A Web Service for Efficient Ontology Comparison

James Z. Wang Farha Ali Rashmy Appaneravanda
Department of Computer Science
Clemson University, Box 340974
Clemson, SC 29634-0974, USA
+1-864-656-7678
{jzwang, fali, rappane}@cs.clemson.edu

Abstract

With the growing access to heterogeneous and independent data repositories, determining the semantic difference of two ontologies is critical in information retrieval, information integration and semantic web services. In this paper, we develop a web service for ontology comparison based on our proposed senses refinement algorithm, which builds a senses set to accurately represent the semantics of the input ontology. The senses refinement algorithm automatically extracts senses from the electronic lexical database WordNet (locally installed or online), removes unnecessary senses based on the relationship among the entity classes of the ontology, and specifies relations and constraints of the concepts in the refined senses set. The senses refinement converts the measurement of ontology difference into simple set operations based on set theory, thus ensures the efficiency and accuracy of the ontology comparison. Our experimental studies show that the proposed senses refinement algorithm outperforms the naive algorithm in terms of efficiency and accuracy. We believe our web service is the first available online measurement tool for ontology comparison.

1. Introduction

The growth of World Wide Web as a knowledge repository has invigorated research for automatic extraction of knowledge from the Web. Recent studies have led to the tremendous success in semantic web [1] in which data can be automatically processed by software agents. Among all essential components of the semantic web, ontology plays the most important role since it makes the extraction and formalization of semantics possible. Ontology is an explicit formal specification on how to represent the objects, concepts and other entities, which are assumed to exist in some area of interest, and the relationships among them.

Much work related to ontology has been done in different areas including ontology presentation, construction and integration. Some researchers focus on defining common languages for ontology presentations [2, 3, 4]. The others build ontologies for different applications [5, 6, 7]. The concepts in ontology are represented in natural language words. As meaning of words and understanding of concepts differ in different communities, different users might use the same word for different concepts, or use different words for the same concept, or they might make different ontological assumptions about their concepts. Such possible heterogeneity causes problems in interoperability of knowledge resources. Due to the heterogeneity and independency of the data sources and data repositories, measuring the semantic similarity of two different ontologies is critical in information retrieval, information integration and semantic web queries [8, 9, 10, 11, 12]. Especially when P2P semantic web services become popular [13, 14], it is necessary to provide an online tool for efficiently measuring the semantic similarity of two ontologies. The semantic web agents in P2P semantic network may use the tool to make query routing decisions based on the semantic similarity of the ontologies provided by semantic web services. Currently there is no such tool available on internet due to the complexity of existing ontology comparison algorithms and certain requirement of human involvement in these algorithms.

In this paper, we fill the void by developing a web service for ontology comparison based on a novel senses refinement algorithm, which builds senses sets to accurately represent the concepts and semantic constraints of the input ontologies. The rest of this paper is organized as follows. We first discuss the background and existing approaches to ontology similarity problem in section 2. Then we give the formal definition of ontology difference based on set theory in section 3. We propose our senses refinement (SR) algorithm and discuss its advantages in section 4. In section 5, we use experimental studies to prove the efficiency and accuracy of our proposed senses refinement algorithm. We discuss the web service implementation issues in section 6. Finally we give our conclusion and discuss the future work in section 7.

2. Background and existing approaches

Recent studies in semantic web have emphasized on using ontologies and semantic similarity functions as mechanisms for directing queries across heterogeneous information repositories. Several approaches have been proposed to deal with the heterogeneity of ontologies. One approach is ontology integration by mapping the different ontologies into a more generic ontology [15, 16], or by vocabulary heterogeneity resolution [17, 18] of various ontologies. Once ontologies are integrated, the semantic similarity of entity classes is typically determined as a function of the path distance between terms in the hierarchical structure underlying this shared ontology [19, 20]. The semantic similarity of entity classes within the shared ontology can also be calculated using feature-matching [21] based on characteristics of objects or information content [9, 10] based on information theory.

There are two problems existing in ontology integration approaches. First, building a shared ontology is a very complicated process which is not suitable for online semantic web query processes. Second, these methods are designed to compare entity classes within the ontologies, yet no method has been proposed to measure the semantic similarity of two ontologies. Determining the semantic similarity of two ontologies is as important as measuring the semantic similarity of entity classes within the ontologies. Measuring the semantic similarity between two ontologies can help peer grouping and query routing in P2P semantic web services, as well as identifying potential collaboration in research areas such as GIS and bioinformatics.

The shared ontology idea has been taken to its extreme by SUMO (Suggested Upper Merged Ontology) [22]. Sanctioned by IEEE, SUMO suggests building a merged ontology by sharing ideas from all the available ontologies. The terms in SUMO will be mapped to WordNet [23] synsets to promote the use of SUMO in natural language understanding applications. The idea is that ontology designers will design their ontologies in natural language and then look for the SUMO entries in WordNet corresponding to the concepts they use, so that two different ontology designers will use the same term for the same concept. SUMO helps reducing the complexity of concept mapping, yet it does not address the requirement of ontology comparison. Furthermore, deriving the integrated ontology from a manual or semi-automatic process is not suitable for our online semantic web query process.

Another approach tries to create a computational model to assess semantic similarity among entity classes from different and independent ontologies without constructing *a priori* a shared ontology [24]. This approach uses a matching process to establish links among ontologies while keeping them autonomous. However it focuses on the semantic similarity of entity classes and does not allow deep processes due to the complexity of matching process. Thus using this approach to measure the semantic similarity of two ontologies is not practical.

In this paper, after giving a formal definition of ontology difference based on set theory, we propose an efficient ontology comparison algorithm that uses a novel senses refinement algorithm to convert ontology semantic difference measurement into set operations. The ultimate goal is to develop an ontology comparison web service that can not only address the aforementioned problems in existing approaches, but also provide accurate measurement of semantic difference of ontologies by automatically extracting senses from WordNet.

3. Ontology Difference

Most existing studies focus on measuring the semantic similarity of two entity classes in the same ontology or in different ontologies. No definition has been made to address the semantic similarity or difference between two ontologies. Although most articles use similarity to describe the semantic distance between a pair of entity classes in ontologies, we feel the term “difference” fits more naturally in comparing two different ontologies.

There are many ways to measure the difference between two given objects. For numeric data values, the difference can be calculated by using dissimilarity formulas. Yet for non numeric type of objects, it is necessary to correlate non numeric data to numeric values so that the difference can be quantified. Tversky defined a similarity measurement model [21] based on set theory so that difference in characteristics between objects can be evaluated by set operations. This similarity measurement model is also in agreement to an information-theoretic definition of similarity [25].

In this paper, we define our ontology measurement formula based on the normalization of Tversky’s model to give a numeric measurement of ontology difference. To facilitate set operations, we use senses set to summarize the semantics of the ontology. A senses set for an entity class is a set of synonym words denoting the concept of the entity class. A senses set for an ontology is obtained by extracting synonym words related to the ontology semantics from the senses sets of all concepts in the ontology. Assume the senses set of Target ontology is *T* and the senses set of Source ontology is *S*. The difference of set *T* from set *S*, denoted by $T-S$, is defined as

$$T-S=\{x \mid x\in T \wedge x\notin S\}$$

We use cardinality of the senses set to correlate the non numeric ontology semantics into numeric value. The cardinality of set $T-S$ indicates how many distinct synonym words existing in Target senses set *T* are not in Source senses set *S*. The cardinality of set T represents the number of distinct synonym words in Target senses set *T*. Thus the semantic difference between two ontologies can be defined by function $D(T,S)$ in following equation:

$$D(T,S)=\frac{|T-S|}{|T|} \qquad (1)$$

Based on Equation 1, we have $0\le D(T,S)\le 1$. When there is no common element between sense sets *T* and *S*, i.e., $|T-S|=|T|$, $D(T,S)=1$. On the other hand, if set T is a subset of set S ($T\subseteq S$), i.e., $|T-S|=$ 0, then $D(T,S)=0$.

This ontology difference measurement formula is not forced to satisfy symmetry property which is preserved by semantic distance based models [26]. That is, the semantic difference from ontology A to ontology B may not be the same as the semantic difference from ontology B to ontology A. Employing such an asymmetric measurement is important because we must ensure the ontology difference evaluations sensible to human judgments, in which cognitive properties of similarity play key roles. For instance, assume the senses set of ontology A and B are S_A and S_B respectively. If $S_A\subset S_B$, then $D(S_A,S_B)=0$ and $D(S_B,S_A)>0$. $D(S_A,S_B)=0$ means semantics existing in ontology A is also in ontology B. On the other hand, $D(S_B,S_A)>0$ means that ontology B includes some concepts that are not present in ontology A. Thus allowing the asymmetry in semantic difference of ontologies has significant importance in information retrieval and semantic web services. Especially in P2P semantic web services, asymmetric measurement of ontology difference allows semantic peer agents make proper decisions not only in self-configuring the P2P semantic overlay network but also in routing the semantic web queries. Similar asymmetric measurement approach is also adopted by some entity class comparison studies [24].

4. Efficient Ontology Comparison

Our proposed ontology measurement tries to correlate the non numeric ontology semantics into numeric cardinality of sets. Using only the concept labels of the entity classes can not yield accurate ontology comparison results, because the same concept may be represented by different words in different ontologies. It is necessary to discover the senses of the concepts to ensure accurate set operations. Thus how to efficiently build senses set that can accurately represent the semantics of the ontology becomes critical in ontology comparison. We propose a senses refinement algorithm that satisfies both efficiency and accuracy criteria.

4.1 Senses Refinement Algorithm

There are many entity classes associated with various concepts in an ontology. Each concept may have many senses because the evolution of the natural language has produced polysemy that the same word denotes more than one meaning. Yet not all senses of a concept should be included in the senses set for the ontology. Besides senses, the relations ("is-a" or "part-whole" relation) of concepts within the ontology also contributes to the semantics of the ontology. Furthermore, features of the entity classes add constraints to the ontology semantics. To build a senses set to accurately represent the semantics of the ontology, we have to answer the following questions:

- How do we automatically obtain the senses set for a concept in ontology?
- What senses of a concept should be included in the senses set for the ontology?
- What senses of a concept should be excluded from the senses set of the ontology?
- How can we represent the relations of concepts in the senses set for ontology?

In this paper, we take advantage of the electronic lexical database WordNet as does in SUMO project. The difference is that we automatically extract the synonym words and relations from WordNet for our ontology comparison while they use a manual or semi-automatic process to derive a shared ontology.

We design a proper programming interface to WordNet so that the senses for a concept can be automatically extracted and converted into the data structure used in our senses refinement algorithm for senses set construction. Once all senses of the concepts in an ontology are extracted out of WordNet, a naive algorithm to build the senses set for the ontology is to union all senses sets of individual concepts in the ontology. For instance, assume {C_1, C_2, …, C_n} are concepts in ontology O, and {S_1, S_2, …, S_n} are their corresponding senses sets extracted from WordNet. Using the naive algorithm for senses set construction, the senses set for ontology O can be calculated as $S_O=S_1\cup S_2\cup\cdots\cup S_n$.

However this naive approach has some problems. First, the evolution of the natural language has produced polysemy that the same word denotes more than one meaning. Not all senses of a concept should be included in the senses set for the ontology. Having unrelated senses in the ontology senses set will diminish the accuracy of measuring the ontology difference. Second, having too many unnecessary senses in the senses set hinders the efficiency of ontology comparison because larger number of elements in senses set incurs higher computation cost for set operations. Third, relations among entity classes in the ontology have to be included in the senses set so that the semantics of the ontology can be accurately represented by the senses set. The naive algorithm for senses set construction does not make any attempt to include relations in the senses set.

To address the aforementioned problems, we propose a senses refinement algorithm that refines the senses set of the ontology based on the semantic relationships between the parent concepts and the children concepts. There are two kinds of semantic relationships between the parent concept and the child concept according to WordNet. Hyponymy, i.e., "is-a" relation, is the most common relation used in ontologies. The "is-a" relation is transitive and asymmetric, and defines a hierarchical structure in which concepts inherit the entire characteristics from their superordinate concepts. Meronymy is the "part-whole" relation in which the child concept is part of the parent concept. These relations determine whether a particular sense of a concept should be included in the senses set of the ontology. Our senses refinement (SR) algorithm is based on "is-a" relation since it is the dominate relationship in ontologies. The algorithm is depicted in Figure 1.

The senses refinement algorithm explores the "is-a" relations between entity classes in ontology to determine whether a particular sense of a concept belongs to the senses set of the ontology. For each parent concept, the algorithm checks whether one of its senses is a hypernym of at least one synonym word of its children. If a match is found, the synonym sets for both the parent concept and the child concept are added to the senses set of the ontology. This process repeats until all entity classes in the ontology are examined. The algorithm returns the refined senses set of the ontology.

In "is-a" relation, the child concept is a specialization of its parent concept in the relationship hierarchy. So each sense of the child concept should be a specialization of at least one of the senses of the parent concept. The senses of the child concept that are not the specialization of any sense of the parent concept do not belong in the senses set for the given ontology. Similarly any sense of the parent concept that is not the generalization of any sense of its children concepts should not be included in the senses set for the ontology.

For instance, consider the simple ontology depicted in Figure 2.

```
Algorithm SR(Ontology O)
begin
  Q = {};
  P = {p | p∈ O && p is a parent in ontology O}
  for any $p \in$ P
    P_flag = false;
    $S_p$ = Senses set of p from WordNet;
    C = { c | c is a child of p in ontology O }
    for any $c \in$ C
      C_flag = false;
      $S_c$ = Senses set of c from WordNet;
      for any $s \in S_c$
        H = Hypernym set of s from WordNet;
        for any $h \in$ H
          if ( $h \in S_p$ )
            C_flag = true; P_flag = true;
            if ( $h == p$ )
              x = $c$ + "_is-a_" + $p$;
              Q = Q ∪ { x };
            else
              Q = Q ∪ { s };
            endif
          endif
        endfor
      endfor
      if(!C_flag)
        Q = Q ∪ { $c$ };
      endif
    endfor
    if (!P_flag)
      Q = Q ∪ { $h$ };
    else
      Q = Q ∪ { $p$ };
    endif
  endfor
  return Q;
end
```

Figure 1: Senses Refinement Algorithm

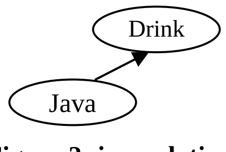

Figure 2: is-a relation.

The word "java" has three most obvious senses, i.e. a type of Coffee, an object-oriented programming language, or an island. Since object-oriented programming language and island are not specialization of drink, these senses of java do not belong to the senses set of this ontology. To further specify the relation between "drink" and "java", the senses refinement algorithm changes "java" into "java_is-a_drink" in the senses set of the ontology.

4.2 Ontology Comparison Based on SR Algorithm

Using the proposed senses refinement algorithm, we design a simple ontology comparison algorithm in Figure 3. This algorithm takes two ontologies as the input parameters and returns their semantic difference in numeric value.

```
Algorithm OntoCmp($O_S$, $O_T$)
begin
   S = SR($O_S$)
   T = SR($O_T$)
   $D = \frac{|T - S|}{|T|}$
   return D
end
```

Figure 3: Ontology Comparison Algorithm

To further demonstrate the execution flow of our ontology comparison algorithm, we apply the algorithm on some simple ontologies and show the steps of senses refinement and ontology comparison. Assume we have two ontologies, OntoBeverage and OntoPL, defined by "is-a" relation hierarchy. OntoBeverage in Figure 4 is a simple ontology representing two beverages, Java and Beer. OntoPL in Figure 5 is a simple ontology representing programming language Java. We further assume OntoBeverage is the target ontology and OntoPL is the source ontology.

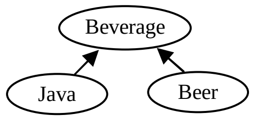

Figure 4: OntoBeverage

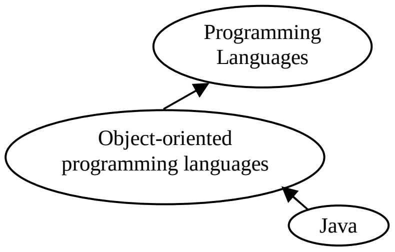

Figure 5: OntoPL

To compare those two ontologies, we need to get the refined senses set **T** for target ontology OntoBeverage and the refined senses set **S** for source OntoPL respectively. First we need to get the concepts and their senses with associated hypernyms for those two ontologies. In this example, we only consider first-level hypernyms, for more accurate results we can use hypernyms of higher levels. Table 1 contains the concepts and the senses with the associated hypernyms for ontology OntoBeverage obtained from WordNet.

Table 1: Senses and Hypernyms for OntoBeverage

Concepts	Senses	Hypernyms	In Sense Set?
Beverage	beverage, drink, drinkable, potable	food, nutrient	Yes
Java	1. Java	Island	No
	2. coffee, java	beverage, drink, drinkable, potable	Yes
	3. Java	object-oriented programming language, object-oriented programing language	No
Beer	Beer	brew, brewage	No

To get the refined target senses set **T**, we examine all concepts in the ontology starting from the root concept Beverage. First the senses set is empty, i.e., **T** = ***{ }***. Then we determine what senses of the concepts should be included in set **T** using our senses refinement algorithm. Looking at the parent concept Beverage and the child concept Java, the hypernyms of the second senses set of the child concept Java have common elements with the senses of its parent concept Beverage, thus the senses set ***{beverage, drink, drinkable, potable}*** for parent concept Beverage and the senses set ***{coffee, java}*** for child concept Java should be included in the senses set of the target ontology **T**. Now **T** = ***{beverage, drink, drinkable, potable, coffee, java}***. For the first and the third senses sets of the child concept Java, their Hypernyms have no common element with the senses of the parent concept Beverage, thus those senses for child concept Java can not be included in set **T**.

In addition to excluding the unrelated senses of the concepts, our senses refinement algorithm also specifies senses to reflect the relationship of child and parent concepts. Sometimes the synonyms sets for different senses of a concept contain the same word as the concept label itself. For example, "java" is the word used for child concept label in OntoBeverage. Three different senses sets for concept java can be extracted from WordNet. Among those three senses sets for java, only the second senses set can be included in the senses set for the ontology and the concept label "java" is in this senses set. In the meantime, the related parent concept label "beverage" is in the hypernyms set of java. This can be used to identify the "is-a" relation between concept "java" and "beverage". The "is-a" relation can also be used to differentiate "java" from other senses. To retain the "is-a" relationship in the senses set for ontology OntoBeverage, we specify sense "java" as "java_is-a_beverage". So the senses set for ontology OntoBeverage is **T** = ***{beverage, drink, drinkable, potable, coffee, java_is-a_beverage}*.**

Finally, if a concept does not have a single sense that matches with one of its parents' senses or a parent does not have a single sense that matches with hypernyms of all the senses of its children, we just include the concept label in the senses set of the ontology. Based on this rule, "beer" is added into the refined senses set T for OntoBeverage. Thus, **T = *{beverage, drink, drinkable, potable, Coffee, java_is-a_beverage, beer}***.

Similarly we can get the concepts and the senses with the associated hypernyms for source ontology OntoPL from WordNet. They are presented in

Table 2. Using our proposed senses refinement algorithm, we can get the refined senses set for source ontology OntoPL. That is, **S = *{programming language, programing language, object-oriented programming, language, object-oriented programing language, java_is-a_object oriented programming language}***. Using Equation 1, we get,

$$D(T,S) = \frac{|T - S|}{|T|} = 1$$

Table 2: Senses and Hypernyms for OntoPL

Concepts	**Senses**	**Hypernyms**	**In Sense Set?**
Programming Language	Programming language, programming lanuage	Artificial language	Yes
Object-oriented Programming Language	Object-oriented Programming Language, Object-oriented Programing Language	Programming language, programming lanuage	Yes
Java	1. Java	Island	No
	2. coffee, java	beverage, drink, drinkable, potable	No
	3. Java	object-oriented programming language, object-oriented programing language	Yes

Now let's change the source ontology to another ontology, OntoDrink, depicted in Figure 6. OntoDrink is a simple ontology representing some drinks. Now we want to use this ontology as the source ontology to compare with the target ontology OntoBeverage shown in Figure 4.

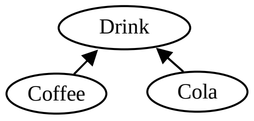

Figure 6: OntoDrink

Table 3 contains the concepts and the senses with the associated hypernyms for ontology OntoDrink obtained from WordNet.

Table 3: Senses and Hypernyms for OntoDrink

Concepts	**Senses**	**Hypernyms**	**In Sense Set?**
Drink	1. drink ,	small indefinite quantity, small indefinite amount	No
	2. drink, drinking, boozing, drunkenness, crapulence	intemperance, intemperateness	No
	3. beverage, drink, drinkable, potable	food, nutrient	Yes
	4. drink	body of water, water	No
	5. swallow, drink, deglutition	consumption, ingestion, intake, uptake	No
Coffee	1. coffee, java	beverage, drink, drinkable, potable	Yes
	2. coffee, coffee tree	tree	No
	3. coffee bean, coffee berry, coffee	Seed	No
	4. coffee, deep brown, umber, burnt umber	brown, brownness	No
Cola	1. cola, genus cola	dilleniid dicot genus	No
	2. cola, dope ,	soft drink	No

Using our senses refinement algorithm, we can get the refined senses set **S = *{beverage, drink, drinkable, potable, coffee_is-a_drink, java, cola}*** for the new source ontology OntoDrink. As discussed before, we have already got the refined target senses set **T** = ***{beverage, drink, drinkable, potable, coffee, java_is-a_beverage, beer}*.** Using equation 1, we get,

$$D(T,S) = \frac{|T - S|}{|T|} = 0.2857$$

These two examples demonstrate how our proposed ontology comparison algorithm works to measure the semantic difference of two ontologies based on set theory. Our ontology comparison algorithm shows the semantic difference between ontology OntoBeverage and OntoPL is 1. It means even though they are using the same concept label for one of their concepts, they are representing very different data. Conversely if we only look at the concept labels of Ontology OntoDrinks and OntoBeverage, they seem to be totally different. However, our ontology comparison algorithm reveals that the difference between these two Ontologies is just 0.2857. So these two Ontologies represent very similar concepts, although they have used different concept labels.

5. Performance study

Accuracy and efficiency are two criteria in evaluating the online ontology comparison tool. We have done some experimental studies to evaluate our proposed senses refinement algorithm in terms of those two performance metrics. Because there is no precedent work that measures the semantic difference of two ontologies (existing studies focus on measuring the semantic similarity of entity classes in ontologies), we will compare our senses refinement algorithm with the naive algorithm for senses set construction discussed in section 4.1.

5.1 Efficiency of the proposed ontology comparison algorithm

To evaluate how quickly the proposed ontology comparison algorithm returns the semantic difference of two given ontologies, we run the evaluated ontology comparison tools on some simple ontologies presented in Figure 7. We choose these Ontologies because most of their concepts have a lot of senses. Thus the time spent in senses refinement and senses specification should be noticeable when our senses refinement algorithm is used to construct the senses sets for the ontologies.

We implement our ontology comparison tool using J2SE 1.4.2. For performance comparison, we also implement an ontology comparison algorithm based on the naive algorithm for senses set construction. Without loss of generality, we compare each ontology with itself. We monitor the processing time for ontology comparison using different ontology comparison algorithms. The experimental studies are conducted on a desktop computer equipped with 1.3GHz Intel Pentium M processor and 512 MB RAM. The evaluated ontology comparison tools run as Java application under Windows XP. The experimental results are depicted in Table 4.

Table 4: Processing times using different ontology comparison algorithms

Ontology	Processing Time (milliseconds)	
	SR Algorithm	Naive Algorithm
Onto 5.1.1	0.06	1.03
Onto 5.1.2	0.32	0.59
Onto 5.1.3	0.06	0.26
Onto 5.1.4	0.12	0.29
Onto 5.1.5	0.14	3.21
Onto 5.1.6	0.08	0.66

We must note that the processing time reported in Table 4 do not include the time for fetching senses and hypernyms from the WordNet online, because the response latencies from WordNet online web server vary from time to time due to internet traffic and server workload. Actually, ignoring this time can give a better assessment on efficiency of the ontology comparison algorithms.

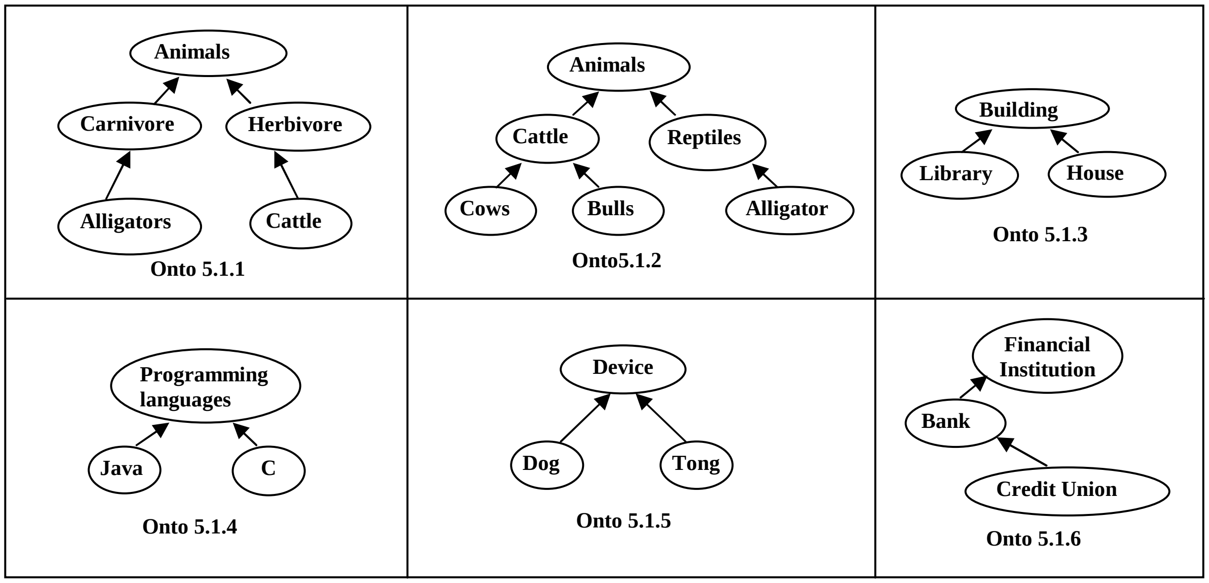

Figure 7: Ontologies for evaluating efficiency of ontology comparison

Intuitively the naive algorithm for sense construction should have better performance in terms of processing time because it simply computes the union of senses of all concepts in the ontology. On the other hand, our proposed senses refinement algorithm needs to eliminate unnecessary senses by investigating the relationships between concepts. It also needs to denote the relations in the senses set. All these processes are more complex than simple union used in the naive algorithm.

However, the senses refinement algorithm usually generates much smaller senses set for a given ontology if the ontology contains a lot of senses. Thus the time for calculating the ontology difference is reduced. The results shown in Table 4 have proven that the processing time of ontology comparison using our senses refinement algorithm is less than that using the naive algorithm. We must note here that the ontology comparison algorithm using our proposed senses refinement algorithm may not always be faster than that using the naive senses set construction algorithm if the concepts in the compared ontologies do not have many different senses. In these cases, the time saving in comparing the refined senses sets can not compensate the time that spends on senses refinement.

5.2 Accuracy of the proposed ontology comparison algorithm

To evaluate the accuracy of the proposed ontology comparison algorithm in measuring the semantic difference of ontologies, we run the ontology comparison tool on some simple ontologies presented in Figure 8. We compare the measurement results with that obtained by the ontology comparison algorithm based on the naive algorithm for senses set construction.

We choose these ontologies because they are very simple. Thus the accuracy of the ontology comparison can be easily judged by human observation. The experimental results are depicted in Table 5. The ontology comparison results show that our ontology comparison algorithm is more accurate than the ontology comparison algorithm based on naive senses set construction. The accuracy evaluation is based on our subjective judgment since the ontologies are very simple.

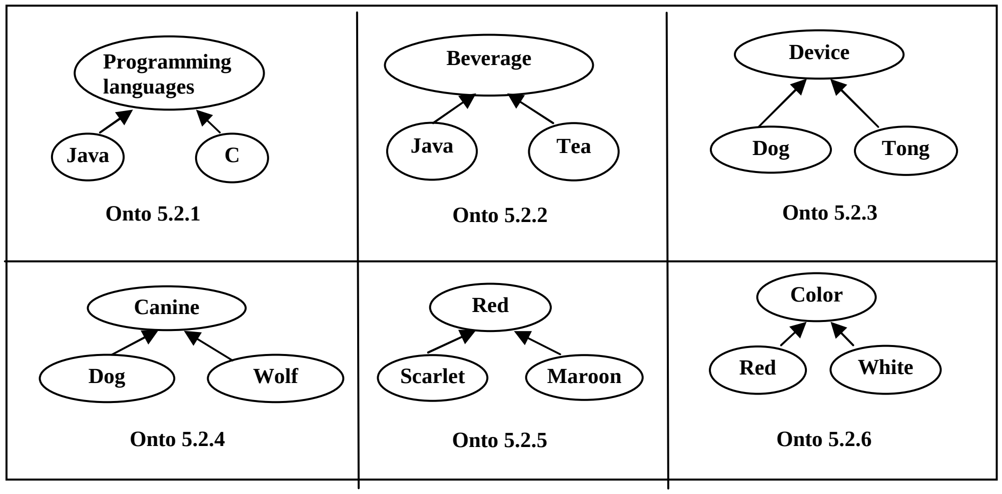

Figure 8: Ontologies for evaluating accuracy of ontology comparison.

Table 5: Ontology differences using different comparison algorithms

Target Ontology	**Source Ontology**	**Ontology Difference Value**	
		SR Algorithm	**Naive Algorithm**
Onto 5.2.1	Onto 5.2.2	1.0	0.916
Onto 5.2.2	Onto 5.2.1	1.0	0.818
Onto 5.2.3	Onto 5.2.4	1.0	0.120
Onto 5.2.4	Onto 5.2.3	1.0	0.435
Onto 5.2.5	Onto 5.2.6	0.857	0.3076
Onto 5.2.6	Onto 5.2.5	0.857	0.3076

Obviously we can tell that Onto 5.2.1 is totally different from Onto 5.2.2, and Onto 5.2.3 and Onto 5.2.4 represent entirely different concepts. The semantic difference of Onto 5.2.1 and Onto 5.2.2, and the semantic difference of Onto 5.2.3 and Onto 5.2.4 are clearly revealed by the ontology comparison algorithm based on our senses refinement algorithm. However the ontology comparison tool based on the naive senses set construction algorithm shows those ontologies have some similarity. Especially, Onto 5.2.3 and Onto 5.2.4 are measured as very similar as shown in Table 5 when the naive senses set construction algorithm is used.

Onto 5.2.5 and Onto 5.2.6 have some similarity because Onto 5.2.5 expands one child concept "red" in Onto 5.2.6. However Onto 5.2.5 gives more special meaning to concept red while Onto 5.2.6 represents more general senses of color white and red. When using Onto 5.2.5 as the target ontology and Onto 5.2.6 as the source ontology, we should not expect that semantic definitions for scarlet and maroon in Onto 5.2.5 are not included in senses set of the concept red in Onto 5.2.6. Thus semantic difference D(Onto 5.2.5, Onto 5.2.6) should be large since scarlet and maroon contribute 66% of the concepts in Onto 5.2.5. Our senses refinement algorithm reveals the difference value of 0.857 while the ontology comparison based on the naive senses set construction algorithm shows a very small difference value at 0.3076.

Overall, the experimental results prove that the senses sets built by our senses refinement algorithm accurately represent the ontology semantics.

6. Web service for ontology comparison

To provide an online ontology comparison tool, we integrate our proposed ontology comparison algorithm into a web service [27], ***OntoCmpService,*** which accepts a pair of ontologies written in OWL and returns a numeric value to represent their semantic difference. The web service ***OntoCmpService*** make use of the online electronic lexical database WordNet to generate senses sets for the input ontologies. We use J2EE SDK from Sun Microsystems as our development tool. ***OntoCmpService*** is implemented using JAX-RPC. It has a single interface class OntoIF that specifies the web service methods exposed to the public.

In ***OntoCmpService***, the exposed method is *ontoCompare* which is implemented in the class OntoImpl. The wscompile tool converts the Web Service interface to a WSDL file. We use Jena 2.1 Semantic web framework from HP labs to extract the concept labels and relationships from the input OWL files. The web service is deployed using Sun's Java systems Application Server. The interactions between the client and our ***OntoCmpService*** are depicted in Figure 9.

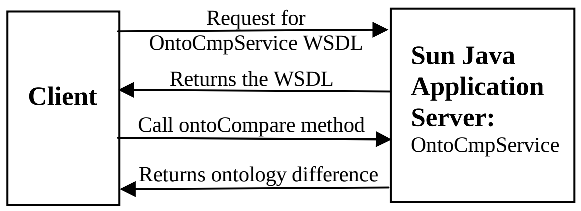
Client
Request for OntoCmpService WSDL
Returns the WSDL
Call ontoCompare method
Returns ontology difference
Sun Java Application Server:
OntoCmpService

Figure 9: Interactions between client and OntoCmpService

When a client requests for the ontology comparison web service, a WSDL file for *OntoCmpService* is returned to the client. The client then uses the stub class generated from WSDL file to call *ontoCompare* method for ontology comparison.

7. Conclusion and future studies

In this paper, we develop a web service for ontology comparison based on our proposed senses refinement algorithm, which builds senses sets to accurately represent the semantics of input ontologies. The senses refinement algorithm automatically extracts senses from the electronic lexical database WordNet (locally installed or online), removes unnecessary senses based on the relationship among the entity classes of the ontology, and specifies relations and constraints of the concepts in the refined senses set. The senses refinement converts the measurement of ontology semantic difference into simple set operations based on set theory, thus ensures the efficiency and accuracy of the ontology comparison. Our experimental studies show that the proposed senses refinement algorithm outperforms the naive algorithm in terms of efficiency and accuracy. We believe our web service is the first available online measurement tool for ontology comparison.

The proposed senses refinement algorithm focuses on "is-a" relations of the entity classes to discover the semantics of the ontology. Although "is-a" relation is the most common relation used in ontology, the "part-whole" relations [28], including "part-of", "whole-of" and "has-a" relations, may be used to further define the ontology semantics. Furthermore, attributes, functions and parts may be used to denote detailed semantic features about the entity classes in ontology. We are currently extending the senses refinement algorithm so that it can integrate the "part-whole" relations and semantic features of entity classes into the senses set construction for ontology comparison.

8. References

[1] T. Berners-Lee, J. Hendler, and O. Lassila. The SemanticWeb. *ScientificAmerican*, 284(5):34–43, 2001.

[2] Resource Description Framework (RDF), *W3C Semantic Web Actitity*. http://www.w3.org/RDF/

[3] Web Ontology Language (OWL), *W3C Semantic Web Actitity*. http://www.w3.org/2004/OWL/

[4] F. van Harmelen, P. F. Patel-Schneider, and I. Horrocks. Reference description of the daml+oil (march 2001) ontology markup language. http://www.daml.org/2001/03/reference.html, march 2001.

[5] C. Schlenoff, A. Knutilla and S. Ray. A Robust Ontology for Manufacturing Systems Integration. *In 2nd International Conference on Engineering Design and Automation*. 1998, Mai, HI.

[6] The Spatial Data Transfer Standard, *http://mcmcweb.er.usgs.gov/sdts/*

[7] The Gene Ontology Consortium: Creating the gene ontology resource: design and implementation. *Genome Research*, 11(8):1425-33, August 2001.

[8] N. Guarino, C. Masolo, and G. Vetere, OntoSeek: Content-Based Access to the Web, IEEE Intelligent Systems, 14(3), 70-80, May 1999.

[9] Philip Resnik. Semantic Similarity in a Taxonomy: An Information-Based Measure and Its Application to Problems of Ambiguity in Natural Language, *Journal of Artificial Intelligence Research (JAIR)*, Volume 11, pp. 95-130. 1999.

[10] J. J. Jiang and D. W. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of International Con-ference on Research in Computational Linguistics*, Taiwan, 1998.

[11] J. Lee, M. Kim, and Y. Lee, Information retrieval based on conceptual distance in is-a hierarchies. *Journal of documentation* 49(2), pp.188-207, 1993.

[12] A. Goni, E. Mena, and A. Illarramendi. Querying heterogeneous and distributed data repositories using ontologies. *In Proceedings of the 7th European-Japanese Conference on Information Modelling and Knowledge Bases* (IMKB'97), Toulouse, France, May 1997.

[13] W. Nejdl, B. Wolf, C. Qu, S. Decker, M. S. A. Naeve, M. Nilsson, M. Palmer, and T. Risch. EDUTELLA: A P2P networking infrastructure based on RDF. *In 11th World Wide Web Conference*, Honolulu, Hawaii, USA, May 2002.

[14] Christoph Tempich, Steffen Staab and Adrian Wranik, REMINDIN': Semantic Query Routing in Peer-to-Peer Networks Based on Social Metaphors, *The 13th World Wide Web Conference (WWW 2004)*, May 2004, New York, USA.

[15] P. Weinstein and P. Birmingham, Comparing Concepts in Differentiated Ontologies, *in 12th Workshop on Knowledge Acquisition, Modeling, and Management.* 1999, Banff, Canada.

[16] H. Stuckenschmidt, and I. J. Timm, Adapting communication vocabularies using shared ontologies. *In Proceedings of the Second International Workshop on Ontologies in Agent Systems, Workshop at 1st International Conference on Autonomous Agents and Multi-Agent Systems,* Bologna, Italy, 15-19 July 2002.

[17] V. Kashyap and A. Sheth, Semantic Heterogeneity in Global Information Systems: The Role of Metadata, Context, and Ontologies, *Cooperative Information Systems: Trends and Directions*, pp. 139-178, M. Papazoglou and G. Schlageter (eds.), Academic Press, London, UK. 1998.

[18] E. Mena, A. Illarramendi, V. Kashyap, and A. Sheth, OBSERVER: An Approach for Query Processing in Global Information Systems Based on Interoperation across Preexisting Ontologies. *Distributed and Parallel Databases,* 8(2): pp. 223-271, 2000.

[19] M. Bright, A. Hurson, and S. Pakzad, Automated Resolution of Semantic Heterogeneity in Multidatabases. *ACM Transactions on Database Systems*, 19(2), pp. 213-253, 1994.

[20] C. Collet, M. Huhns, and W. Shen, Resource Integration Using a Large Knowledge Base in Carnot. *Computer*, 24(12), pp. 55-62, 1991.

[21] A. Tversky, Features of Similarity. *Psychological Review,* 84(4), pp. 327-352, 1977.

[22] A. Pease, I. Niles, and J. Li, The Suggested Upper Merged Ontology: A Large Ontology for the Semantic Web and its Applications. *In Working Notes of the AAAI-2002 Workshop on Ontologies and the Semantic Web*, Edmonton, Canada, July 28-August 1, 2002.

[23] Christine Fellbaum (ed.), WordNet: An Electronic Lexical Database. The MIT Press, May 1998.

[24] M. A. Rodriguez and M. J. Egenhofer, Determining semantic similarity among entity classes from different ontologies; *IEEE Transactions on Knowledge and Data Engineering*. 2003.

[25] D. Lin, An Information-Theoretic Definition of Similarity (eds.), *International Conference on Machine Learning (ICML'98),* Madison, WI, 1998.

[26] R. H. Rada, H. Hili, E. Bicknell, and M. Blettner, Development and Application of a Metric on Semantic Nets. *IEEE Transactions on System, Man, and Cybernetics*, 19(1), pp. 17-30, 1989.

[27] Online Ontology Comparison, *http://www.cs.clemson.edu/~jzwang/ontocomp.htm*

[28] M. Winston, R. Chaffin, and D. Herramann, A Taxonomy of Part-Whole Relations. *Cognitive Science*, Volume 11, pp. 417-444, 1987.