represented by a node for the *subject*, a node for the *object*, and an arc for the *predicate* that is directed from the subject node to the object node. Each statement in RDF is a tripe corresponding to a single arc with a beginning node and an ending node. The principle idea of RDF is that all the things are naturally described by *object-attribute-value* triples. In the triple, *resources* are described using *properties* which have *values*. One characteristic that that distinguishes RDF from other knowledge representation language is that RDF uses Uniform Resource Identification (URI) references as 'resources' or 'properties' [1].

In order to be machine-process-able, RDF employs eXtensible Markup Language (XML) as its normative interchange syntax. Therefore, RDF has an XML syntax and can be seen as an object-oriented formalism for metadata statements. These metadata can rely on common ontology represented using RDF Schema (RDF-S).

For example, rdfs:Class and rdfs:Property are used to define the class of resources that are RDF classes and/or RDF properties; and rdfs:subClassOf and rdfs:subPropertyOf are used as properties to define relationship between RDF-S classes.

Here the example is used and represented in RDF and RDF-S. An RDF file must declare namespace to describe statements and description or resources.

Header of the RDF file:
<rdf:RDF xml:lang="en"
xmlns: rdf =
"http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns: rdfs
="http://www.w3.org/2000/01/rdf-schema#">

The following is an example of RDF statement/instance to be represented. Here, the example is shown in twoRDF representations this time: in graph mode expressed by Fig. 2 and in RDF/XML. The two representations have equivalent meaning.

(1) Graph format:

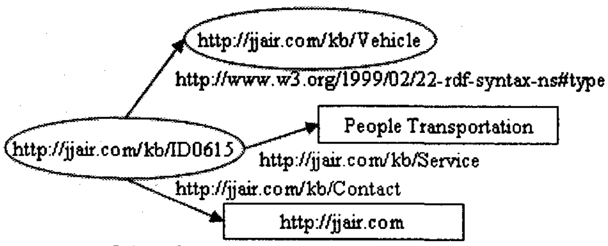

Fig. 2. An RDF graph format.

(2) RDF/XML format:
<? xml version = 1.0 ?>
<rdf:Description ID="AirService">
<rdf:RDF
xmlns:rdf =
"http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs =
"http://www.w3.org/2000/01/rdf-schema#"
xmlns:vehicle = "http://jjair.com/vehicle">
<rdfs:Class rdf:about =
"http://jjair.com/vehicle/Airplane" />
<rdf:Property rdf:about =
"http://jjair.com/vehicle/Airplane/Service" />
<rdf:Property rdf:about =
"http://jjair.com/vehicle/Airplane/Contact" />
<vehicle:Airplane rdf:about =
http://jjair.com/ID0615>
<vehicle:Service>People
Transportation</vehicle:service><vehicle:Contact>
http://jjair.com</vehicle:Contact>
</vehicle:Airplane>
</rdf:Description>
</rdf:RDF>

2.3. CGs and RDF conversion

RDF models and CG models are both graph models and are very similar [1, 6]. The model of CG formalism is based on (1) a support made of a concept type lattice and of a relationtype set, a set of individual markers enabling the designation of instances, a conformity relation between markers and types, and (2) a base of conceptual graphs built on this support. It therefore seems natural to translate a) the RDF statements into a base of CG-facts b) the hierarchy of classes appearing in an RDF-S into a concept type hierarchy in CG, and c) hierarchy of properties appearing in a RDF-S into a relation type hierarchy in CGs. Therefore, it is possible to convert CGs to RDF while retaining semantics, and vice versa.

Corby et al. [6] proposed an approach for translating RDF to CG. In this study, they implemented a prototype using the NOTIO CG platform and the VRP RDF parser from ICS Forth [6]. Yaho and Etzkorn [1] developed an automated CG to RDF converter. Since their propose is to propose a cooperative ontology for translating knowledge model represented in CGs into knowledge models represented in RDF, they incorporated NOTIO, the CGs'handling API, and the Jena, the RDF handling API, Into their automatic converter: NOTIO packagesare use to parse CG models in CGIF and build CG models in memory; and Jena packages are used to build RDF models and string the models into RDF/XML after the models being transferred.

3. Transformation of CG and RDF

We will describe a framework that performs a conversion of CGs to RDF (RDF to CGs) models and discuss the experimental results attained by using the framework to translate some typical examples. A basic RDF statement says something like: 'The producer of the resource found at http://jj.shmall.com/ID9027 is SJEC.' it can be stated as a triple by this way:

producer('http://jj.shmall.com/ID9027', 'SJEC')

Several statements can be written about the same resource, for example:

product('http://jj.shmall.com/ID9027', 'PDPTV-A100')
date('http://jj.shmall.com/ID9027', '20061015')

Written with the RDF/XML syntax:

```
<rdf:Description about =
"http://jj.shmall.com/ID9027">
  <producer>SJEC</producer>
  <product>PDPTV-A100</product>
  <date>20061015</date>
</rdf:Description>
```

This can be interpreted in PROLOG+CG as:

```
[Resource: "jj.shmall.com/ID9027"]-
- producer -> [Literal: SJEC],
- product -> [Literal: "PDPTV-A100"],
- date -> [Literal = "20061015"].
```

To show the mapping of nested RDF descriptions, we can express that the value of the 'company' property is itself a description of another resourece:

```
<rdf:Description about =
"http://jj.shmall.com/ID9027">
 <product>PDPTV-A100</product>
 <producer>SJEC</producer>
 <company>
   <rdf:Description about =
"http://jj.shmall.com/SJEC">
      <industry>Appliance</industry>
   </rdf:Description>
 </company>
</rdf:Description>
```

The RDF description will be translated into the following PROLOG+CG:

```
[Resource: "jj.shmall.com/ID9027"] -
- product -> [Literal: "PDPTV-A100"],
- producer -> [Literal: SJEC],
- company -> [Resource:
"jj.shmall.com/SJEC"],
- industry -> [Literal: Appliance].
```

4. Conclusions and future work

We have presented an approach towards cooperative use of the Web ontologies based on CGs and RDF-S. Our experiencewith PROLOG+CG and RDF shows this proposal is feasible, and a similar strategy should apply to any knowledge representation ontology.

In the future, the challenge is to study a logic programming and query based on PROLOG+CG and OWL. Since OWL is used as an open standard for developing large scale ontologies on the Semantic Web, in our future research we plan to study a intelligent query language for OWL statements and the mapping to appropriate CG models.

References

[1] Yao, H. and Etzkorn, L., Automated conversion between different knowledge representation formats, *Knowledge-Based* Systems, 19, 404-412, 2006.

[2] Sowa, J.F., Conceptual graph standard, working document: ISO/JTC1/SC 32/WG2000, 2001-04-02, http://www.jfsowa.com/cg/cgstand.htm, 2001.

[3] Kabbaj, A. and Petersen, U., PROLOG+CG version 2.0 user's manual, http://prologpluscg.sourceforge.net, 2006.

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

[5] Loke, S.W., Declarative programming of integrated peer-to-peer and Web bases systems: the case of Prolog, *The Journal of Systems and Software*, 79, 523-536, 2006.

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