Finding Semantic Web Ontology Terms from Words

Lushan Han
University of Maryland, Baltimore County
Baltimore MD, 21250 USA
lushan1@umbc.edu

Tim Finin
University of Maryland, Baltimore County
Baltimore MD, 21250 USA
finin@cs.umbc.edu

Yelena Yesha
University of Maryland, Baltimore County
Baltimore MD, 21250 USA
yeyesha@cs.umbc.edu

ABSTRACT
The Semantic Web was designed to unambiguously define and use ontologies to encode data and knowledge on the Web. Many people find it difficult, however, to write complex RDF statements and queries because it requires familiarity with the appropriate ontologies and the terms they define. We describe a framework that eases the experiences in authoring and querying RDF data, in which we focus on automatically finding a set of appropriate Semantic Web ontology terms from a set of words used as the labels of nodes and edges in an incoming semantic graph.

Keywords
Ontology Search, Ontology Reuse, Data Interoperability

1. INTRODUCTION
One important goal of the Semantic Web is to facilitate data sharing and integration through the use of ontologies (In this paper, ontologies refer particularly to vocabularies, i.e. classes and properties but not individuals). The Semantic Web doesn't rely on a single huge, all-encompassing ontology since reaching global consensus on vocabulary is infeasible. Rather it encourages the "bottom-up" development of small sharable ontologies and their reuse for data interoperability. For example, the popularity of the FOAF ontology for describing people and GEO for geographical positions greatly foster interoperability and integration. However, there are some challenges in this picture hindering the fast growth of the Semantic Web.

First, finding and selecting appropriate ontologies and terms is a difficult task for most users. The process becomes much more complex when the terms come from a large collection of specialized ontologies. Although domain specific applications can help in finding proper ontology terms, they also limit expressive power of users. Moreover, the problem of finding proper ontology terms is transfered to programmers or web site owners, for whom getting and maintaining familiarity with existing ontologies remains very hard. Since reusing is hard, people tend to create duplicated concepts in their own ontologies, which in turn makes consensus even harder.

Second, understanding how to promote and extend exisiting ontologies is still an unanswered question. The difficulty of finding appropriate ontologies is one reason. Another is that we are not very clear about the criteria for good ontology. Furthermore, different organizations or applications tend to promote their own ontologies, making the consensus even harder. The evolution of ontologies should be directly driven by people's information requirements, but in reality people can only interact with applications.

Thirdly, only a fraction of published Semantic Web data comes directly from users. Most of Semantic Web data are derived from databases or translated from other data sources. However, the Semantic Web would not become an overwhelming thing unless there is a way allowing people to freely express their information need, both in storing and retrieve their data. Although a graph of entities and relations can probably be easily understood by users, an RDF graph appears very complex to ordinary people. The auxiliary nodes used in complex structures such as lists and in representing ternary and higher order relations makes RDF graphs abstruse. Besides, there is still no standard about how to represent temporal and spatial data in RDF, which is, however, commonly used in human language. Thus, we need a high level graph representation (*semantic graph*) on the top of RDF data model, which looks natural to people and is sufficiently convenient so that people can directly use it to express their information needs.

These challenges are actually intertwined with each other. In this paper we present a framework intended to solve them simultaneously and we particularly focus on solving the problem of automatically finding ontology terms from words used as the labels of nodes and edges in an incoming *semantic graph*.

2. FRAMEWORK
Our framework has four components: a *semantic graph* interface for users to express their information needs, an underlying triplestore for storing and query RDF data, a collection of distributed Semantic Web ontologies harvested by a semantic web search engine, such as Swoogle [1] or Sindice [2], and a system to automatically translate a *semantic graph* to RDF graph.

2.1 Semantic Graph

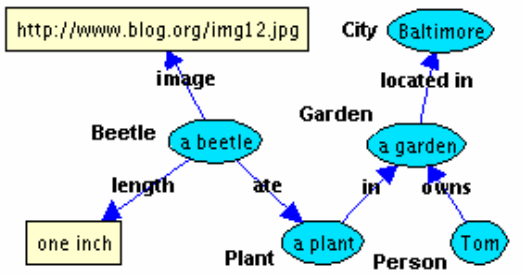

Figure 1: An example of a simple semantic graph.

An example *semantic graph* is illustrated in the Figure 1. It has the same semantic as the short text "One inch long beetle ate a plant in Tom's garden located in Baltimore city. It has an image at http://…". The semantic graph, unlike an RDF graph, allows users to directly use English vocabulary to label the nodes and edges. This is very convenient for users because they are not required to exactly remember the ontology terms. Typically, metadata systems like RDFa and Flickr's machine tags all use predefined vocabularies. Using *semantic graph* will greatly improve their user experi-

ence. By looking at the labels, you will find they are very similar to local names of terms lexicalized in ontologies. Anyway, they are all words. Words are very effective in communicating with human so that ontology terms are typically lexicalized using meaningful words. It is also possible to automatically learn a semantic graph from sentences as nowadays parse trees can be well learned using machine learning techniques.

2.2 Translating Semantic Graph to RDF

Translating a semantic graph to RDF graph involves syntactic mapping of two graph representations in complex structures, like list and bag, high order relationships and some others. However, the most important task is to map labels of nodes and edges to proper terms in distributed ontologies on the Semantic Web. The labels of nodes, such as *Plant*, are mapped to classes while the labels of edges, such as *ate,* are mapped to properties. We don't need map the data values and individuals like *one inch* and *Baltimore* because they are not schema but data that we will store and retrieve in the underlying triple store.

The very goal of translation is that data stored previously can be retrieved later despite of the mismatching vocabulary used in the input semantic graphs, though describing the same or close concepts, at different times. It is very likely that people can use different labels to describe the same concept since language often provides a rich selection of synonyms. For example, in Figure 1 *owns* can be replaced with *has* and *plant* can be replaced with *flower.* Overcoming vocabulary mismatch problem requires us matching on concepts, which goes beyond the surface comparison of words. Another major issue is that there are many duplicated concepts defined in different ontologies. So we will try to select terms in the most popular ontologies in the mapping.

Besides the vocabulary mismatch problem introduced by synonyms, word sense ambiguity is the other hard problem. The meanings of words (phrases) can only be disambiguated within a context, which is typically formed by a set of words with structures. In the example from Figure 1 the words used as node labels are typically concepts, such as *"Beetle"* and *"Plant"* and the words used as edge labels are typically properties or relations, such as "ate". There are two relationships that can happen between the words, which are memorized by human brain. One is the concept-property relationship such as *"Beetle-ate"* and the other is the co-occurrence relationship to different extents between concepts such as *"Plant-Garden"*. These relationships help form the context and therefore disambiguate the words.

To find consistent ontology terms matching the labels in a semantic graph we need have "contexts" on the Semantic Web. Semantic Web ontologies are typically small and context-specific. Most define a set of terms for a few or a very limited number of highly related concepts. People often use mixed vocabularies in authoring an RDF document and this practice bestows upon Semantic Web ontologies a very useful feature – ontologies can be connected by their co-occurrences in existing RDF documents. Thus we can form the ontology co-occurrence network with weighted directed edges indicating conditional probabilities that ontologies accompany each other. In this network, every ontology has a "context" which is not confined to the terms it defines but also includes the terms defined in its neighboring ontologies. According to the conditional probability that a neighboring ontology is connected to the central ontology, the terms defined in the neighboring ontology are assigned the corresponding weight in the context of the central ontology. The terms in the central ontology have unit weights. There are other ways to assign weights to the terms in an ontology context, but this is simple and effective. We choose using "ontology context" due to its simplicity and easy availability.

Finding the most consistent terms for the words used as labels in a semantic graph is actually the same question as finding the most related ontology context for the words. This can be achieved by finding the ontology context that returns the highest weight sum of its terms which match the words. By "match" we mean the local names of terms are synonym of the words. However, we still need normalize the term weight sum through dividing it by the number of terms defined in the central ontology. In this way, broad, all-encompassing ontologies such as CYC, WordNet and DBPedia cannot return a high value. Since ontologies may define duplicated concepts on the Semantic Web and we want to promote the popular ones, the normalized term weight sum is thus integrated with popularity using a simple weighted multiplication.

Slight change in the incoming semantic graph may lead to different optimal ontology context. However, they are highly likely to still stay at the top places. We can use different policies in writing and querying RDF data in the underlying triple store to hold the very goal of translation. In writing, we use the matching terms only in the optimal ontology context. In querying, we can compose multiple SPARQL queries using the matching terms in top k ontology contexts.

Determining if phrases are synonyms is a hard problem but seems feasible. Much research has been done in calculating similarity of words using WordNet and information content [3]. We have started working on extending it to phrase level. Synonyms can be picked using a similarity threshold, for example, 0.9. In this way false synonyms could be selected. However, because the weight sum computation is based on the whole set of words it has ability to resist limited noise in selecting the suitable ontology contexts.

3. IMPLEMENTATION AND RESULTS

An implementation and preliminary results of our framework is specified in the technical report [4]. We are also exploring immediate applications of these techniques to suggesting terms to encode spreadsheets [5] in RDF using words found in column headings.

4. REFERENCES

[1] L. Ding et al. Swoogle: A search and metadata engine for the semantic web. In: Proc. 13th ACM Conference on Information and Knowledge Management, 2004.

[2] G. Tummarello, et al..: Sindice.com: Weaving the open linked data. Proc. 6th Int. Semantic Web Conf., 2007.

[3] D. Lin,1998. An information-theoretic definition of similarity. In Proc. of the Int. Conf. on Machine Learning.

[4] Lushan Han, et al., Finding Appropriate Semantic Web Ontology Terms from Words, Technical Report, Ebiquity Lab, May 2009. http://ebiquity.umbc.edu/paper/html/id/457/

[5] Han, Lushan et al. RDF123: from Spreadsheets to RDF. In: Proc. 6th Int. Semantic Web Conf., (ISWC), Springer (2008)