INTRODUCTION
The Web consists of numerous pages encoded in HTML; its contents are readable and understandable for human, but not understandable for machine. So it's hard for machine to process the web contents meaningfully. But the web is enormous and still growing continuously, which makes it urgent for the web contents to be machine understandable so that the web contents can be processed automatically and meaningfully by machine. To solve these problems, Tim Berners-Lee put forward the Semantic Web [1][2], which is an extension of the current web. The Semantic Web is not just a platform for presenting information, but it can be understood and used to reason by machine.

To make the web contents understandable to machine and suitable for inference, we need to establish ontology and use terms defined in ontology as metadata to annotate the web contents. Compared with the presentation markups used on the web nowadays, these markups are semantic markups. [11]

In this paper, we distinguish the definitions of ontology and domain ontology. Based on definitions in [3,4,5,6], the definitions of domain, domain conceptualization, domain ontology and ontology are given as follows.

Definition1. Domain is a section of the world about which we wish to express some knowledge; domain conceptualization is to abstract a set of terms and a set of knowledge from the domain in terms of the tasks to be solved and the ontological commitment of ontology language used; domain ontology is the set of domain terms, the set of domain knowledge, it’s explicit specification of domain conceptualization, usually, we use ontology language to write down this specification; ontology is explicit specification of conceptualization about the world, there is only one ontology about the world, no application needs to use the whole ontology.

The relation between ontology and domain ontology is defined as follows:

Definition2. Suppose the world W can be divided into n domains, then ontology of W can be obtained by integrating domain ontologies of these n domains.

In practical application, domain ontology or the integration of several domain ontologies is needed.

In this paper, we concentrate on domain ontology representation, reasoning and integration for the Semantic Web, the organization of this paper is shown as follows.

Organization of This Paper
In section 2 we introduce ontology languages for the Semantic Web and choose OWL Lite and SWRL, which are suitable trade-offs between expressivity of knowledge and complexity of reasoning problems, as standard ontology languages on the Semantic Web. In section 3 we propose DORRSSW approach (domain ontology representation and reasoning for the Semantic Web). In section 4, we propose MDOISW approach (multiple domain ontologies integration for the Semantic Web). In section 5, DORRSSW approach and MDOISW approach are used to solve example problems to illustrate the procedures of these two approaches. In section 6 the related works are introduced. Section 7 gives the conclusion.
MULTIPLE DOMAIN ONTOLOGIES INTEGRATION FOR THE SEMANTIC WEB

In the original phase of the Semantic Web, there are few domain ontologies; to accomplish tasks in different domains, we use DORRSW approach to establish domain ontologies one by one. With the number of domain ontologies on the Semantic Web increasing, we often meet such a situation: to accomplish some task that needs the support of multiple existing domain ontologies, and every existing domain ontology alone can’t satisfy the need of that task. Two choices can be made:

1. Use DORRSW approach to create a new domain ontology about these domains, but we can’t utilize existing domain ontologies, so it costs a lot of repeated work; this new generated domain ontology is still an “island”, and it’s difficult for knowledge sharing.

2. Integrate domain ontologies needed to form a virtual domain ontology about these domains, we can utilize existing domain ontologies and realize knowledge sharing.

In this paper, we choose the second way.

Theorem 1. There are m domains, and we have constructed domain ontologies for these domains. Suppose we need to integrate these domain ontologies: do_1, do_2, ..., do_m, these domain ontologies are encoded in OWL Lite & SWRL and identified by URI: uri_1, uri_2, ..., uri_m; ∀i ∈ [1, m], do_i = (v_i, k_i); v_i is the set of domain terms, k_i is the set of domain knowledge, then

\[ \text{Integration} (\text{do}_i) \]

The set of domain knowledge for the integration of m domains can be divided into two parts. The first part is \( \bigcup k_i \), which reflects union of knowledge from m domain; the second part is IBK, which reflects the newly appeared knowledge for the integration of m domains, knowledge in IBK can’t be achieved in single domain of m domains. Knowledge in \( (\bigcup k_i) \cup \text{IBK} \) should keep consistency, or it denotes contradiction.

For using URI naming mechanism, every term is unique, so

\[ \sum_{i=1}^{m} \#(v_i) = \sum_{i=1}^{m} \#(v_i) \]

End.

Based on theorem 1, during the process of integrating domain ontologies, because \( \bigcup v_i \) and \( \bigcup k_i \) can be obtained by utilizing the existing domain ontologies, so the key of integrating domain ontologies is how to construct IBK. Based on this idea, the procedure of MDOISW approach is:

First, construct IBK. Use uri_IBK to identify IBK (uri_IBK can be a newly created URI, or an existing URI), suppose term t is defined in do_i, if t is to be quoted or defined in IBK, then in IBK, we can use uri_I#t to denote term t; after all the quotations and definitions are completed, we have

\[ \text{Integration} (\text{do}_i) \]

The ontology engineers do most of the work in this phase.

Second, call suitable reasoners to check consistency. Suitable reasoners are reasoners which can accomplish the inference problems of OWL Lite & SWRL, such as Vampire. After the second step, we have a virtual domain ontology about those domains being integrated. We can utilize this virtual domain ontology to solve problem. The machines do most of the work in this phase.

APPLICATION EXAMPLES

In section 5, firstly, we present application example of DORRSW approach, then we present application example of MDOISW approach.

Use DORRSW Approach to Represent an Example Animal Domain Ontology and Make Inference

1. Construct the set of domain terms

Suppose according to task to be solved, we need to consider animal domain as follows:

\[ \forall i \in \text{OWL Lite \& SWRL and identified by URI: uri_i} \]

Definition 3. OWL Lite makes the commitment that the world consists of class, datatype object property, datatype property, individual, data value; the ontological commitment of SWRL is the same.

Second, construct domain ontology. By using syntax of OWL Lite & SWRL and the terms in the set of domain terms, construct the set of domain knowledge; domain ontology is \{ the set of domain terms, the set of domain knowledge \}, so here we have domain ontology. The ontology engineers do most of the work in this phase.

Third, call suitable reasoners to reason for solving problem and checking consistency. Suitable reasoners are reasoners which can accomplish the inference problems of OWL Lite & SWRL, such as Vampire. This phase can solve problem and prevent inconsistent knowledge to occur during constructing domain ontology. The machines do most of the work in this phase.

Proof: \[ \text{Integration} (\text{do}_i) \]

domain ontology of the domain that is the integration of m domains; and \( \forall i \in [1, m] \), do_i = (v_i, k_i); so we can use the terms defined in m domain ontologies as terms of \[ \text{Integration} (\text{do}_i) \], then we obtain the following equation,

\[ \sum_{i=1}^{m} \#(v_i) \]

\( \#(v_i) \) denotes cardinality of set.
Person is animal; the parent of person is person, male person is man, female person is woman; the domain and range of hasParent is animal; hasAge is functional property, the value of hasAge is nonnegative integer; Adam is a person, his age is 13.

In terms of ontological commitment of OWL Lite, we obtain six types of terms:

**Class:** Animal, Male, Female, Person, Man, Woman  
**Datatype:** NonNegativeInteger  
**Object property:** hasParent  
**Datatype property:** hasAge  
**Individual:** Adam  
**Data value:** 13

### 2 Construct domain ontology

By using terms above, encode the domain ontology in OWL Lite as follows:

```xml
<owl:Class rdf:ID="Animal">
  <owl:equivalentClass>
    <owl:intersectionOf rdf:parseType="Collection">
      <owl:Class rdf:resource="# Animal"/>
      <owl:Restriction>
        <owl:onProperty rdf:resource="#hasParent"/>
        <owl:allValuesFrom rdf:resource="#Person"/>
      </owl:Restriction>
    </owl:intersectionOf>
  </owl:equivalentClass>
</owl:Class>

<owl:Class rdf:ID="Male">
  <owl:equivalentClass>
    <owl:intersectionOf rdf:parseType="Collection">
      <owl:Class rdf:resource="#Person"/>
      <owl:Class rdf:resource="#Male"/>
    </owl:intersectionOf>
  </owl:equivalentClass>
</owl:Class>

<owl:Class rdf:ID="Female">
  <owl:equivalentClass>
    <owl:intersectionOf rdf:parseType="Collection">
      <owl:Class rdf:resource="#Person"/>
      <owl:Class rdf:resource="#Female"/>
    </owl:intersectionOf>
  </owl:equivalentClass>
</owl:Class>

<owl:Class rdf:ID="Person">
  <owl:equivalentClass>
    <owl:intersectionOf rdf:parseType="Collection">
      <owl:Class rdf:resource="#Animal"/>
      <owl:Restriction>
        <owl:onProperty rdf:resource="#hasParent"/>
        <owl:allValuesFrom rdf:resource="#Person"/>
      </owl:Restriction>
    </owl:intersectionOf>
  </owl:equivalentClass>
</owl:Class>

<owl:Class rdf:ID="Man">
  <owl:equivalentClass>
    <owl:intersectionOf rdf:parseType="Collection">
      <owl:Class rdf:resource="#Person"/>
      <owl:Class rdf:resource="#Male"/>
    </owl:intersectionOf>
  </owl:equivalentClass>
</owl:Class>

<owl:Class rdf:ID="Woman">
  <owl:equivalentClass>
    <owl:intersectionOf rdf:parseType="Collection">
      <owl:Class rdf:resource="#Person"/>
      <owl:Class rdf:resource="#Female"/>
    </owl:intersectionOf>
  </owl:equivalentClass>
</owl:Class>

<Person rdf:ID="Adam">
  <hasAge>13</hasAge>
</Person>
```

The animal domain ontology can also be encoded in description logic:

```logic
Man ≡ Person ⊓ Male  
Woman ≡ Person ⊓ Female  
Person ≡ Animal ⊓ ∀hasParent Person  
≥ 1 hasParent ⊑ Animal  
T ⊑ ∀hasParent Animal  
T ⊑ ≤ 1 hasAge  
T ∀hasAge NonNegativeInteger
```

Call suitable reasoners to reason

According to this animal domain ontology, we can use Vampire to make inference for solving problem. For example, we can inquire of the reasoner about a question, such as “is Animal(Adam) true?” The reasoner will give a positive answer.

### 3 Call suitable reasoners to reason

Using MDOISW Approach to Integrate Two Example Domain Ontologies

Suppose the animal domain ontology ando_a defined in section 5.1 can be identified by uri_a, another animal domain ontology ando_b can be identified by uri_b. In ando_b, RobertAdam is MalePerson, MalePerson is Human, encode ando_b in OWL Lite & SWRL:

```xml
<owl:Class rdf:ID="MalePerson">
  <owl:equivalentClass>
    <owl:intersectionOf rdf:parseType="Collection">
      <owl:Class rdf:resource="#Person"/>
    </owl:intersectionOf>
  </owl:equivalentClass>
</owl:Class>

<owl:Class rdf:ID="RobertAdam"/>
</owl:Class>

<swrl:Variable rdf:ID="x"/>

<ruleml:Imp>
  <ruleml:body rdf:parseType="Collection">
    <swrl:ClassAtom>
      <swrl:className rdf:resource="MalePerson"/>
    </swrl:ClassAtom>
    <swrl:Variable rdf:resource="x"/>
  </ruleml:body>
  <ruleml:head rdf:parseType="Collection">
    <swrl:ClassAtom>
      <swrl:className rdf:resource="x"/>
    </swrl:ClassAtom>
  </ruleml:head>
</ruleml:Imp>
```

Copyright © 2006, Idea Group Inc. Copying or distributing in print or electronic forms without written permission of Idea Group Inc. is prohibited.
Now we have two domain ontologies, suppose the reality is: RobertAdam in ando_b is Adam in ando_a. MalePerson in ando_b is Man in ando_a, and Human in ando_b is Person in ando_a.

We can’t know “uri_a#Adam is man” if we only use ando_a to reason; we can’t know “uri_b#RobertAdam is 13 years old” if we only use ando_b to reason. So we need to integrate ando_a and ando_b, then we can solve problems that need the support of integration of ando_a and ando_b. In the next, MDOISW approach is used to integrate ando_a and ando_b.

1 Construct IBK

Create uri_IBK to locate IBK for the integration of ando_a and ando_b; in IBK, uri_a#Man, uri_a#Adam, uri_a#Person, uri_b#MalePerson, uri_b#RobertAdam, uri_b#Human are quoted and defined, and we can encode IBK in OWL Lite:

```xml
<owl:Class rdf:about="uri_b#MalePerson">
  <owl:sameAs rdf:resource="uri_a#Man"/>
</owl:Class>

<owl:Class rdf:about="uri_b#Human">
  <owl:sameAs rdf:resource="uri_a#Person"/>
</owl:Class>

<owl:Class>
  <owl:equivalentClass rdf:resource="uri_a#Person"/>
</owl:Class>

<owl:Thing rdf:about="uri_b#RobertAdam">
  <owl:sameAsIndividualAs rdf:resource="uri_a#Adam"/>
</owl:Class>

Encode IBK in description logic:

\[
\text{MalePerson}\equiv\text{Person}
\]

\[
\text{Adam}=\text{RobertAdam}
\]

2 Check consistency

By using Vampire, check consistency for integration of ando_a and ando_b; the result shows that the integration is consistent.

3 Call suitable reasoners to reason for solving problem

According to the integration of ando_a and ando_b, we can use Vampire to make inference for solving problem. For example, we can inquire of the reasoner about a question, such as “is Man(Adam) true?” The reasoner will give a positive answer. This solves problem that can’t be answered by using ando_a only.

RELATED WORKS

In [6], five design criteria for ontologies are proposed to guide and evaluate the design. We present them as follows:

1. Clarity: An ontology should effectively communicate the intended meaning of defined terms.
2. Coherence: An ontology should be coherent: that is, it should sanction inferences that are consistent with the definitions.
3. Extendibility: An ontology should be designed to anticipate the uses of the shared vocabulary.
4. Minimal encoding bias: The conceptualization should be specified at the knowledge level without depending on a particular symbol-level encoding.
5. Minimal ontological commitment: An ontology should require the minimal ontological commitment sufficient to support the intended knowledge sharing activities.

Ontology integration is introduced detailedly in [13]. The definitions of ontology mapping, ontology alignment, ontology articulation, ontology merging, ontology translation and ontology integration are distinguished. Though the name of the paper is “ontology mapping…”, ontology integration is the main focus of that paper based on the definition of ontology integration given in that paper. Present research works about ontology integration don’t fit for using on the Semantic Web, such a large distributed environment. The essential reason is that inference efficiency is unsatisfiable in those works, as those works don’t choose a suitable formal ontology language; so we have no effective way to check consistency after ontology integration.

In [14], based on experience of using ontology-editing environments such as Protégé-2000, Ontolingua and Chimaera, an ontology-development methodology for declarative frame-based systems is described. The steps in the ontology-development process are listed; the complex issues of defining class hierarchies and properties of classes and instances are addressed. In [15], Mike Uschold and Michael Gruninger propose an informal approach to developing ontology, which includes the following steps: identify purpose and scope, ontology capture, ontology coding, integrating existing ontologies, evaluation, documentation. Then a more rigorous approach to the development of ontologies is considered, and the role of formal languages in the specification, implementation, and evaluation of ontologies is discussed. These works are valuable references for the research in our paper.

CONCLUSION

To make the web contents understandable to machine and suitable for inference, we need to establish ontology and use terms defined in ontology as metadata to annotate the web contents. In practical application, domain ontology or the integration of several domain ontologies is needed. In this paper, we propose DORRSW approach and MDOISW approach to represent domain ontology, integrate multiple domain ontologies and make inference for consistency check and problem solving. The application examples prove the effectiveness of these two approaches. From these discussions, the basic situations of domain ontology representation, reasoning and integration for the Semantic Web are clarified, so the basic knowledge of developing ontology for the Semantic Web is provided.

ACKNOWLEDGEMENT

Supported by the National Natural Science Foundation of China under Grant No.60172012 and the National Science Foundation of HuNan Province under Grant No.03JJY3110.

REFERENCES


