Fourth International Conference on Semantics, Knowledge and Grid

Measuring Semantic Similarity between Resource Spaces

Zhixing Huang [1], Yuhui Qiu [2]

Semantic Grid Lab, Faculty of Computer and Information Science,
Southwest University, Chongqing, China, 400715
[1]huangzx@swu.edu.cn [2]yhqiu@swu.edu.cn

***Abstract*— A Web Resource Space Model (RSM) is a semantic data model for specifying, storing, managing and locating Web resources by appropriately classifying the contents of resources. For the task of detecting and retrieving relevant RSMs, one needs means for measuring the similarity between RSMs on a canonical scale. This paper proposes a general framework to automatically measure the similarity between RSMs, we break down the overall task and analyze the similarity between resource space models from three different levels, i.e. coordinate, axis and resource space. A variety of strategies and measurements that capture those differences are presented. Moreover, we theoretically analyzed the variation of similarities about Join, Disjoin, Merge and Split operations on RSMs. The empirical results of our method on Ontology Alignment Evaluation Initiative (OAEI) benchmark data set are also present in this paper.**

I. INTRODUCTION

A Web Resource Space Model (in simple RSM) is a semantic data model for specifying, storing, managing and locating Web resources by appropriately classifying the contents of resources [1]. A Web resource space is a multi-dimensional classification space, it makes use of the advantage that classification is a method of efficiently managing various resources and a basic method for human beings to know the real word and synthesize experience. The resource space model maps various resources (information, knowledge and service) into a multi-dimensional semantic space. The model is a semantic coordinates system that normalizes the classification semantics. The normal form theory and integrity theory of the RSM ensure the correctness of representation and operation on the resource space. The RSM is equipped with the SQL-like Resource Operation Language ROL to implement such operation as Join, Disjoin, Merge and Split [1]. Compared to ontology, RSM is based on strict mathematic foundations and it can provide many flexible functions and operations. RSM makes an excellent complement to current ontology technique for resource management.

However, in open or evolving systems, such as the semantic web, different parties would, in general, adopt different RSMs. Thus, just using RSM, like just using ontology, does not reduce *heterogeneity*: it raises heterogeneity problems at a higher level. The efficient resource management highly depends on the human behavior mode of dealing with resources and the mode of storing resources in the resource space. Up to now, we still hard to use existing taxonomy and commonsense as the classification method to establish consensus between designers and users. With the growing access to heterogeneous and independent data repositories, the treatment of differences in the structure and semantics of the data stored in those repositories play a major role in information systems. Recent investigation in information retrieval and data integration have emphasized the use of semantic similarity functions as a mechanism for comparing objects that can be retrieved or integrated across heterogeneous repositories. We needs means for measuring the similarity between RSMs on a canonical scale.

So, how may we measure the similarity of RSMs or of RSM parts? One could make use of the similarity of RSMs' corresponding ontologies. Although the Resource Space can be automatically constructed by transforming Web Ontology Language description files (e.g. RDF[1], OWL[2]) [2], we should not ignore the intrinsic differences between ontology and resource space. For instance, the inheritance structure of classes in RSMs should be tree(s), however most of the ontology descriptions (e.g. OWL) support multi-inheritance, and the inheritance structure is not a tree but a graph. Moreover, the ontology descriptions define the classes and their properties in a integrated way, but in RSM the classes and the properties are defined in separated axes. It means that a same RSM could be transformed from different ontologies, thus the similarity between the ontologies cannot reflect the distinction of RSMs. For the task of detecting and retrieving relevant RSMs, we should design a novel approach to measuring the similarity between RSMs. In this paper, we propose a solution through comparing the structural and semantic differences between different RSMs. To our knowledge, this paper is the first work measuring the similarity between RSMs.

The rest of this paper is organized as follows: In Section 2, the related works are briefly introduced. Section 3 describes the preliminaries of resource space model as well as the comparison between RSM and ontology. A general framework of measuring the similarity between RSMs as well as its implementation and properties are given in Section 4. We break down the overall task and analyze the similarity between resource space models from three different levels, i.e. coordinate, axis and resource space. The theoretical properties

[1]http://www.w3.org/RDF/.

[2]The W3C (www.w3.org) recommended the Web Ontology Language (OWL) in 2004 to support advanced Semantic Web applications by facility publication and sharing of ontology.

978-0-7695-3401-5/08 $25.00 © 2008 IEEE
DOI 10.1109/SKG.2008.12

of the similarity about RSM operations (including on Join, Disjoin, Merge and Split) and the complexity analysis of the method are also presented in this section. In Section 5, the experimental results on Ontology Alignment Evaluation Initiative(OAEI) benchmark data set are given. Finally, the conclusion and future work are presented in Section 6.

II. Related Work

The methods of evaluating the similarity between ontology entities have been studied widely [3][4][5][6][7][8][9][10]. Many different matching solutions have been proposed so far from various viewpoints, e.g., databases, information systems, and artificial intelligence. Among the process of calculating the similarity between ontologies, ontology matching plays a prominent important role. Ontology matching finds correspondences between semantically related entities of different ontologies. These correspondences may stand for equivalence as well as other relations, such as consequence, subsumption, or disjointness between ontology entities. To obtain the overall similarity between two ontologies, Meadche and Staab proposed a set of measures for describing the similarity of different ontology parts at the lexical and conceptual level [7][11]. Andrea et al. have presented a model for semantic similarity across different ontologies. The similarity model provides a systematic way to detect similar entity classes across ontologies based on the matching process of each of the specification components in the entity class representations (i.e., synonym set, distinguishing features, and semantic neighborhoods) [9]. Doan et al. have developed Glue, a system that employs learning techniques to semi-automatically create semantic mappings between ontologies [4]. Ram and Park provided a systematic method for automatically detecting and resolving various semantic conflicts in heterogenous schemas [8]. J. Euzenat presented the state of the art and the latest research results in ontology matching by providing a detailed account of matching techniques and matching systems in a systematic way from theoretical, practical and application perspectives [5]. Moreover, the evaluation criteria of ontology matching algorithms has also been discussed in [12] based on extending classical precision and recall rates. However, the systematic method for automatically measuring the similarity between resource space models has seldom been investigated so far.

Resource space model maps various resources (information, knowledge and service) into a multi-dimensional semantic space [1]. The mappings between three typical semantic models (Web Ontology Language, Relational Database Model and Resource Space Model) as well as related strategies and theories enable one to support the other have been investigated in [13]. Different from ontology description language which stores a entity and its properties together, resource space model separates them into different axes. The mapping from OWL description onto resource space includes: mapping inheritance hierarchy onto inheritance axis, mapping properties onto property axes, mapping individuals onto resources. In particular, Resource space can be constructed based on transforming existing ontologies, the detailed method has been proposed in [2].

Matching coordinates between two resource spaces is similar to matching entities between ontologies. To measure the similarity between resource spaces, we also need to analyze the difference between their axes. Compared with the method analyzing ontology entity difference by counting their superclass or subclass number [7], the tree distance approach is more suitable when evaluating the difference between axes, since the axis in resource space model can be viewed as a ordered labeled tree. However, the tree edit distance problem is difficult one. It has been proven that, if the trees are not ordered, the problem is NP-complete [14]. A survey about this problem has been presented in [15]. Despite the inherent complexity of the mapping problem in its generic formulation, there are serval practical applications that can be modelled using restricted formulations of it. The algorithms with quadratic costs have been proposed in [16][17][18]. Tree edit distance also has been used in measuring the similarity between XML documents. Nierman and Jagadish have proposed a structural similarity matric for XML documents based on an edit distance between ordered labeled trees [19].

III. Preliminaries of Resource Space Model

A resource space is a multi-dimensional classification space. The basic semantic elements of the Resource Space Model are *resource, resource space, axis and coordinate.* A resource space consists of a name and a set of axes, denoted as $RS(X_1, X_2, \cdots, X_n)$. Each axis X_i represents a classification method. X_i is partitioned by a set of coordinates denotes as $X_i =< C_{i1}, C_{i2}, \cdots, C_{im} >$.

A C represents a set of resources, denoted as $R(C)$. The semantics of a coordinate is represented by name, basic datatype, a set of concepts, or a coordinate tree (low-level coordinates finely classify their common ancestor). Thus, the measure of similarity between two coordinates should at least include these three parts.

An *axis* regulates a set of coordinates. An axis presents higher classification level than its coordinates. Two axes can be regarded as equivalent if their name are the same in semantics and the names of all the corresponding coordinates are the same in semantics.

A *resource space* regulates a set of axes and the refined classification relationship. A resource is determined by locating the point it belongs to and by selecting from the resource set according to its name and content description.

A. Operations of the Resource Space

The following are four operations of the resource space model [2]:

1) Join operation: If two resource spaces RS_1 and RS_2 specify the same type of resources and they have n $(n \geq 1)$ common axes, then they can be joined into one resource space RS such that RS_1 and RS_2 share these n common axes and $|RS| = |RS_1| + |RS_2| - n$, where $|RS|$ represents the number of dimensions of the

TABLE I
THE COMPARISON OF RSM AND ONTOLOGY

	RSM	Ontology
Dimension	N	2
Class Inheritance	Single	Multiple
Class Hierarchy	Tree	Graph
Class and Property Definition	Separately	Together
Visualization	3D	2D
Operation	Split, Join, etc.	Insert, Delete, etc.
Query Language	ROL	SPARQL

RS. RS is called the $join$ of RS_1 and RS_2, denoted as $RS_1 \cdot RS_2 \Rightarrow RS$.

2) Disjoin operation: A resource space RS can be disjoined into two resource spaces RS_1 and RS_2 (denote as $RS \Rightarrow RS_1 \cdot RS_2$) that specify the same type of resources as that of RS such that they have n ($1 \leq n \leq \min(|RS_1, RS_2|)$) common axes and $|RS| - n$ different axes, and $|RS| = |RS_1| + |RS_2| - n$.
3) Merge operation: If two resource spaces RS_1 and RS_2 specify the same type of resources and satisfy: (1) $RS_1 = RS_2 = n$; and (2) have $n - 1$ common axes, and there exist two different axes X_1 and X_2 satisfying the merge condition, then the two spaces can be merged into one RS by retaining the $n-1$ common axes in the new space and including a new axis $X = X_1 \cup X_2$.
4) Split operation: A resource space RS can be split into two resource spaces RS_1 and RS_2 that store the same type of resources as that of RS and that have $|RS| - 1$ common axes by splitting an axis X into two: X' and X'', such that $X = X' \cup X''$. This split operation is denoted as $RS \Rightarrow RS_1 \cup RS_2$.

To analyze the quality of a RSM, we can use following criterions: A *first-normal-form* (1NF) resource space is a resource space and there does not exist name duplication (semantic overlap) between coordinates at any axis. A *second-normal-form* (2NF) resource space is a *first-normal-form*, and for any two coordinates are independent from each other. A *third-normal-form* (3NF) resource space is a second-normal form and any two axes of it are orthogonal with each other.

B. The relationship between RSM and Ontology

The Resource Space can be automatically constructed by transforming Web Ontology Language description files, but this two typical semantic models have many intrinsic differences.

We briefly outline the main differences between RSM and Ontology descriptions (e.g. RDF, OWL) in the following:

- Most of the ontology descriptions support multi-inheritance, thus the inheritance structure is a graph, If the inheritance hierarchy is a tree or trees, it can be transformed into an axis to represent the category of resource. However the inheritance structure of classes in RSMs should be tree(s). The graph should be converted into tree(s) because the coordinates on an axis in a RSM should be tree(s). The degree of matching between two models determines the efficiency.
- The ontology descriptions define the classes and their properties in a integrated way, but in a RSM the classes and the properties are defined in separated axes.
- RSM is a multi-dimensional semantic space, while ontology can be viewed as a 2-dimensional semantic space. This character also determines their visualization methods.
- A suitable user interface for RSM navigation should be designed in the 3D space, but ontology is often visualized in 2D space for convenience, although there still exists some softwares can display ontology in 3D model [20].
- RSM provides a set of flexible operations on it, despite the normal operations on a ontology, such as insert and delete. Moreover, the two models use different query language ROL (Resource Operation Language) and SPARQL (Simple Protocol and RDF Query Language) respectively.

Thus, due to the existence of those distinctions, we should not expect to measure the similarity between RSMs by directly using the methods designed for ontologies. We propose a novel solution in the next section.

IV. SIMILARITY BETWEEN RESOURCE SPACES

A. Framework

The framework of calculating the similarity between two resource spaces are demonstrated in Figure 1. As we mentioned in previous section, a resource space model includes three granularities: coordinate, axis and space. Obviously, if we want to measure the similarity between two RSMs, we should know the similarity of each axes pair first; and analogously, we should figure out the differences of coordinate pairs before we calculate the similarity of two axes. Thus, the process involves those three main steps:

- Step 1: Calculate the similarity of each coordinate pair, and in each coordinate pair, the coordinates should belong to different axes;
- Step 2: Calculate the axis similarity after mapping the corresponding coordinates. Since an axis in a resource space can be viewed as a hierarchical tree, we can measure the difference between two axes by measuring the *tree edit distance* of two hierarchical trees;
- Step 3: After calculating the similarity of axes and mapping the corresponding axes which belong to different resource spaces. We can calculate the similarity between two resource spaces based on aggregating the similarities of those axes pairs in the final.

As shown in Figure 1, it should be noted that the preprocessing step, viz. transform ontology to resource space model, is an optional step if we have already constructed the resource space. In that situation, we may compare two resource space models from the step 1, i.e. coordinate similarity calculation, directly.

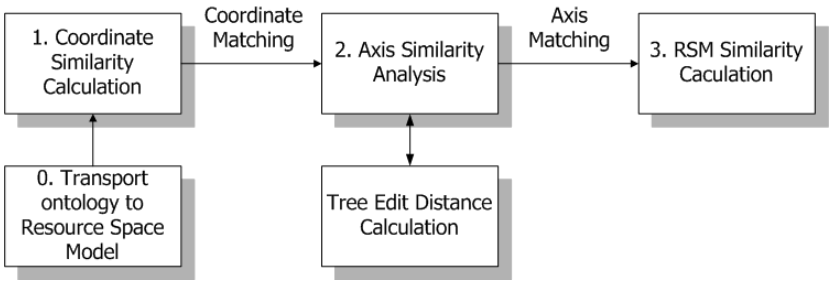

Fig. 1. The framework of measuring the similarity between resource spaces, where 1-3 represent the main processing steps and 0 represents preprocessing step.

B. Similarity of Coordinates

Different from ontology description language which stores a entity and its properties together, the resource space model separates them into different axes. The mapping from OWL description onto resource space includes: mapping inheritance hierarchy onto inheritance axis, mapping properties onto property axes, and mapping individuals onto resources [13]. To measure the similarity between two resource spaces, first of all, we should distinguish two kinds of axes in resource spaces, they are *property* axes and *concept* axes[3]. Moreover, the property axes can be further classified into *object* properties axes and *datatype* axes. The coordinates in different types of axes should be compared according to different methods and criterions.

For instance, a class Image has properties: format and year. The format property includes JPEG, BMP, etc., it should be viewed as object property. We can adapt *name-based techniques* [5] to measure the similarity between object properties. Meanwhile the year property is described as numbers, it should be viewed as datatype property. We can compare them based on their internal structures, such as *domain comparison* [5] method.

Definition 1: (Coordinate Similarity) Given two coordinates C_i and C_j, the similarity between those two coordinates is defined as follows:

- If C_i and C_j are both in datatype axes, $\sigma_c(C_i, C_j) = \pi_n\sigma_n(C_i, C_j) + \pi_d\sigma_d(C_i, C_j)$;
- If C_i and C_j are both in concept axes or both in object axes, $\sigma_c(C_i, C_j) = \sigma_n(C_i, C_j)$;
- otherwise, $\sigma_c(C_i, C_j) = 0$.

Parameter π_n and π_d define the importance of coordinate name and datatype respectively when C_i and C_j are both datatype coordinates, and $\pi_n + \pi_d = 1$.

To measure the similarity between two concepts, the *string edit distance* formulated by Levenshtein [21] is a well-established method. It measures the minimum number of token insertions, deletions, and substitutions requires to transform one string into another using a dynamic programming algorithm.

[3]It has also been called as *inheritance* axis in [13].

$$\sigma_c(C_i, C_j) = \frac{\max(|C_i|, |C_j|) - ed_s(C_i, C_j)}{\max(|C_i|, |C_j|)} \in [0, 1]. \quad (1)$$

Example 1: The string edit distance is denoted as ed_s in Equation 1. For example, the string edit distance between the two coordinate names $InProceedings$ and $In\_Proceedings$ is 1, $ed_s(InProceedings, In\_Proceedings) = 1$, because one insertion operation changes the string $InProceedings$ into $In\_Proceedings$, thus $\sigma_c(InProceedings, In\_Proceedings) = 13/14$.

Moveover, a more compressive method for evaluating the difference between two concepts can be adapted by using external lexicons or dictionaries, such as WordNet [22]. Since there exists a lot of synonyms in our vocabularies, they have the same or similar meanings but have quite different spellings.

Datatype coordinates are often represented as multiplicities, so we compare them using multiplicity similarity. Given two datatype coordinates C_i and C_j with expression domain range $[l, h]$ and $[l', h']$, the similarity between them is described as follows:

$$\sigma_d(C_i, C_j) = \begin{cases} 0 & l' > h \text{ or } l > h' \\ \dfrac{\min(h, h') - \max(l, l')}{\max(h, h') - \min(l, l')} & otherwise. \end{cases}$$

Example 2: If we compare a datatype with range $[0, 6]$ to datatypes with range $[2, 8]$, $[8, 12]$ and $[0, \infty]$ respectively, the comparison will yield 0.5, 0 and 6/MAXINT.

Note that since in this paper we focus on discussing the overall procedure of evaluating similarity between resource spaces, the various methods weighting the difference of two concepts and properties will not be exhaustively discussed here. Please see a survey of such methods in [5] for details.

C. Similarity of axes

Two axes can be regarded as equivalent if their name are same in semantics and the names of all the corresponding coordinates are same in semantics. However, when axes are designed by different users or organizations, it is hard to guarantee the axes will be completely equivalent. To measure the different between two axes, we should match the corresponding coordinates of these two axes first.

Based on the definition of the similarity of coordinates, we can derive the similarity of two axes. First of all, we should make alignment between coordinates before we measure the similarity between two axes. There are two notions of optimal matching of two sets of coordinates in this context. The first one is a optimum called *stable marriage* which consists of an assignment such that any permutation of two assignments provides a worse result. An algorithm for computing stable marriage is the Gale-Shapley algorithm [23].

Definition 2: (Stable marriage matching) Given two sets of concepts c and c' and a similarity function $\sigma : c \times c' \to [0, 1]$, extract a one-to-one alignment $M \subseteq c \times c'$, such that for any $(p, q) \in M$, $\sigma(p, q) + \sigma(r, s) \geq \sigma(p, s) + \sigma(r, q)$.

Another matching strategy is the global optimum, or maximum weight matching. It is the assignment for which there does not exist any other assignment with better weighting.

Definition 3: (Maximum weight matching) Given two sets of concepts c and c' and a similarity function $\sigma : c \times c' \to [0, 1]$, extract a one-to-one alignment $M \subseteq c \times c'$, such that for any one-to-one alignment $M' \subseteq c \times c'$,

$$\sum_{(p,q)\in M} \sigma(p, q) \geq \sum_{(p,q)\in M'} \sigma(p, q)$$

The matching strategies can be used in the coordinate matching phase as we described in this section, moreover these strategies also can be used in the axis matching phase as we discussed in next subsection.

The detailed process of calculating axis similarity is described in Algorithm 1. In this algorithm, after matching the coordinates come from two different axes, we filter the alignments with low similarity (lower than user given threshold), then we can measure the similarity between the axes based on aggregating the similarity of coordinates and hierarchical difference of the concepts.

Algorithm 1 $SimAxis(X, Y)$

1: **for** each coordinate pair (C_i, C_j), $C_i \in X$ and $C_j \in Y$ **do**
2: calculate the similarity $\sigma_c(C_i, C_j) = SimCoordinate(C_i, C_j)$.
3: **end for**
4: select coordinate pairs (C_i, C_j) into set Λ using a matching strategy.
5: (optional) filter alignments in Λ with low similarity.
6: $\bar{\sigma}_c = \frac{1}{|\Lambda|} \sum_{(C_i,C_j)\in\Lambda} \sigma_c(C_i, C_j)$.
7: $\sigma_a = \bar{\sigma}_c \times \frac{|X| + |Y| - ed_t(X, Y)}{|X| + |Y|}$.
8: **return**

Definition 4: For two axes $X(C_1, C_2, \cdots, C_n)$ and $Y(C'_1, C'_2, \cdots, C'_m)$, where n and m present the coordinate number of axes X and Y respectively, the similarity between two axes X and Y is defined as σ_a, and $\sigma_a \in [0, 1]$.

Since an axis in a resource space can be viewed as a hierarchical tree, we can measure the difference between two axes by measuring the *tree edit distance* [14] of two hierarchical trees.

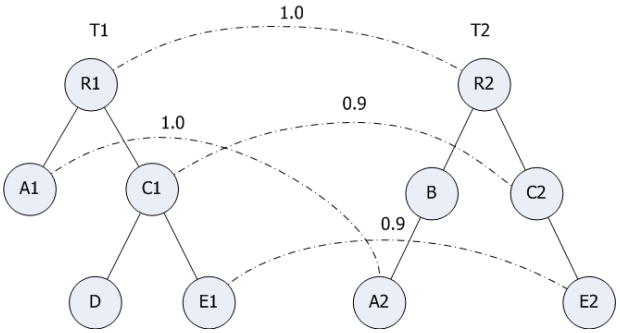

Fig. 2. A Mapping Example

In Algorithm 1, equation $\bar{\sigma}_c = \frac{1}{|\Lambda|} \sum_{(C_i,C_j)\in\Lambda} \sigma_c(C_i, C_j)$ represents the average coordinate similarity in mapping set Λ, and $ed_t(X, Y)$ represents the tree edit distance between axes X and Y.

Example 3: Suppose there are two hierarchical trees T_1 and T_2, their nodes and the mapping between T_1 and T_2 are demonstrated in Figure 2, and with the similarity $\sigma_c(R_1, R_2) = 1.0, \sigma_c(A_1, A_2) = 1.0, \sigma_c(C_1, C_2) = 0.9, \sigma_c(E_1, E_2) = 0.9$. Based on the mapping between T_1 and T_2, we can calculate the tree edit distance $ed_t(T_1, T_2) = 2$, which means we insert node $'B'$ and delete node $'D'$. According to Algorithm $SimAxis$, we have $\bar{\sigma}_c(T_1, T_2) = (1+1+0.9+0.9)/4 = 0.95$, then we get $\sigma_a = (10-2)/10 \times 0.95 = 0.76$.

Theorem 1: If two axes $X(C_1, C_2, \cdots, C_n)$ and $Y(C'_1, C'_2, \cdots, C'_n, \cdots, C'_m)$, where $m \geq n \geq 0$, have n common coordinates, then the similarity between X and Y $\sigma_a(X, Y)$ is n/m.

Proof: Without loss of generality, suppose $C_1, C_2, \cdots, C_n$ and $C'_1, C'_2, \cdots, C'_n$ are the common coordinates between axes X and Y, moreover we assume that for any $i \in [1, n]$, $C_i = C'_i$, it means that $\sigma_a(C_i, C'_i) = 1$. Moreover, for $|\Lambda| = n$ and $\max(|X|, |Y|) = m$, thus we have $\sigma_a(X, Y) = n/m$. ■

Lemma 1: For any axes X, we have: 1) $\sigma_a(X, X) = 1$; 2) $\sigma_a(X, \emptyset)$=0; 3) $\sigma_a(\emptyset, \emptyset) = 1$.

This Lemma can be immediately concluded from the result of Theorem 1.

Theorem 2: If an axis X can be split into two X_1 and X_2, such that $X = X_1 \cup X_2$, $X_1 \cap X_2 = \emptyset$, $|X| = n$, $|X_1| = n_1$ and $|X_2| = n_2$, where $0 \leq n_1, n_2 \leq n$, then we have:

1) $\sigma_a(X, X_1) = n_1/n$,
2) $\sigma_a(X, X_2) = n_2/n$,
3) $\sigma_a(X, X_1) + \sigma_a(X, X_2) = 1$;

Proof: For axes $X1$ and $X2$ are split from X, $X1$ and $X2$ have n_1 and n_2 common coordinates with X respectively, from the result of Theorem 1, we have $\sigma_a(X, X_1) = n_1/n$ and $\sigma_a(X, X_2) = n_2/n$. Moreover, since $X1 \cup X2 = X$ and $X1 \cap X2 = \emptyset$, we can conclude that $|X_1| + |X_2| = |X|$, i.e. $n_1 + n_2 = n$, thus we have $\sigma_a(X, X_1) + \sigma_a(X, X_2) = 1$. ■

D. Similarity between Resource Spaces

For the sake of simplicity, we suppose that all axes have the same importance in a resource space model.

Definition 5: For two resource spaces $RS(X_1, X_2, \cdots, X_n)$ and $RS'(Y_1, Y_2, \cdots, Y_m)$, where n and m present the dimension(axis) number of resource space RS and RS' respectively, the similarity between resource space RS and RS' is defined as σ_R.

Algorithm 2 $SimResourceSpace(RS, RS')$

1: **for** each axis pair (X_i, Y_j), $X_i \in RS$ and $Y_j \in RS'$ **do**
2: calculate the similarity $\sigma_a(i,j) = SimAxis(X_i, Y_j)$.
3: **end for**
4: select axis pairs (X_i, Y_j) into set Δ using a mapping strategy.
5: filter the alignments in Δ with low similarity.
6: $\sigma_R = \min(n/m, m/n) \prod_{(X_i,Y_j)\in\Delta} \sigma_a(X_i, Y_j)$
7: **return**

Theorem 3: If two resource spaces $RS(X_1, X_2, \cdots, X_n)$ and $RS'(X'_1, X'_2, \cdots, X'_n, \cdots, X'_m)$, where $m \geq n \geq 0$, have n common axes, then the similarity between RS and RS' $\sigma_R(RS, RS')$ is n/m.

Proof: Without loss of generality, suppose that axes $X_1, X_2, \cdots, X_n$ and $X'_1, X'_2, \cdots, X'_n$ are the common axes between resource space RS and RS', for any $1 \leq i \leq n$, we have $\sigma_a(X_i, X'_i) = 1$, due to $|\Lambda| = n$ and $\max(|X|, |Y|) = m$, thus we have $\sigma_a(X, Y) = n/m$. ■

Lemma 2: 1) $\sigma_R(RS, RS)$=1; 2) $\sigma_R(RS, \emptyset)$=0; 3) $\sigma_R(\emptyset, \emptyset)$=1;

This Lemma can be immediately concluded from the result of Theorem 3.

Theorem 4: If a resource space $RS(X_1, X_2, \cdots, X_n)$ is split into two resource spaces RS_1 and RS_2 that store the same type of resources as that of RS and have $n-1$ common axes by splitting an axis X into two axes X' and X'' such that $X = X' \cup X''$ and $X' \cap X'' = \emptyset$, then we have:

1) $\sigma_R(RS, RS_1) = \sigma_a(X, X') \in [0, 1]$,
2) $\sigma_R(RS, RS_2) = \sigma_a(X, X'') \in [0, 1]$,
3) $\sigma_R(RS, RS_1) + \sigma_R(RS, RS_2) = 1$.

Proof: Without loss of generality, suppose $X_1, X_2, \cdots, X_{n-1}$ are common axes and only axis X_n is split into two axes X'_n and X''_n, thus $\sigma_a(X_n, X'_n) \in [0, 1]$ and $\sigma_a(X_n, X'_n) \in [0, 1]$. For $X = X' \cup X''$ and $X' \cap X'' = \emptyset$, according to Theorem 1, we have $\sigma_a(X_n, X'_n) + \sigma_a(X_n, X'_n) = 1$. Moveover, in this circumstance, since $\min(n/m, m/n) = 1$ and $\sigma_a(X_i, X'_i) = 1$ $(1 \leq i \leq n-1)$, we have $\sigma_R(RS, RS_1) = \sigma_a(X_n, X'_n)$, $\sigma_R(RS, RS_2) = \sigma_a(X_n, X''_n)$ and $\sigma_R(RS, RS_1) + \sigma_R(RS, RS_2) = 1$. ■

Theorem 5: If a resource space $RS(X_1, X_2, \cdots, X_n)$ can be disjoined into two resource spaces RS_1 and RS_2 (denoted as $RS \Rightarrow RS_1 \cdot RS_2$) that specify the same type of resources as that of RS and have $m (1 \leq m \leq \min(|RS_1|, |RS_2|) < n)$ common axes and $n - m$ different axes, and $|RS| = |RS_1| + |RS_2| - m$, then we have:

1) $\sigma_R(RS, RS_1) = |RS_1|/n$,
2) $\sigma_R(RS, RS_2) = |RS_2|/n$,
3) $\sigma_R(RS, RS_1) + \sigma_R(RS, RS_2) - m/n = 1$.

Proof: Since RS_1 and RS_2 are disjoined from RS, thus RS_1 and RS_2 have $|RS_1|$ and $|RS_2|$ common axes with RS respectively. For any axis X in RS_1 or RS_2 we can find X' which $\sigma_a(X, X') = 1$, thus $\sigma_R(RS, RS_1) = |RS_1|/n$ and $\sigma_R(RS, RS_2) = |RS_2|/n$. Moveover, since $|RS| = |RS_1| + |RS_2| - m$, we have $\sigma_R(RS, RS_1) + \sigma_R(RS, RS_2) - m/n$ = $(|RS_1| + |RS_2|)/n - m/n$ = 1. ■

Example 4: Suppose we have a resource space $RS(X_1, X_2, X_3, X_4, X_5)$ with five different axes, RS can be disjoined into two resource spaces $RS_1(X_1, X_2, X_3)$ and $RS_2(X_1, X_2, X_4, X_5)$, so RS_1 and RS_2 have 2 common axes and $5 - 2 = 3$ different axes. According to Theorem 5, we have: $\sigma_R(RS, RS_1) = 0.6$, $\sigma_R(RS, RS_2) = 0.8$, and $\sigma_R(RS, RS_1) + \sigma_R(RS, RS_2) - 2/5 = 1$.

Lemma 3: If $\sigma_R(RS, RS_1) = a$, $\sigma_R(RS, RS_2) = b$, where $a, b \in [0, 1]$, then $\sigma_R(RS_1, RS_2) \in [\max(0, a + b - 1), 1]$;

Proof: If $a + b \leq 1$, the possible value of $\sigma_R(RS_1, RS_2)$ is in the interval $[0, 1]$, and if $a + b > 1$, $\sigma_R(RS_1, RS_2)$ is in the interval $[a + b - 1, 1]$. More specifically, in those two cases, no matter whether $a+b$ is greater or less than 1, $\sigma_R(RS_1, RS_2) = 1$ when $RS_1 = RS_2$. Combining above two cases, we can conclude that $\sigma_R(RS_1, RS_2) \in [\max(0, a + b - 1), 1]$. ■

E. Complexity Analysis

Since the number of axis is far less than the number of coordinate, the main computational burden lies on step 1, step 2 and the coordinate matching procedure among the whole processes.

In step 1, we need calculate similarity of each coordinate pair. Suppose there are total n_1 and n_2 coordinates in two different axes which belong to different resource spaces. We should calculate $n_1 n_2$ different pairs in the worst case. In coordinate matching process, if we use stable marriage matching strategy, the computation complexity is $O(\min(n_1, n_2) n_1 n_2)$.

In step 2, we need calculate the tree edit distance. Although it has been proven that, if the trees are not ordered, the problem is NP-complete [14], there are serval practical applications that can be modelled using restricted formulations of it. For example, the first algorithm for the mapping problem was presented in [16], and its complexity is $O(n_1 n_2 h_1 h_2)$, where n_1 and n_2 are the sizes of the trees and h_1 and h_2 are their heights. This is a dynamic programming algorithm that recursively calculates the edit distance between the strings formed by the sets of children vertices of each internal vertex in the tree. In [17], an algorithm was presented with cost $(d^2 n_1 n_2 min(h_1, l_1) min(h_2, l_2))$, where d is the edit distance between trees and l_1 and l_2 are the number of leaves in each tree. In [19], the *top-down* edit distance problem with cost $O(n_1 n_2)$ was proposed. Thus, the computational complexity of calculating the similarity between two axes is approximately $O(min(n_1, n_2) n_1 n_2)$.

Further, the whole computational complexity for evaluating the similarity between two resource spaces is

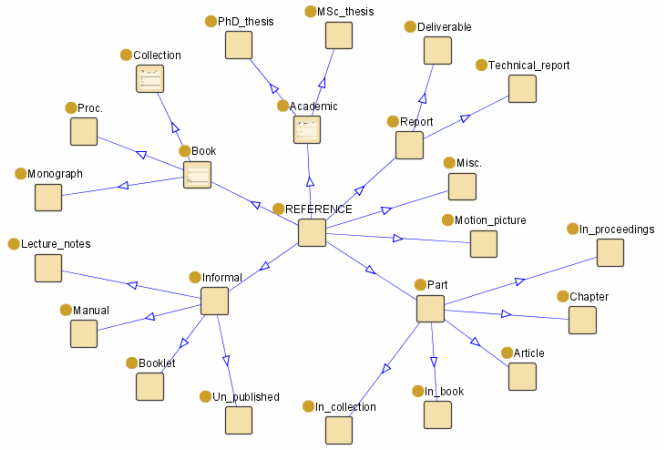

Fig. 3. A fragment of the ontology in OAEI benchmark package

$\sum_{i=1}^{|RS_1|}\sum_{j=1}^{|RS_2|} O(min(n_i, n_j)n_i n_j)$, where $|RS_1|$ and $|RS_2|$ represent the dimensions.

V. Experiments

We implement the our framework using Jena[4] Package. It provides a programmatic environment for RDF, RDFS and OWL, SPARQL and includes a rule-based inference engine. Since there is no resource space data set yet, we built resource spaces from transforming existing ontologies. We select Ontology Alignment Evaluation Initiative (OAEI) benchmark systematic suite[5] which has been used to evaluate different ontology matching techniques.

A. Data Set

Ontology Alignment Evaluation Initiative benchmark systematic suite is an artificial data set built from one OWL ontology on bibliography topic. It contains 33 named class, 24 object properties, 40 data properties, 56 named individuals and 20 anonymous individuals. Figure 3 shows a fragment of OAEI ontologies benchmark. The picture is created by using Java package Protégé[6] and Jambalaya[7].

The table below summarizes what has been retracted from the reference ontology. There are here 6 categories of alteration: Name of entities that can be replaced by (R/N) random strings, (S)ynonyms, (N)ame with different conventions, (F) strings in another language than english. Comments Comments can be (N) suppressed or (F) translated in another language. Specialization Hierarchy can be (N) suppressed, (E)xpansed or (F)lattened. Instances can be (N) suppressed Properties can be (N) suppressed or (R) having the restrictions on classes discarded. Classes can be (E)xpanded, i.e., replaced by several classes or (F)latened.

[4]http://jena.sourceforge.net/
[5]http://oaei.ontologymatching.org/
[6]Protégé: http://protege.stanford.edu/.
[7]Jambalaya: http://www.thechiselgroup.org/jambalaya.

B. Experimental Results

A fraction of experimental results is shown in Table III. In Table III column 2, 3 and 4 list the similarity between corresponding axes, the number in parenthesis indicates the coordinate number of its axis, and the similarity between each resource space and Test 101 are shown in the last column.

From the experimental results, we can conclude that: 1) the similarity between resource spaces can partially reflect the overall similarity between their corresponding ontologies. 2) The accuracy of measuring similarity between resource spaces is highly depended on the accuracy of the coordinate matching. However, we can improve the accuracy by using the alignment result of ontology matchings.

VI. Conclusions and Future Work

A Web Resource Space Model is a semantic data model for specifying, storing, managing and locating Web resources by appropriately classifying the contents of resources. For the task of detecting and retrieving relevant RSMs, one needs means for measuring the similarity between RSMs on a canonical scale. This paper proposes a general framework to measure the similarity between RSMs, we break down the overall task and capture the similarity of resource space models into three different levels, i.e. the coordinate, the axis and the overall resource space. The theoretical properties of the similarity of RSMs are analyzed, including on Join, Disjoin, Merge, Split operations. The empirical tests on OMEI benchmark data set are also given.

For simplifying the discussion, in this paper we ignore the instances in resource spaces, however our current model can be extended by comparing the instance difference between resource space models and it belongs to our next research agenda. Another future direction of our work includes expending current framework to evaluating the similarity between Extended Resource Space Models [24] which can deal with the uncertainty in resource classification and resource operation by incorporating probability concepts.

Acknowledgment

This work was partially supported by National Basic Research Program of China (973 project no. 2003CB317008), Natural Science Foundation Project of CQ CSTC (no. 2008BB2005) and a scientific research foundation of Southwest University (no. SWUB2007056). We are grateful to Dr. Yunpeng Xing, Dr. Juan Yang and Dr. Yun Bai for their insightful comments.

References

[1] H. Zhuge, *The Web Resource Space Model*. Springer, 2007.
[2] H. Zhuge, P. Shi, Y. Xing, and C. He, "Transformation from owl description to resource space model," in *The 1st Asian Semantic Web Conference*, 2006, pp. 4–23.
[3] P. Ceravolo, E. Damiani, and M. Viviani, "Bottom up extraction and trust-based refinement of ontology metadata," *IEEE Transaction on Knowledge and Data Engineering*, vol. 19, no. 2, pp. 149–163, Feb. 2007.

TABLE II
OMEI TEST SET

Test	Name	Comments	Hierarchy	Properties	Class	Comment
101	0	0	0	0	0	Reference alignment
102	0	0	0	0	0	Irrelevant ontology
103	0	0	0	0	0	Language generalization
104	0	0	0	0	0	Language restriction
201	R	0	0	0	0	No names
202	R	N	0	0	0	No names, no comments
203	0	N	0	0	0	No comments
204	C	0	0	0	0	Naming conventions
205	S	0	0	0	0	Synonyms
221	0	0	N	0	0	No specialisation
222	0	0	F	0	0	Flatenned
223	0	0	E	0	0	Expanded hierarchy
224	0	0	0	N	0	No instance
225	0	0	0	0	R	No restrictions

TABLE III
COMPARISON WITH TEST 101

Test	Datatype axis	Object axis	Class axis	Similarity
101	1.00(46)	1.00(26)	1.00(36)	1.00
102	0.01(1)	0.25(16)	0.227(136)	0.00
103	1.00(46)	1.00(26)	1.00(36)	1.00
104	1.00(46)	1.00(26)	1.00(36)	1.00
201	0.30(46)	0.25(26)	0.24(36)	0.02
202	0.30(46)	0.25(26)	0.24(36)	0.02
203	1.00(46)	1.00(26)	1.00(36)	1.00
204	0.92(46)	0.90(26)	0.93(36)	0.77
205	0.63(45)	0.43(26)	0.56(36)	0.15
221	1.00(45)	1.00(26)	0.50(36)	0.50
222	1.00(45)	1.00(26)	0.59(32)	0.59
223	1.00(45)	0.98(27)	0.35(69)	0.34
224	1.00(46)	1.00(26)	1.00(36)	1.00
225	1.00(46)	1.00(26)	1.00(36)	1.00

[4] A. Doan, J. Madhavan, P. Domingos, and A. Halevy, *Handbook on Ontologies in Information Systems*. Springer-Velag, 2003, ch. Ontology matching: A machine learning approach.

[5] J. Euzenat and P. Shvaiko, *Ontology matching*. Heidelberg (DE): Springer-Verlag, 2007.

[6] Y. Li, Z. A. Bandar, and D. Mclean, "An approach for measuring semantic similarity between words using multiple infomation source," *IEEE Transactions on Knowledge and Data Engineering*, vol. 15, no. 4, pp. 871–881, 2003.

[7] A. Maedche and S. Staab, "Measuring similarity between ontologies," in *Proc. Of the European Conference on Knowledge Acquisition and Management (EKAW-2002)*. LNCS, Springe, October 1-4, 2002, Madrid, Spain.

[8] S. Ram, J. Park, and M. Viviani, "Semantic confilct resolution ontology: An ontology for detecting and resolving data and schema-level semantic conflicts," *IEEE Transaction on Knowledge and Data Engineering*, vol. 16, no. 2, pp. 189–201, Feb. 2004.

[9] M. A. Rodriguez and M. J. Egenhofer, "Determining semantic similarity among entity classes from different ontologies," *IEEE Trans. on Knowledge and Data Eng.*, vol. 15(2), pp. 442–456, 2003.

[10] W. Hu, G. Cheng, D. Zheng, X. Zhong, and Y. Qu, "The results of falcon-ao in the oaei 2006 campaign." in *Ontology Matching*, 2006.

[11] P. Cimiano, A. Hotho, and S. Staab, "Learning concept hierarchies from text corpora using formal concept analysis," *Journal of AI Research*, vol. 24, pp. 305–339, 2005.

[12] J. Euzenat, "Semantic precision and recall for ontology alignment evaluation," in *Proc. 20th International Joint Conference on Artificial Intelligence (IJCAI)*. AAAI Press, 2007, pp. 248–253.

[13] H. Zhuge, Y. Xing, and P. Shi, "Resource space model, owl and database: Mapping and integaration," *ACM Transactions on Internet Technology*, vol. 18, no. 4, 2008.

[14] K. Zhang, R. Statman, and D. Shasha, "On the editing distance between unordered labeled trees," *Inf. Process. Lett.*, vol. 42, no. 3, pp. 133–139, 1992.

[15] P. Bille, "A survey on tree edit distance and related problems," *Theor. Comput. Sci.*, vol. 337, no. 1-3, pp. 217–239, 2005.

[16] K.-C. Tai, "The tree-to-tree correction problem," *J. ACM*, vol. 26, no. 3, pp. 422–433, 1979.

[17] J. T. L. Wang, B. A. Shapiro, D. Shasha, K. Zhang, and K. M. Currey, "An algorithm for finding the largest approximately common substructures of two trees," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 8, pp. 889–895, 1998.

[18] K. Zhang and D. Shasha, "Simple fast algorithms for the editing distance between trees and related problems," *SIAM J. Comput.*, vol. 18, no. 6, pp. 1245–1262, 1989.

[19] A. Nierman and H. Jagadish, "Evaluating structural similarity in xml documents," in *Proceedings of 5th International Workshop on the Web and Databases*, Madison, WI, 2002.

[20] M.-A. Storey, N. F. Noy, M. Musen, C. Best, R. Fergerson, and N. Ernst, "Jambalaya: an interactive environment for exploring ontologies," in *IUI '02: Proceedings of the 7th international conference on Intelligent user interfaces*. New York, NY, USA: ACM, 2002, pp. 239–239.

[21] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," *Soviet Physics Doklady*, vol. 16, pp. 189–201, 1966.

[22] C. Fellbaum, *WordNet: An Electronic Lexical Database*. the MIT press, 1998.

[23] D. Gale and L. S. Shapley, "College admissions and the stability of marriage," *American Mathematical Monthly*, vol. 69, pp. 9–14, 1962.

[24] H. Zhuge, E. Yao, Y. Xing, and J. Liu, "Extended resource space model," *Future Generation Computer Systems*, vol. 21, pp. 189–198, 2005.