RiMOM: A Dynamic Multistrategy Ontology Alignment Framework

Juanzi Li, Jie Tang, *Member*, *IEEE*, Yi Li, and Qiong Luo

Abstract—Ontology alignment identifies semantically matching entities in different ontologies. Various ontology alignment strategies have been proposed; however, few systems have explored how to automatically combine multiple strategies to improve the matching effectiveness. This paper presents a dynamic multistrategy ontology alignment framework, named RiMOM. The key insight in this framework is that similarity characteristics between ontologies may vary widely. We propose a systematic approach to quantitatively estimate the similarity characteristics for each alignment task and propose a strategy selection method to automatically combine the matching strategies based on two estimated factors. In the approach, we consider both textual and structural characteristics of ontologies. With RiMOM, we participated in the 2006 and 2007 campaigns of the Ontology Alignment Evaluation Initiative (OAEI). Our system is among the top three performers in benchmark data sets.

Index Terms—Heterogeneous databases, knowledge and data engineering tools and techniques, ontology languages.

1 Introduction

Ontology, as the means to conceptualize domain knowledge, has become the enabler of the fulfillment of the Semantic Web vision. It aims to make data sharable. Unfortunately, ontologies themselves are heterogeneous and distributed. Defined by different organizations or by different people in the same organization, ontologies can have vastly different characteristics. Specifically, entities (including concepts, relations, or instances) with the same meaning may have different labels in different ontologies; the same label may represent different meanings.

For example, type the keyword "publication" in Swoogle [8], a Semantic Web search engine, more than 2,774 ontologies will be returned. Therefore, in order to achieve *semantic interoperability* across ontologies, it is necessary to discover the alignment across ontologies.

Considerable work has been made on automating the process of ontology alignment, either focusing on specific applications or aiming at providing a generic way for various applications, as summarized in recent surveys [7], [18], [31], [49], [61], [57], [22]. The existing techniques are mostly based on calculating similarities between entities of two ontologies by utilizing various types of information in ontologies, e.g., entity names, taxonomy structures, constraints, and entities' instances. These methods can be classified into two categories: using a single strategy versus combining multiple strategies. In the former, all available information are defined as features in a single similarity function; while in the latter, different similarity functions are defined based on different types of information, and a composite method is used to combine the results of different similarities. In recent years, the combination method becomes more and more popular, due to its ease of extension and flexibility. In our previous work, we also proposed RiMOM [60] for ontology alignment by combining different strategies. Experimental results show that the combination method outperforms the single strategy based method in many cases.

However, several problems for ontology alignment are still needed to further investigate:

1. Is it always right to use the combination of different strategies for ontology alignment? Actually, our preliminary experiments show that a combined method may underperform a single strategy in some cases. Considering, for instance, two ontologies defined in different languages (e.g., English and French) but have the same taxonomy structure. A structure-based strategy, which is the similar structures in the two ontologies, will be effective; whereas a label name-based strategy may be useless (even play negative effect), as these ontologies are defined in different languages.
2. When should we use a combination method and when should a single strategy? A major limitation of existing approaches is that they need tune the thresholds (or weights) for each strategy so as to find the optimal configuration in the combination. However, the "optimal" configuration may work for some cases but may not succeed when we change the context. Hence, the challenge is to find theoretical criteria to determine when a special strategy should be used, given an alignment task.

A challenging issue in traditional methods is that both single and combination strategies are statically determined without considering characteristics of the alignment task. Basically, we need an effective mechanism to automatically

- *J. Li, J. Tang, and Y. Li are with Tsinghua University, Beijing, 100084 P.R. China. E-mail: ljz@keg.cs.tsinghua.edu.cn, jietang@tsinghua.edu.cn, peterli@163.com.*
- *Q. Luo is with the Hong Kong University of Science and Technology, Room 3554 (via lifts 25,26), Clear Water Bay, Kowloon, Hong Kong. E-mail: luo@cse.ust.hk.*

Manuscript received 6 Dec. 2007; revised 4 May 2008; accepted 8 Sept. 2008; published online 18 Sept. 2008.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-2007-12-0585.
Digital Object Identifier no. 10.1109/TKDE.2008.202.

determine in what cases, a single strategy method should be used, and in what cases, a combination method should be used. Moreover, in a combination, there lacks a systematic way to determine to what degree, each strategy should impact the alignment result.

Based on these considerations, we extend our previous work [60] and propose a dynamic multistrategy ontology alignment framework, which is still named RiMOM. RiMOM was a multiple strategy ontology alignment framework based on risk minimization of Bayesian decision [60]. It employs multiple ontology alignment strategies and sets the combination weight by manual. In the new version of RiMOM proposed in this paper, given two input ontologies at runtime, it automatically determines, which ontology alignment methods to be used, what kinds of information to use in the similarity calculation and how to combine multiple methods as necessary. This paper aims at formalizing a dynamic multistrategy ontology alignment framework in an analytic and systematic way. Specifically, we make the following contributions:

1. We formalize the problem of dynamic multistrategy selection in the ontology alignment and define the major tasks in dealing with the problem.
2. We define two similarity factors, which quantitatively estimate the similarity characteristics between two ontologies. We propose our dynamic multistrategy selection method based on the two similarity factors.
3. We propose a comprehensive framework to dynamically select and combine individual ontology alignment strategies, considering both the textual and structural similarity metrics of two ontologies.

We implemented RiMOM and participated in the 2006 and 2007 campaign of the Ontology Alignment Evaluation Initiative (OAEI 2006 and OAEI 2007) [19]. On the benchmark data set, RiMOM achieves the best results among the nine participants in OAEI 2006 and takes the third place in OAEI 2007 [40], [20].

The rest of this paper is organized as follows: In Section 2, we give a formal definition of the ontology and ontology alignment and formalize the major tasks in the dynamic multiple strategy ontology alignment. In Section 3, we give an overview of our framework RiMOM. In Section 4, we describe the strategies in RiMOM. In Section 5, we present our strategy selection method. In Section 6, we give the experimental results. Finally, before concluding the paper, we review the related work.

2 Ontologies and Ontology Alignment

In this section, we give the definitions related to ontologies and ontology alignment.

2.1 Ontology

Definition 1. Ontology. *An ontology is a formal specification of a shared conceptualization [26]. We describe the ontology as a 6-tuple:*

$$O = \{C, P, H^C, H^P, A^O, I\},$$

where C and P are the sets of concepts and properties, respectively. H^C defines the hierarchical relationships $H^C \subset C \times C$. $(c_i, c_j) \in H^C$ denotes that concept c_i is the subconcept of c_j. Similarly, H^P defines the hierarchical relationships between each property and its subproperties, $H^P \subset P \times P$. A^O is a set of axioms. I is a set of instances of concepts and properties.

Some standard languages, such as the Web Ontology Language (OWL) [50], describe ontologies. OWL provides vocabularies to define the formal semantics of ontology. It uses *owl:Class* and *rdfs:subClassOf* to define the concepts and subconcepts, *rdfs:Property* and *rdfs:subPropertyOf* to define property and subproperties and use *rdfs:domain* and *rdfs:range* of a property to define what concepts can have the property and what instances of the concepts can be the values of the property.

We refer to entities in an ontology as concepts, properties and instances, and define them based on OWL.

$Meta(e)$ is a set of words describing the metadata of entity e, such as its name, label, and comment:

$$Meta(e) = \{w_j | j \in [1, N_m],\ words\ occouring\ in\ the\ metadata\ of\ e\}, \tag{1}$$

where $e \in C \cup P \cup I$.

$Hier(e)$ denotes a set of subconcepts of concept e or a set of subproperties of property e:

Given concept $c \in C$,

$$Hier(\text{c}) = \{c_i | i \in [1, N_{hc}],\ sub\ concepts\ of\ c\}. \tag{2}$$

Given property $p \in P$,

$$Hier(p) = \{p_i | i \in [1, N_{hp}],\ sub\ properties\ of\ p\}. \tag{3}$$

$Inst(e)$ is a set of instances of concept e or a set of instances of property e.

Given concept $c \in C$,

$$Inst(c) = \{i_j | j \in [1, N_{ic}],\ instances\ of\ c\}. \tag{4}$$

Given property $p \in P$,

$$Inst(p) = \{i_j | j \in [1, N_{ip}],\ instances\ of\ p\}. \tag{5}$$

$Rest(c)$ is a set of properties and concepts in which each property is a property of concept c, and each concept is used to describe concept c:

$$Rest(c) = \left\{ e_j \middle| \begin{array}{l} i \in [1, N_{rc}],\ properties\ or\ concepts \\ used\ to\ restrict\ concept\ c,\ excluding \\ its\ hierarchical\ relationships \end{array} \right\}. \tag{6}$$

$Doma(p)$ is a set of concepts that have the property p:

$$Doma(p) = \{c_j | j \in [1, N_{dp}],\ concepts\ having\ the\ property\ p\}. \tag{7}$$

$Rang(p)$ is a set of concepts, whose instances can be the value of the property p:

$$Rang(p) = \left\{ c_j \middle| \begin{array}{l} j \in [1,\ N_{rp}],\ concepts\ whose\ instances \\ can\ be\ the\ values\ of\ property\ p \end{array} \right\}. \tag{8}$$

```
<owl:Class rdf:ID="Publisher">
   <rdfs:subClassOf rdf:resource="#Institution"/>
   <rdfs:label xml:lang="en">Publisher</rdfs:label>
   <rdfs:comment xml:lang="en">The publisher of books or
journals.</rdfs:comment>
</owl:Class>
<owl:Class rdf:ID="Monograph">
   <rdfs:subClassOf rdf:resource="#Book"/>
   <rdfs:label xml:lang="en">Monograph</rdfs:label>
   <rdfs:comment xml:lang="en">A book that is a single entity,
as opposed to a collection.</rdfs:comment>
   <rdfs:subClassOf>
      <owl:Restriction>
         <owl:onProperty rdf:resource="#chapters"/>
         <owl:allValuesFrom rdf:resource="#Chapter"/>
      </owl:Restriction>
   </rdfs:subClassOf>
</owl:Class>
<owl:ObjectProperty rdf:ID="proceedings">
   <rdfs:domain rdf:resource="#InProceedings"/>
   <rdfs:range rdf:resource="#Proceedings"/>
   <rdfs:subPropertyOf rdf:resource="#isPartOf"/>
   <rdfs:label xml:lang="en">proceedings</rdfs:label>
   <rdfs:comment xml:lang="en">A reference to the proceedings
in which the entry appears.</rdfs:comment>
</owl:ObjectProperty>
<foaf:Person rdf:about="#a971541439">
   <rdfs:label>Alberto Trombetta</rdfs:label>
   <foaf:name             rdf:datatype="http://www.w3.org/2001/
XMLSchema#string">Alberto Trombetta</foaf:name>
</foaf:Person>
```

Fig. 1. Snippet of ontology definition.

All the above definitions indicate a set of elements. The notation N_x denotes the number of elements in the corresponding set.

Definition 2. Concept description-$Description(c)$. *A concept $c \in C$ is described by a 4-tuple:*

$$Description(c) = \{Meta(c),\ Hier(c),\ Rest(c),\ Inst(c)\}. \tag{9}$$

Definition 3. Property description-$Description(p)$. *A property $p \in P$ is described by a 5-tuple:*

$$Description(p) = \left\{ \begin{array}{l} Meta(p),\ Hier(p), \\ Doma(p),\ Rang(p),\ Inst(p) \end{array} \right\}. \tag{10}$$

Fig. 1 shows a snippet of an example ontology extracted from the benchmark data set of OAEI 2006. We have the following information:

1. "Publisher" and "Monograph" are two concepts and the subconcepts of "Institute" and "Book."
2. "Monograph" has the property of "chapters."
3. "Proceedings" is a property, and the subproperty of "isPartof."
4. Property "proceedings" has the domain of "InProceedings" and the range of "Proceedings."
5. Attributes "name," "label," and "comment" contain the metadata.
6. "#a971541439" refers to a person instance. The person's name is "Alberto Trombetta."

2.2 Ontology Alignment

Ontology alignment takes two ontologies as input and determines as the output the alignment result between entities of the input ontologies.

Definition 4. Ontology alignment. *Given two ontologies O_1 and O_2, an alignment (or alignment task) finds, for each entity in O_1, a corresponding entity in O_2. O_1 is called the* source ontology *and O_2 the* target ontology*.*

In this paper, we deal with 1:1 alignment, i.e., for an entity in the source ontology, find at most one entity in the target ontology. Further, we do not differentiate between ontology alignment and ontology matching.

Adapting from OAEI [19], we formally define an ontology alignment result as

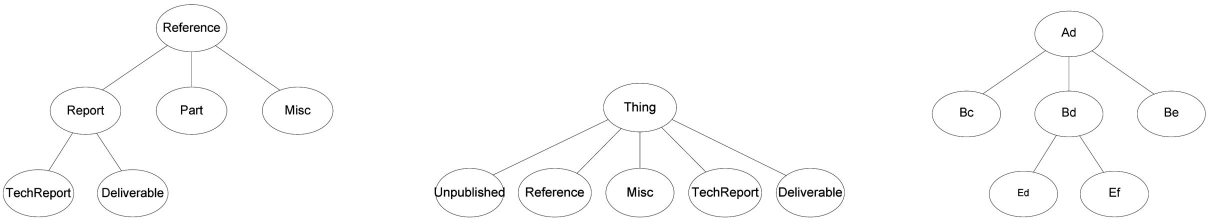
$$Align(O_1, O_2) = \left\{ \begin{array}{l} (e_{i1}, e_{i2}, con_i, relation_i) | \\ e_{i1} \in O_1, e_{i2} \in O_2, con_i \in [0,1], \\ relation_i \in \{exact, narrower, broader, overlap\} \end{array} \right\}. \tag{11}$$

Each 4-tuple $(e_{i1}, e_{i2}, con_i, relation_i)$ in $Align(O_1, O_2)$ represents that entity e_{i1} in O_1 is aligned to entity e_{i2} in O_2 with the confidence con_1 and the alignment type $relation_i$. The alignment type can be exact alignment ($exact$), narrowing alignment ($narrower$: e_{i1} is a subentity of e_{i2}), broadening alignment ($broader$: e_{i1} is the superentity of e_{i2}) and partially overlapping alignment. con_i is a numeric value. The higher the con_i value, the more reliable the alignment.

Fig. 2 shows three example ontologies. We use ontology Fig. 2a as the target ontology, Figs. 2b and 2c as source ontologies. The alignment results from (b) to (a) and from (c) to (a) are shown in Table 1. "None" in the table denotes that there is no aligned entity in the target ontology.

2.3 Dynamic Multistrategy Ontology Alignment

The goal of dynamic multistrategy selection in ontology alignment is to detect: for a specific alignment task, which strategy should be used and how confident we should be



(a) (b) (c)

Fig. 2. Fragments of three ontologies to be aligned.

TABLE 1
Alignment Result Examples

Alignment from (b) to (a)

Entity in (b)	Mapped entity	Confidence	Relation
Reference	Reference	1.0	exact
Thing	None		
TechReport	TechReport	1.0	exact
Deliverable	Deliverable	1.0	exact
Unpublished	None		
Misc	Misc	1.0	exact

Alignment from (c) to (a)

Entity in (c)	Mapped entity	Confidence	Relation
Ad	Reference	1.0	exact
Bd	Report	1.0	exact
Ed	TechReport	1.0	exact
Ef	Deliverable	1.0	exact
Bc	Part	1.0	exact
Be	Misc	1.0	exact

with a selected strategy. We define the major tasks in dynamic multistrategy ontology alignment as follows:

1. *Definition of the criteria for strategy selection*. Find criteria to quantitatively characterize the ontologies to be aligned.
2. *Dynamic selection of multiple strategies*. Select strategies for alignment and determine how to combine the selected strategies.

The main challenge is in the dynamicity; in particular, we need to strike a balance between efficiency and effectiveness so that the selection procedure is sufficiently accurate yet reasonably fast.

2.4 Similarity Factors between Two Ontologies

In this paper, we use the similarity between two entities to denote the alignment confidence con_i. As we know, different ontologies have different characteristics. For two ontologies O_1 and O_2, we define two similarity metrics, label similarity factor $F\_LS(O_1, O_2)$ and structure similarity factor $F\_SS(O_1, O_2)$, their values range from 0 to 1.

Definition 5. Label similarity factor. *The label similarity factor describes the similarity between two ontologies based on the entities' names:*

$$F\_LS(O_1, O_2) = \frac{\#\, iden\_conc\_label + \#\, iden\_prop\_label}{\max(|C_1| + |P_1|, |C_2| + |P_2|)}, \tag{12}$$

where $|C_1|$ and $|C_2|$, $|P_1|$ and $|P_2|$ represent the number of concepts and the number of properties in O_1 and O_2, respectively. #iden_conc_label and #iden_prop_label represent the number of identical name pairs in the concept's and property's names of two ontologies:

$$\#\, iden\_conc\_label = \left|\left\{ (c_i, c_j) \,\middle|\, \begin{array}{l} c_i \in C_1, c_j \in C_2, \exists w_i \in Meta(c_1), \\ \quad w_j \in Meta(c_2), w_i = w_j \\ w_i\ and\ w_j\ are\ the\ words\ in\ the \\ \quad names\ of\ c_i\ and\ c_j \end{array} \right\}\right|, \tag{13}$$

$$\#\, iden\_prop\_label = \left|\left\{ (p_i, p_j) \,\middle|\, \begin{array}{l} p_i \in P_1, p_j \in P_2, \exists w_i \in Meta(p_1), \\ \quad w_j \in Meta(p_2), w_i = w_j, \\ w_i\ and\ w_j\ are\ the\ words\ in\ the \\ \quad names\ of\ p_i\ and\ p_j \end{array} \right\}\right|. \tag{14}$$

Take Fig. 2 as an example. $F\_LS(a, b) = 4/6$, and $F\_LS(a, c) = 0/6$. These values indicate that ontologies Figs. 2a and 2b have similar label description, while ontologies Figs. 2a and 2c have very different label description.

Definition 6. Structure similarity factor. *The structure similarity factor evaluates the similarity of two ontologies based on their structure information:*

$$F\_SS(O_1, O_2) = \frac{(\#\, comm\_nonl\_conc + \#\, comm\_nonl\_prop)}{\max(\#\, nonl\_C_1 + \#\, nonl\_P_1, \#\, nonl\_C_2 + \#\, nonl\_P_2)}. \tag{15}$$

$\#nonl\_C_1$ denotes the number of concepts in O_1 that have subconcepts and likewise for $\#nonl\_C_2$. #comm_nonl_conc is calculated as follows: if concepts $c_1 \in C_1$ and $c_2 \in C_2$ have the same number of subconcepts and the same path length from the root concept to them in the ontology, then we add one to #comm_nonl_conc. After enumerating all pairs, we obtain the final score of #comm_nonl_conc:

$$\#\, nonl\_C_1 = |\{c_i | c_i \in C_1,\ Hier(c_i) \neq \phi\}|, \tag{16}$$

$$\#\, nonl\_C_2 = |\{c_i | c_i \in C_2,\ Hier(c_i) \neq \phi\}|, \tag{17}$$

$$\#\, comm\_nonl\_conc(C_1, C_2) = \left|\left\{ (c_i, c_j) \,\middle|\, \begin{array}{l} |Hier(c_i)| = |Hier(c_j)| \neq 0, c_i \in C_1, c_j \in C_2, \\ length(root_1, c_i) = length(root_2, c_j) \end{array} \right\}\right|. \tag{18}$$

$length(root, e)$ is the path length from the root entity to entity e in its hierarchical structure. Similarly, we can calculate $\#nonl\_P_1$, $\#nonl\_P_2$, and #comm_nonl_prop.

Again take Fig. 2 as an example. $F\_SS(a, b) = 0/2 = 0$, and $F\_SS(a, c) = 2/2 = 1$. It shows that ontology Fig. 2a and ontology Fig. 2b have different structures while ontology Fig. 2a and ontology Fig. 2c have similar structures.

3 Similarities and Overview of RiMOM

The core of ontology alignment is to find semantically corresponding entities from the input ontologies. In this section, we first examine similarity measures between entities, as well as the similarity characteristics between two ontologies, and then present the overview of our dynamic multistrategy ontology alignment framework.

3.1 Entity Similarity and Two Similarity Factors

The similarity between two entities is the foundation of ontology alignment. The higher the similarity of two entities, the more likely the two entities to be aligned. We

TABLE 2
Ontology Similarity Example

Ontology	F_LS	F_SS
#104	1.00	0.90
#201	0.05	1.00
#205	0.20	1.00
#221	1.00	0.00
#222	0.95	0.83
#233	1.00	0.00
#254	0.10	0.00
#301	0.20	0.00
#302	0.61	0.33
#304	0.80	0.54

denote the similarity of two entities e_1 and e_2 as $sim(e_1, e_2)$. For two concepts $e_1 \in C_1$ and $e_2 \in C_2$, $sim(e_1, e_2)$ is defined as the combination of $sim\_Meta(e_1, e_2)$, $sim\_Hier(e_1, e_2)$, $sim\_Rest(e_1, e_2)$, and $sim\_Inst(e_1, e_2)$, which in turn denote the similarities between $Meta(e_1)$ and $Meta(e_2)$, between $Hier(e_1)$ and $Hier(e_2)$, between $Rest(e_1)$ and $Rest(e_2)$, and between $Inst(e_1)$ and $Inst(e_2)$, respectively:

$$sim(e_1, e_2) = f\begin{pmatrix} sim\_Meta(e_1, e_2), sim\_Hier(e_1, e_2), \\ sim\_Rest(e_1, e_2), sim\_Inst(e_1, e_2) \end{pmatrix}. \tag{19}$$

Similarly, for two properties $e_1 \in P_1$ and $e_2 \in P_2$, $sim(e_1, e_2)$ is defined as

$$\begin{aligned} &sim(e_1, e_2) \\ &= f\begin{pmatrix} sim\_Meta(e_1, e_2), sim\_Hier(e_1, e_2), \\ sim\_Doma(e_1, e_2), sim\_Rang(e_1, e_2), sim\_Inst(e_1, e_2) \end{pmatrix}, \end{aligned} \tag{20}$$

where the five component similarities represent the similarities of two properties in metadata, hierarchy structure, the description of domain and range of property, and the instances, respectively.

We use the similarity factors defined in Section 2.4 to characterize the ontologies similarity. We studied similarities between each of the 10 ontologies and the so-called reference ontology in the OAEI benchmark data set (cf., Table 2). In Table 2, we see that the similarity characteristics of different ontology pair vary largely.

Again, in Fig. 2, the label similarity between ontologies Figs. 2a and 2b, $F\_LS(a, b)$ is nonzero, and the structure similarity $F\_SS(a, b)$ is zero. This indicates that the alignment result of ontologies Figs. 2a and 2b should rely more on the similarity in the label information than in the structure. In comparison, $F\_LS(a, c)$ is zero, and $F\_SS(a, c)$ is not; therefore, the alignment between ontologies Figs. 2a and 2c should more rely on the structure information.

As can be seen from the examples, we can use F_LS and F_SS as the indicators of the effectiveness of ontology alignment strategies. If F_LS is large, it suggests that there is much lexical overlap in the entity description, which implies that the label based strategies are more reliable; if F_SS is large, it suggests that the two ontologies have similar structures, which indicates the structure-based strategies are more effective. In this paper, we propose the strategy selection to select different ontology strategies in the ontology alignment framework.

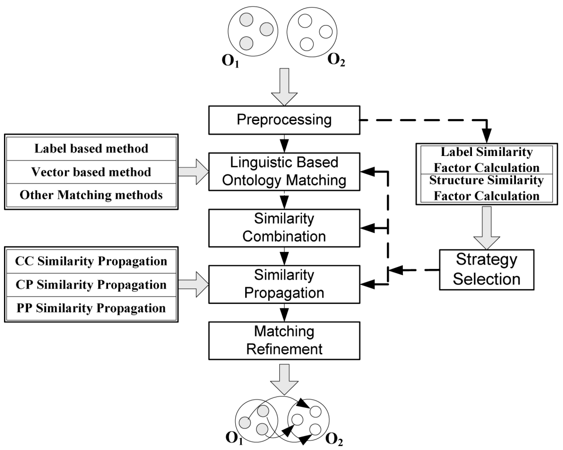

O1
O2
Preprocessing
Label based method
Vector based method
Other Matching methods
Linguistic Based Ontology Matching
Label Similarity Factor Calculation
Structure Similarity Factor Calculation
Similarity Combination
CC Similarity Propagation
CP Similarity Propagation
PP Similarity Propagation
Similarity Propagation
Strategy Selection
Matching Refinement
O1
O2

Fig. 3. RiMOM alignment processing flow.

3.2 Overview of RiMOM

Fig. 3 illustrates the processing flow of RiMOM:

1. *Preprocessing.* Given two ontologies, RiMOM generates the description for each entity. Then, it calculates the two similarity factors, which will be used in the following steps.
2. *Linguistic-based ontology alignment.* In this step, multiple linguistic-based strategies are executed. Each strategy uses different ontological information and obtains a similarity result for each entity pair. These strategies will be dynamically selected to be included in different alignment tasks.
3. *Similarity combination.* This step combines the similarity results obtained by the selected strategies. The weights in the combination are determined by the two similarity factors.
4. *Similarity propagation.* This step considers structural similarity. We use three similarity propagation strategies, namely, Concept-to-Concept, Property-to-Property, and Concept-to-Property.
5. *Alignment generation and refinement.* This step fine tunes and outputs the alignment result.

As shown in Fig. 3, strategy selection is used in three of the five steps: Step 2, Step 3, and Step 4. It determines what information should be used in a linguistics-based strategy, the combination weights in similarity combination and the similarity propagation strategy.

4 Ontology Alignment Strategies in RiMOM

Many ontology alignment strategies have been proposed. In principle, most of them can be incorporated into our framework. We classify these strategies into two categories, linguistic based and structure based. We present a few that performed well in our experiments.

4.1 Linguistic-Based Strategies

4.1.1 Edit-Distance-Based Strategy

Given two entities, e_1 for O_1 and e_2 for O_2, we first define $sim\_Name(w_1, w_2)$ using word edit distance, where w_1 and w_2 are the words in the names of two entities. Then, we define $sim\_Name(e_1, e_2)$, which denotes the similarity between two entities. The calculation of $sim\_Name(w_1, w_2)$ and $sim\_Name(e_1, e_2)$ can refer to our paper [60].

4.1.2 Vector Distance (VD)-Based Strategy

The edit-distance-based strategy makes use of the information of an entity's name. There is also other useful context information in ontology such as comments and instances of entities. We use a vector to represent this kind of context information and propose a VD-based strategy. Different from other vector-based strategy, we construct the content of vector in a dynamic way according to the similarity characteristics of ontologies.

For each entity e in the ontology O, we view its context information as a document $D(e)$. The text in $D(e)$ is tokenized into words with stemming and stop word removal. We construct $D(e)$ for concept entity $e \in C$ as

$$D(e)=\left\{\begin{array}{l}(w_1, count(w_1)),\ldots,(w_n, count(w_n))\\ \left| w_i \in \left\{\begin{array}{l} Meta(e)\cup \bigcup\limits_{e_j\in Hier(e)} Meta(e_j) \\ \cup \bigcup\limits_{e_j\in Rest(e)} Meta(e_j)\cup \bigcup\limits_{e_j\in Inst(e)} Meta(e_j)\end{array}\right\}\right.\end{array}\right\}. \tag{21}$$

According to (21), the words in $D(e)$ include those in the metadata of the concept c, its properties, subconcepts, and instances.

For property entity $e \in P$,

$$D(e) = \left\{\begin{array}{l}(w_1, count(w_1)),\ldots,(w_n, count(w_n))\\ \left| w_i \in \left\{\begin{array}{l} Meta(e)\cup \bigcup\limits_{e_j\in Hier(e)} Meta(e_j)\cup \bigcup\limits_{e_j\in Inst(e)} Meta(e_j) \\ \cup \bigcup\limits_{e_j\in Doma(e)} Meta(e_j)\cup \bigcup\limits_{e_j\in Rang(e)} Meta(e_j)\end{array}\right\}\right.\end{array}\right\}. \tag{22}$$

$D(e)$ consists of the words in the metadata of property e, the concepts connected to e, and instances of e.

Then, we construct a weighted feature vector using $tf * idf$ where tf_i is the frequency of word w_i occurring in $D(e)$, denoted as $count(w_i)$, and idf is the inverse of the number of documents containing the word w_i. In this way, each entity in source ontology O_1 and the target ontology O_2 is converted into a corresponding weighted feature vector $V(e_1)$ and $V(e_2)$, respectively.

The similarity between two entities e_1 and e_2 $sim\_Vec(e_1, e_2)$ is then calculated as the cosine of the two vectors. For each entity e_1, we calculate $sim\_Vec(e_1, e_2)$ for each e_2 and select the entity e_2 with the maximal similarity value as the candidate alignment entity of e_1.

This VD-based strategy provides us with the flexibility of using different kinds of information available in ontologies. For example, we can remove $Hier(e_2)$ in the generation of the document or add some other features. This flexibility enables us to dynamically select different information for different alignment tasks.

4.2 Structure-Based Strategies

The structural information is useful for finding alignments when two ontologies share similar structures. The intuition behind is: if two entities from ontologies O_1 and O_2 are similar to each other, then the similarity of their related entities is increased.

To exploit the structure information, we use an adaptive variation of the similarity flooding (SF) [46] for ontology alignment. Two main processes in the method are pairwise connectivity graph (PCG) construction and similarity propagation. Specifically, we represent each ontology to be aligned as a directed labeled graph (DLG). Each edge in a DLG is represented as a triple (s, p, o), where s and o are the source and target nodes, and p is the label of the edge (relation). Two DLGs are then converted to a PCG:

$$\begin{aligned}((x,y),p,(x',y')) \in \mathrm{PCG}(A,B) \Leftrightarrow (x,p,x')\\ \in A \text{ and } (y,p,y') \in B.\end{aligned}$$

Each node in the PCG represents a candidate alignment pair between the two DLGs. Based on the PCG, we can construct a similarity propagation graph (SPG). Each edge in the SPG is associated with a weight that indicates how much the similarity of a given matching pair would be propagated to the neighborhood matching pairs.

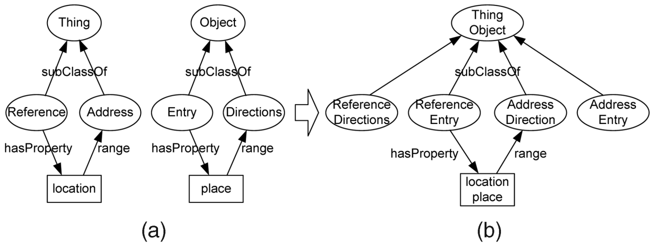


Fig. 4. Example of DLG_O and SPG_O.

The similarity propagation starts from initial similarities between nodes of two DLGs and runs an iterative propagation in the SPG. The iteration stops when no similarity changes or after a predefined number of steps.

4.2.1 Construction of DLG_O and SPG_O

In ontology alignment task, we use DLG_O and SPG_O to represent the DLG and SPG described in [46], respectively. We use $NO(O)$ to denote the nodes in $DLG\_O(O)$. The edges in $DLG\_O(O)$ come from the ontological structure information including $HasSubConcept$, $HasSibling$, $HasProperty$, $HasRange$, and $HasSubProperty$.

For concept $c \in NO(O)$, there are the following edges:

- $(c, HasSubConcept, c_s(i))$, for $c_s(i) \in NO(O) \cap Hier(c)$, $i = 1, 2, \ldots, |Hier(c)|$;
- $(c, HasProperty, p_c(i))$, for $p_c(i) \in NO(O) \cap Rest(c)$, $i = 1, 2, \ldots, |Rest(c)|$; and
- $(c_s(i), HasConceptSibling, c_s(j))$, for $c_s(i) \in NO(O) \cap Hier(c)$, $c_s(j) \in NO(O) \cap Hier(c)$, and $c_s(i) \neq c_s(j)$, $i, j = 1, 2, \ldots, |Hier(c)|$.

For property $p \in NO(O)$, the possible edges are

- $(p, HasSubProperty, p_s(i))$, for $p_s(i) \in NO(O) \cap Hier(p)$, $i = 1, 2, \ldots, |Hier(p)|$;
- $(p, HasRange, p_c(i))$, for $p_c(i) \in NO(O) \cap Rang(p)$, $i = 1, 2, \ldots, |Rang(p)|$; and
- $(p_s(i), HasPropertySibling, p_s(j))$, for $p_s(i) \in NO(O) \cap Hier(p)$, $p_s(j) \in NO(O) \cap Hier(p)$, and $p_s(i) \neq p_s(j)$, $i, j = 1, 2, \ldots, |Hier(p)|$.

Given two ontologies O_1 and O_2, we generate $DLG\_O(O_1)$ and $DLG\_O(O_2)$. The construction of $SPG\_O(O_1, O_2)$ is the same as that of the SPG in [46].

Fig. 4 shows an example of DLG_O and SPG_O. Fig. 4a is the DLG_Os of ontologies O_1 and O_2, and Fig. 4b is $SPG\_O(O_1, O_2)$. In SPG_O, nodes are entity pairs from two

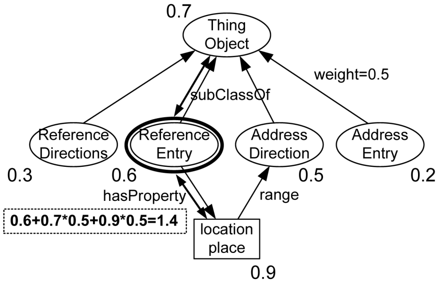


Fig. 5. Example of SF.

ontologies that have some structural relationship in common. For example, "Reference" and "Entry" are two entities in O_1 and O_2. They are constructed into a node in SPG_O because they share the same relationship "*HasProperty*."

4.2.2 Similarity Flooding in Ontology Alignment

In SF, for node (x, y) in SPG_O, $\sigma(x, y)$ is used to denote the similarity between x and y. We chose

$$\sigma^{i+1} = \frac{1}{z}\left(\sigma^0 + \sigma^i + \varphi(\sigma^0 + \sigma^i)\right), \tag{23}$$

$$\varphi(\sigma^0 + \sigma^i) = \sum_{j=1}^{m} w_j \sigma_j^i, \tag{24}$$

$$z = \max_{x' \in SPG\_O}(\sigma^{i+1}), \tag{25}$$

as the iteration equation to perform similarity propagation. σ^0, σ^i, and σ^{i+1} are similarities at the initial time, the $i\text{th}$ and the $(i+1)\text{th}$ iterations, respectively. σ^0 is the similarity between two entities calculated by any ontology alignment strategy or combination of multiple strategies. $\varphi()$ is the function to calculate the increase by considering the similarities of related entities in the $(i+1)\text{th}$ iteration. z is a normalization factor defined in (25).

σ_j^i is the entity similarity of node (x', y') connected to node (x, y) in SPG_O through property p. We simply define the weight w_j of each edge $((x, y), p, (x', y'))$ as the inverse of the number of out-linking relationships for the source node (x, y).

Fig. 5 shows an iteration result for node (Reference, Entry). It uses the similarity of its neighboring nodes (Thing, Object) and (location, place). The similarity between "Reference" and "Entry" after this iteration becomes 1.4 before the normalization.

5 Strategy Selection

Strategy selection is aimed at improving the alignment accuracy for each individual alignment task by dynamically composing the "right" strategies for it. The two similarity factors we define play a key role in strategy selection. Strategy selection works throughout the alignment process, including linguistic alignment, similarity combination, and similarity propagation.

5.1 Feature Selection in Vector Distance-Based Strategy

Strategy selection in the VD-based strategy is used to determine how label and structure information are used in the current alignment task. Specifically, it determines whether, and if so, what structure information is included in the virtual document.

In the VD-based strategy, if F_SS is larger than a threshold ε_1 (currently, we set it with $\varepsilon_1 = 0.9$, refer to Section 6.2.1 for details), and F_LS is smaller than another threshold ε_2 (currently, we set it with $\varepsilon_2 = 0.4$), it indicates that the two ontologies are similar in the hierarchical structure but different in label descriptions. In this case, ontology alignment can rely more on the structure information.

5.1.1 Determination of Hierarchical Information Use

In this step, hierarchical structure information is used only when F_SS is greater than the threshold ε_1. Otherwise, hierarchical information is not considered, and the virtual document of an entity is generated by removing $\cup_{e_j \in Hier(e)} Meta(e_j)$ for $e \in C$ or $p \in P$ from (21) and (22) respectively.

5.1.2 Enhancement of Structure Information

When using the hierarchical structure information, we add three types of structural features to the virtual document vector of both the source and the target ontologies. The three feature types represent the path length from the root concept, the number of properties, and the number of subconcepts of the current entity, respectively.

Take Fig. 2 as an example. In the alignment task from Figs. 2b to 2a, because $F\_SS(a, b) = 0$, the hierarchical information of the ontology will not be considered in the VD-based strategy. However, for the alignment task from Figs. 2c to 2a, because $F\_SS(a, c) = 1$, the hierarchical information, the three structure features, will be added to the entity document in the VD-based strategy.

5.2 Weight Calculation of Similarity Combination

We combine the similarities reported by different ontology alignment strategies as follows:

$$\begin{aligned} &sim(e_1, e_2) \\ &\quad = \frac{(w_{name}\sigma(sim\_Name(e_1, e_2)) + w_{Vec}\sigma(sim\_Vec(e_1, e_2)))}{(w_{name} + w_{Vec})}, \end{aligned} \tag{26}$$

where $sim(e_1, e_2)$ is the combined similarity of e_1 and e_2. w_{name} and w_{Vec} are the weights of different strategies. σ is a sigmoid function, $\sigma(x) = 1/(1 + \exp(-5(x - \alpha)))$, where α is set as 0.5 empirically.

The weights of w_{name} and w_{Vec} are determined by

$$\begin{aligned} w_{name} &= F\_LS / \max(F\_LS, F\_SS), \\ w_{vec} &= F\_SS / \max(F\_LS, F\_SS). \end{aligned}$$

When F_LS is larger than F_SS, the combination relies more on the similarity calculated using the edit-distance-based strategy. Otherwise, it relies more on the similarity calculated using the VD-based strategy.

5.3 Selection of Similarity Propagation Strategy

A good similarity propagation method could enhance the impact of structural information on the similarity between two entities. Strategy selection in similarity propagation is aimed at selecting "right" propagation strategies.

We classify edges in a DLG_O into three types: Concept-Concept(CC), Concept-Property(CP), and Property-Property(PP). CC edges include *HasSubclass* and *HasConceptSibling* relations, CP edges include *HasRange* and *HasProperty* relations, and PP edges include *HasSubproperty* and *HasPropertySibling* relations. Both CC and PP edges represent hierarchical relationships between entities, while CP edges nonhierarchical relationships. Correspondingly, we define three kinds of propagation strategies: CC similarity propagation, PP similarity propagation, and CP similarity propagation.

For strategy selection in the propagation, if the factor F_SS is larger than a threshold ε_3, then we perform propagation on all edges: PP, CC, and CP edges (currently, we set the threshold as $\varepsilon_3 = 0.25$, refer to Section 6.2.1); otherwise, we only do propagation on the CP edges.

5.4 Parameter Setting

As have discussed, there are in total three sets of parameters and thresholds in RiMOM: 1) the two similarity factors (F_LS and F_SS), 2) the two weights (w_{name} and w_{Vec}) and a smoothing factor α for similarity scores obtained from different alignment strategies, and 3) the three thresholds (ε_1, ε_2, and ε_3) in the dynamic strategy selection. For the first two sets of parameters except α, we automatically calculate them based on the characteristics of the source and the target ontology in an ontology alignment task. For the set of thresholds, we set them experimentally. Specifically, for each threshold, we varied the value from 0 to 1 with an interval 0.1, with the other thresholds fixed. Finally, we use the threshold values that resulted in the best performance on our test data. A learning-based method for automatically finding the best setting of the thresholds would be a more general solution. However, for ontology alignment, a parameter setting learned from one alignment task may be not hold in another task. How to accurately learn the parameters in an unsupervised way and further how to make the learned parameter adaptive to different alignment tasks is interesting future work.

6 Evaluation

We implemented RiMOM in Java and put it online (http://keg.cs.tsinghua.edu.cn/project/RiMOM). We used (OWL-API http://owl.man.ac.uk/index.shtml) to parse the RDF and OWL files. All experiments were carried out on a server running Windows 2003 with two Dual-Core Intel Xeon processors (2.8 GHz) and 3 Gbytes of memory.

6.1 Test Sets and Evaluation Methods

6.1.1 Benchmark Data Set in OAEI 2006

We used the test sets from OAEI 2006 [19]. Its benchmark dataset is in the domain of bibliography. Among the 52 ontologies provided in the benchmark data, one is target ontology and the rest are source ontologies. The gold standard results of each alignment task on all benchmark data are available.

The test data are systematically generated by starting from an original ontology and discarding various information in order to evaluate how an algorithm behaves when some information is missing [19]. There are seven categories of alterations:

1. *Name.* Entity names are replaced by random strings, synonyms, or other language text.
2. *Comments.* Comments are suppressed or translated to another language.
3. *Specialization hierarchy.* It can be suppressed, expended, or flattened.
4. *Instances.* They can be suppressed.
5. *Properties.* They can be suppressed.
6. *Classes.* They can be expanded or flattened.
7. Additions of four real ontologies of the same topic provided by some other organizations.

We classify the source ontologies in the benchmark data into five groups, as shown in Table 3.

TABLE 3
Description of Benchmark Data Set

Name of test sets	Test sets	Ontology characteristics	Number of ontologies
D1	101-104	Similar both in label description and hierarchy structure	4
D2	201-210	Similar in hierarchy structure	10
D3	221-247	Similar in label description	18
D4	248-266	Different in both label description and hierarchy structure	15
D5	301-304	Real world ontologies defined by different institutions	4

6.1.2 Directory and Food Data Sets in OAEI 2006

In addition to the benchmark data, we chose two other data sets from OAEI 2006 [19], directory and food, to evaluate RiMOM.

The directory data consists of real world Web site directories (similar to the open directory or Yahoo's). Each directory ontology is organized as taxonomy, with concept names in a hierarchical structure.

In the food ontology test data, there are two ontologies. One is the SKOS version of the United Nations Food (AGROVOC) thesaurus, and the other is the SKOS version of the United States National Agricultural Library (NAL) Agricultural thesaurus. There are about 16,000 terms in AGROVOC and 41,000 terms in NAL. For the data sets of directory and food ontology, the golden standard results are not publicly available. All results were evaluated by domain experts. Each participant of OAEI 2006 was also asked to evaluate part results of the other participants.

6.1.3 Evaluation Metrics

We use precision, recall, and F1-measure to evaluate the alignment results.

Precision (*P*). It is the percentage of correctly discovered alignments in all discovered alignments.

TABLE 4
Summary of Ontology Alignment Strategies

Notation	Description
NA+VA	Edit distance + Vector distance based strategies
RiMOM	Full-fledged RiMOM
RiMOM-SP	RiMOM without similarity flooding
RiMOM-SS	RiMOM without strategy selection

TABLE 5
Experimental Results of SF (Percent)

Test	RiMOM-SP			RiMOM		
set	P	R	F1	P	R	F1
D1	100.0	100.0	100.0	100.0	100.0	100.0
D2	91.5	82.3	86.7	98.5	95.3	96.9
D3	98.9	99.9	99.4	98.9	100.0	99.4
D4	89.2	36.8	52.1	90.2	54.1	67.6
D5	80.1	80.9	80.5	81.5	81.0	81.2
Overall	94.5	78.6	85.8	95.9	88.1	91.8

Recall (*R*). It is the percentage of correctly discovered alignments in all correct alignments:

$$P = \frac{|m_a \cap m_m|}{|m_a|}, \quad R = \frac{|m_m \cap m_a|}{|m_m|}, \tag{27}$$

$$F1 - measure = \frac{2 \times P \times R}{(P + R)}, \tag{28}$$

where m_a are alignments discovered by RiMOM, and m_m are alignments assigned manually.

6.2 Results on Benchmark Data Set

We first investigated the effect of the two similarity factors, F_SS and F_LS. Then, we tested contributions of SF and strategy selection, separately. Finally, we compared with a few other systems participated in OAEI 2006 and OAEI 2007. Additionally, we examined the memory expense and response time of our system.

Table 4 shows a summary of ontology alignment strategies under comparison. As shown in Table 4, NA+VA is the baseline method. In it, we employed only the edit-distance-based strategy and the VD-based strategy (cf., Section 4). We also tested RiMOM without similarity propagation (RiMOM-SP) and RiMOM without strategy selection (RiMOM-SS).

6.2.1 Effect of Similarity Factors

In our proposed system, F_SS is an important factor to the alignment performance. It is used in the VD-based strategy and in the process of SF, and their corresponding thresholds are ε_1 and ε_3, respectively. We used data sets 248-266 in benchmark data set D4 to set F_SS in the VD-based strategy and SF. Data set D4 is different from the target in both label description and hierarchical structure. Using it to set the parameters could get the representative values that are suitable for other alignment tasks. Fig. 6 shows the performance of factor F_SS in different processes. Figs. 6a and 6b show the effect of F_SS in the VD-based strategy and the effect of F_SS in SF, respectively.

As shown in Fig. 6b, when the threshold of F_SS is set to 1, all kinds of relationships are used in SF, and the precision, recall, and F1-measure are about 84 percent, 62 percent, and 71 percent, respectively. When it is set to 0, no hierarchical information is used, and the precision, recall, and F1-measure are 85 percent, 57 percent, and 68 percent. We can also see from this figure that the best value of the threshold for F_SS is 0.25 in SF, so ε_3 is set to 0.25. In the same way, the best threshold for F_SS is 0.9 ($\varepsilon_1 = 0.9$) in the VD-based strategy, at this time, F_LS is 0.4 ($\varepsilon_2 = 0.4$). Fig. 6c further shows that when F_SS is set to 0.25 in SF and F_SS is set to 0.90, we can get the largest F1 measure.

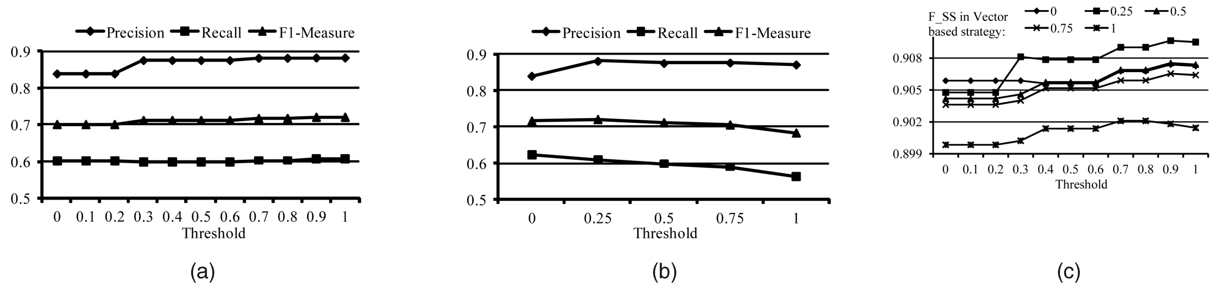
In RiMOM, F_LS is also used to determine which candidate alignment strategy will be considered in the step of alignment refinement. Our experiments show that when F_LS is less than 0.4, the pairs whose similarities are less than 0.2 are not considered as candidate alignments. We can get the best alignment result in this setting.

6.2.2 Effect of Similarity Flood-Based Strategy

With SF, some entity pairs without direct relationships or with weak relationships could be connected indirectly because of the relationship of their related entities.

Table 5 shows the effect of SF. We have following observations:

1. The overall recall is increased from 78.6 percent to 88.6 percent, a 12.7 percent improvement with SF. On the data sets of D2 and D4, the improvements are especially significant (+15.8 percent and +47.0 percent respectively on recall). This confirms that SF is effective for ontology alignment, especially when two ontologies have similar structures, as in D2 and D4.
2. When two ontologies have very similar structures, the SF method can also improve the precision. For



(a) (b) (c)

Fig. 6. (a) F_SS in VD-based strategy. (b) F_SS in SF. (c) Combined effects of F_SS.

TABLE 6
Experimental Results of Strategy Selection (Percent)

Test	RiMOM-SS			RiMOM		
set	P	R	F1	P	R	F1
D1	100.0	100.0	100.0	100.0	100.0	100.0
D2	96.2	92.9	94.5	98.5	95.3	96.9
D3	98.9	99.9	99.4	98.9	100.0	99.4
D4	85.8	49.7	62.9	90.2	54.1	67.6
D5	78.3	79.4	78.8	81.5	81.0	81.2
Overall	93.5	85.5	89.3	95.9	88.1	91.8

TABLE 7
Comparison with Other Participants

Test	1xx		2xx		3xx		H-mean	
System	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec
Refalign	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Edna	0.96	1.00	0.94	0.67	0.94	0.61	0.91	0.54
Automs	0.94	1.00	0.94	0.89	0.91	0.70	0.94	0.67
Coma	1.00	1.00	0.99	0.97	0.84	0.69	0.96	0.83
DSSim	1.00	0.98	0.99	0.67	0.90	0.78	0.98	0.55
Falcon	1.00	1.00	0.97	0.97	0.89	0.78	0.92	0.86
Hmatch	0.91	1.00	0.84	0.70	0.78	0.57	0.84	0.55
Jhuapl	1.00	1.00	0.50	0.98	0.18	0.50	0.22	0.85
OCM	0.95	1.00	0.94	0.69	0.89	0.51	0.93	0.55
Prior	1.00	1.00	0.96	0.79	0.85	0.80	0.95	0.63
RiMOM	1.00	1.00	1.00	0.98	0.83	0.82	0.96	0.88

example, on data set D2, we obtain +7.7 percent improvement on precision, while on data set D4, whose structure similarity is less than D2, the improvement on precision is little.

6.2.3 Effect of Strategy Selection

We performed experiments to test the effect of strategy selection[38], [39]. Table 6 shows the experimental results. As shown in the table, strategy selection can improve the performance of ontology alignment. The overall improvement is increased from 93.5 percent to 95.9 percent in precision (2.6 percent) and from 85.5 percent to 88.1 percent in recall (3.0 percent).

6.2.4 Comparison with Other Participants

We compare our system with other systems participated in OAEI 2006. Table 7 and Fig. 7 show the results for nine participants, the data are provided in [19].

From the results, we see that RiMOM, Falcon, and Coma are three best performers. Fig. 7a shows the precision and recall graphs of OAEI 2006 on the benchmark [19]. It shows that RiMOM can keep the highest precision in most areas of the recall.

6.2.5 Time Performance and Memory Analysis

The response time of RiMOM mainly consists of the following components:

1. calculation of the two similarity factors,
2. individual ontology strategy executions,
3. SF, and
4. postprocessing.

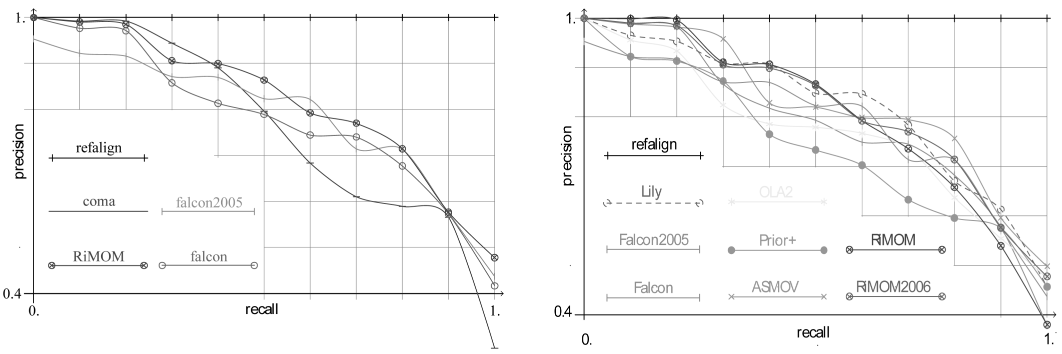
Since components 1-3 are of low computation cost, the iterative process of SF is the dominating factor in the response time. In our experiments with the benchmark data set, the system response time ranges from 0.69 second to 6.70 seconds on the benchmark tests. It indicates that RiMOM is efficient in ontology alignment tasks up to hundreds of concepts and properties. In memory analysis, we found that, for each benchmark task, roughly 50 Mbytes of memory is needed.

6.2.6 Result on OAEI 2007

RiMOM took part in OAEI 2007 benchmark task with no change in method [40]. There are 13 participants in this task. ASMOV, Lily, and RiMOM are top three teams. The precision and recall of these systems are respectively 0.95 and 0.90, 0.96 and 0.89, and 0.95 and 0.87. The result shows that RiMOM are one of the most effective method among all systems both in 2006 and in 2007 [19], [20], as shown in Fig. 7b. At the same time, RiMOM faces the challenge by the some other systems such as ASMOV, Lily, and Falcon in the benchmark task.

6.3 Results on Directory and Food Data Sets

Different from the benchmark data set, the directory datasets contain only the hierarchical and label information. Moreover, there are many synonyms in the labels. For dealing with this, we integrated an alignment strategy based on Wordnet [60]. As there is no property and instances in the directory data, we use CCP for SF. These adjustments resulted in the precision, recall, and F1-measure of 0.39, 0.40, and 0.40, respectively. The matching result is in the second place among all participants in OAEI 2006. In OAEI 2007, we further incorporated a fine-tuning process into our SF



(a) (b)

Fig. 7. Graph of the precision and recall. (a) OAEI 2006. (b) OAEI 2007.

specifically for the directory data. We thus achieved a better result, with the precision, recall, and F1-measure improved to 0.44, 0.71, and 0.55, respectively. It takes the fourth place among all participants.

We also evaluated RiMOM on the Food data in OAEI 2006 and OAEI 2007. The ontologies in the Food data are very large. For large-scale ontologies (e.g., with tens of thousands of entities), RiMOM needs a large amount of memory and a long execution time. We suppressed the structure-based strategies and applied only a simple version of the linguistic-based strategies to improve the efficiency. RiMOM consumed 1 Gbyte of memory in this task and took about eight hours to find the alignment results. Finally, RiMOM takes the second on the Food data in OAEI 2006. In OAEI 2007, we tried another approach based on the background knowledge from Wikipedia. The approach resulted in a precision of 62 percent and a recall of 42 percent.

6.4 Summary

In summary, our experiments show the following results on RiMOM:

1. *High performance.* In OAEI 2006, on the benchmark data sets, RiMOM showed the best performance; on the three tasks of the food data, RiMOM got one first and two second places in OAEI 2006; for the Food data of OAEI 2007, we tried another alignment method based on background knowledge but obtained unsatisfactory results. On the directory data, RiMOM won the second place in OAEI 2006 and the fourth place in OAEI 2007.
2. *Effectiveness of strategy selection.* The proposed strategy selection method can effectively improve the performance of ontology alignment. On the benchmark data set, the average improvement by strategy selection is +2.6 percent in precision and +3.0 percent in terms of recall. For data sets D4, where the differences of label and hierarchy are quite large, the average improvement is +5.5 percent in precision and +8.9 percent in recall.
3. *Contribution of the SF strategy.* SF can significantly improve the recall without hurting precision and can sometimes improve the precision as well. On the benchmark data set, SF improves RiMOM by +12.1 percent in recall and +1.5 percent in precision.
4. *Inefficiency for dealing with large-scale ontologies.* RiMOM still needs a large amount of memory and a long time for finding the alignments of large ontologies. How to improve the efficiency of RiMOM is also one of our ongoing work.

7 Related Work

Schema matching is a similar work to ontology alignment. There are several surveys on schema matching and ontology alignment [18], [22], [23], [31], [55], [57], [61]. Examples of research work related to schema/ontology alignment include alignment debugging [44], alignment ranking [12], ontology merging [54], and semantic data translation [1], [45]. In this section, we review the related work on schema matching, ontology alignment, and the structure-based strategies.

7.1 Schema Matching

Much research work has addressed the schema matching problem [9], [11], [33], [34], [41], [42]. Different methods, for example, similarity-based method, statistics-based method, and composite method have been proposed.

For example, COMA [9], Rondo [47], and Cupid [41] are three composite methods for schema matching. COMA is a schema matching tool supporting multiple schema types [9]. It provides a library of matching algorithms and a framework for combining matching algorithms. It allows the user to use different algorithms and combination strategies, but it is still done manually.

Rondo is a software environment for modeling engineering. It provides many unit primitives for manipulating models (e.g., extract, restrict, and delete) [47]. Rondo mainly uses entity names and taxonomy structures to determine alignments. Its recent work was focused on handling more expressive matching [3].

Cupid implements a generic schema matching algorithm combining linguistic and structural schema matching techniques. It computes the normalized similarity with the assistance of a precompiled thesaurus [41].

All of the three methods focus on how to combine different strategies so as to improve the accuracy of matching. They provide ways to adjust the weight of each strategy or to remove a strategy from the composition. However, they do not consider how to dynamically find the optimal configuration for different alignment tasks.

Some other efforts have been made for constructing a global "view" for multiple schemas. For example, Rodriguez-Gianolli and Mylopoulos have developed a tool, named DIXSE, to support the integration of XML Document Type Definitions (DTDs) into a common conceptual schema [56]. The tool integrates traditional approaches and provides a semiautomatic fashion for help the user to create the common schema.

He and Chang try to provide a unified user interface for querying multiple sources on the deep Web [27]. They propose a unified framework (MGS) for finding alignment among multiple schemas using statistical techniques.

Castano et al. propose an affinity-based unification method for global view construction [5]. The method first assesses the so-called affinity level of semantic relationship between elements in different schemas and then classifies schema elements by the affinity levels using clustering procedures; finally, constructs global views starting from selected clusters by unifying representations of their elements; see also [13], [36], and [62].

This type of work tries to construct a common schema from multiple data sources. The motivation differs in nature from ours, as we focus on dynamic selecting strategies for a given alignment task. Our framework can be adapted to dynamic strategies selection in the construction of the global view for multiple schemas.

Another type of work tries to find the interscheme between different schemas. For example, Palopoli et al. propose a method to find similarities or dissimilarities among scheme objects (called interscheme properties) [52].

The method combines both textual information and the structural information. A graph-based technique for a uniform derivation of interscheme properties including synonymies, homonymies, type conflicts, and subscheme similarities. As for the similarity strategies, the methods in [52] are close to those in our work. The difference is that we not only combine different strategies but also propose a method for automatic strategy selection.

Generally, in comparison with schema matching, ontology alignment has its own unique characteristics [43], [60]. First, comparing with database schemas, ontologies provide higher flexibility and more explicit semantics for defining data. Second, database schemas are usually defined for a specific database, whereas ontology is by nature reusable and sharable. Third, ontology development is becoming a more and more decentralized procedure. Finally, in ontology, the number of knowledge representation primitives is much larger and more complex, e.g., cardinality constraints, inverse properties, transitive properties, disjoint classes, and type-checking constraints.

As a result, in ontology alignment, we can use more and detailed information than in database schemas, for example, both hierarchical and non hierarchical information, as well as description information. It provides more choices on what information to be used in the alignment and also requires more specific considerations on each type of information. In this paper, we use such information to empower the VD-based strategy and multiple SF strategies.

7.2 Ontology Alignment and the Combination of Multiple Ontology Alignment Strategies

Existing work [28], [58] makes use of one type of ontological information or one kind of method for finding ontology alignment. Some previous method [32] uses information-flow theory or a bootstrapping method to find the alignment. These methods can obtain good results on some alignment tasks but may fail on some others as they cannot make use of all kinds of information available in ontologies. Combination of the different alignment strategies has been investigated, aiming at achieving better alignment performance.

For example, GLUE aims at automatically finding ontology alignment for data integration [10], [11]. It uses machine learning techniques to combine different alignment methods. Specifically, it first applies statistical analysis on distributions to generate a similarity matrix. Next, it uses "constraint relaxation" to obtain an alignment.

FOAM [16] achieves high-quality results through a combination of a rule-based approach, a machine learning approach, and the intelligent selection of candidate alignments. It also provides a mechanism to let users set the parameters for a specific alignment task and select the alignment when doubtable alignments are produced.

COMA++ [2] is an advanced version of COMA [9]. It includes new approaches and offers a comprehensive infrastructure to solve large real-world match problems. In addition, COMA++ provides a user friendly interface for improving the practicability and effectiveness of the system. Based on COMA++, eTuner is proposed to automatically tune a schema matching system using synthetic schemas that have the ground-truth matching [37]. Falcon-AO [29], [53] is an automatic tool for aligning ontologies. There are two alignment strategies in Falcon-AO, LMO and GMO. LMO is a matcher based on linguistic matching for ontologies, and GMO is a matcher based on graph matching for ontologies.

Ehrig and Staab propose a Parameterizable Alignment Methods (PAM) [15]. They have developed a bootstrapping approach for acquiring the parameters of different strategies through machine learning techniques.

RiMOM differs from these combination methods in the following aspects. First, some proposed combination strategies [2], [9]], [10], [37] such as GLUE focused on the learning of the combination weights of individual alignment methods either from the training data or using a specific combination of matching results for different alignment tasks. In RiMOM, the combination weights are automatically determined by the characteristics of similarity between two ontologies in linguistic and structural information. Second, many proposed ontology alignment methods are combined after each individual alignment strategy has gotten the candidate alignment results, for example, COMA++ [2]. In comparison, RiMOM first determines what parts of information are reliable and then selects the information to be used in different ontology alignment strategies according to the characteristics of ontology alignment tasks.

A few work has been conducted for adaptive integration of different strategies and different features for a matching task, which is very relevant to our work. For example, Castano et al. propose the H-Match algorithm for performing ontology matching [6]. The H-Match algorithm can dynamically integrate different ontology features for a matching task. Different from this work, we focus on the dynamic integration of multiple matching strategies, while H-Match focuses on the dynamic integration of different features in one matching strategy.

Boukhebouze et al. further propose a method to automate tuning the combination parameters (e.g., weights of different strategies) in schema matching [4]. However, the idea is based on a strict assumption that an algorithm that obtains a good performance in one alignment task (e.g., on the benchmark) with a tuned parameter configuration will also obtain good performances in the other context. On the contrary, we more intend to find the dynamic configuration for every alignment task.

Some other efforts have been made to find alignment beyond one-to-one matching. For example, Euzenat and Valtchev define a universal measure for comparing the entities of two ontologies based on similarities of entity and its related definitions (such as superclasses, properties, instances, etc.) [17]. They propose a method to find the one-to-many relationships between entities by using local matching of entity sets and iterative computation of recursively dependent similarities. Our proposed method can be extended to dynamically detect the utility of different strategies in the one-to-many alignment context.

Giunchiglia et al. propose a method to find the "semantic" mapping between nodes of two graph-like structure (e.g., XML schemas and taxonomies) [24]. They focus on finding semantic relationships (e.g., "equivalent" and "super concept") between nodes. Different from the work, we focus on dynamically combining different strategies.

Recently, utilizing search engines to help find the alignment is another type of method [25]. For example, Gligorov et al. [25] proposes a method based on the search results from Google to find the alignment between ontologies.

7.3 Structure-Based Ontology Alignment

Structural information is proved to be very useful in ontology alignment. Many structure-based ontology alignment/schema matching methods have been investigated. The simplest method is to add the structural information into the linguistic alignment strategies. Such examples include COMA and GLUE.

Another method is to view the schema or ontology as a graph, thus the ontology alignment is converted as a task of graph matching. Such examples are GMO in Falcon and the method of SF. Falcon uses directed bipartite graphs to represent ontologies and measures the structural similarity between graphs. The input of GMO can be a set of matched pairs previously found by other approaches.

SF is proposed to propagate similarities between two entities. It takes the assumption that, if two entities are similar, the entities near them are also similar. It runs an iteration procedure to reflect the influence of similarities of their neighboring entities.

There are two processes in RiMOM in which structural information is exploited. It is used in different ways from the existing structure-based ontology alignment methods. First, we use hierarchical information in the VD-based strategy when the structure similarity between two ontologies is larger than a threshold. Second, most ontology alignment methods used structure information statically. In RiMOM, structure information is used dynamically in the alignment algorithm. For example, when the structure similarity is less than a threshold, only the nonhierarchical information are considered in SF. Third, RiMOM takes the information defined in an ontology and proposes three different flooding strategies for ontology alignment tasks.

There are also many methods proposed to address other issues in the ontology alignment. For example, several systems rely on semantics to find alignment [7]. QOM addresses the efficiency problem of alignment [14]. Euzenat [21] and Johnson et al. [30] focus on evaluation of ontology alignment. In addition, some works study how to align database schema to ontology, e.g., [48]; see also [35], [51], and [59].

7.4 Relationship with Other Alignment Methods

As shown in previous sections, there are some similar methods related to our proposed method. Here, we compared several most relevant methods to our framework. In particular, we compared these methods based on the information and the strategies used for ontology alignment. Specifically, the information used for finding ontology alignment includes textural content of metadata (T), structure of metadata (S), ontological instances (I), and domain background knowledge (K). Strategies designed for ontology alignment primarily include learning based (L), similarity matching based (M), and reasoning based (R). Table 8 shows the comparison of several state-of-the-art methods and our proposed method. We see that most of the methods combine the structure and textural content of metadata, and the instance information. Several of them also make use of the domain background information. As for the alignment strategies, most of those methods employ similarity matching-based strategies, some utilize machine learning for ontology alignment, and a few methods use reasoning-based strategies. However, none of these methods considers the dynamic strategy configuration for ontology alignment, which is the key difference of the proposed framework in this paper from the existing methods.

TABLE 8
Relationship with Several Classical Methods

T - metadata textural content; S - metadata structure;
I - ontological instance; K - domain knowledge.
L - learning-based; M - similarity matching-based;
R - reasoning-based; D - Dynamic strategy

Method	Used Information				Strategy			
	T	S	I	K	L	M	R	D
Coma [2] [9]	√	√	√	√		√		
Falcon [29] [53]	√	√	√			√		
GLUE [10] [11]			√		√			
PROMPT [49]	√	√				√		
Rondo [47]	√	√		√		√		
S-Match [24]	√	√				√	√	
Uniform [52]	√	√	√			√		
Pre-RiMOM [60]	√	√	√	√	√	√		
RiMOM	√	√	√			√		√

8 Conclusion

In this paper, we have proposed a multistrategy framework, RiMOM, to automatically and dynamically compose strategies for individual ontology alignment tasks. We consider both textual and structural similarities in ontologies and compose alignment strategies to be suitable for different similarity characteristics. Experimental results show that our proposed approach can significantly outperform both single strategies and statically combined methods. Furthermore, experimental results on the data sets from OAEI 2006 and OAEI 2007 demonstrate that our system performs better than most of the participants and is among the top three performers on the benchmark data sets.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (90604025, 60703059), Chinese National Key Foundation Research and Development Plan (2007CB310803), and the Chinese Young Faculty Research Funding (20070003093).

References

[1] P. Atzeni, P. Cappellari, and G. Gianforme, "MIDST: Model Independent Schema and Data Translation," *Proc. ACM SIGMOD '07 (Demonstration)*, pp. 1134-1136, 2007.

[2] D. Aumueller, H.H. Do, S. Massmann, and E. Rahm, "Schema and Ontology Matching with COMA++," *Proc. ACM SIGMOD '05 (Demonstration)*, pp. 906-908, 2005.

[3] P.A. Bernstein and S. Melnik, "Model Management 2.0: Manipulating Richer Matchings," *Proc. ACM SIGMOD '07*, pp. 1-12, 2007.

[4] M. Boukhebouze, R. Rifaieh, N. Benharkat, and Y. Amghar, "Benchmarking XML-Schema Matching Algorithms for Improving Automated Tuning," *Proc. IEEE/ACS Int'l Conf. Computer Systems and Applications (AICCSA '07)*, pp. 917-925, 2007.

[5] S. Castano, V. De Antonellis, and S. De Capitani di Vimercati, "Global Viewing of Heterogeneous Data Sources," *IEEE Trans. Knowledge and Data Eng.*, vol. 13, no. 2, pp. 277-297, Mar./Apr. 2001.

[6] S. Castano, A. Ferrara, and S. Montanelli, "Matching Ontologies in Open Networked Systems: Techniques and Applications," *J. Data Semantics V*, pp. 25-63, 2006.

[7] CROSI, "Semantic Integration Techniques Survey," http://www.aktors.org/crosi/deliverables/summary/survey.html, 2005.

[8] L. Ding, T. Finin, A. Joshi, R. Pan, R.S. Cost, and Y. Peng, "Swoogle: A Search and Metadata Engine for the Semantic Web," *Proc. Int 'l Conf. Information and Knowledge Management (CIKM '04)*, pp. 652-659, 2004.

[9] H.H. Do and E. Rahm, "Coma, A System for Flexible Combination of Schema Matching Approaches," *Proc. 28th Int 'l Conf. Very Large Databases (VLDB '02)*, pp. 610-621, 2002.

[10] A. Doan, J. Madhavan, P. Domingos, and A. Halevy, "Learning to Map between Ontologies on the Semantic Web," *Proc. 11th Int 'l Conf. World Wide Web (WWW '02)*, pp. 662-673, 2002.

[11] A. Doan, P. Domingos, and A. Halevy, "Reconciling Schemas of Disparate Data Sources: A Machine-Learning Approach," *ACM SIGMOD Record*, vol. 30, no. 2, pp. 509-520, 2001.

[12] C. Domshlak, A. Gal, and H. Roitman, "Rank Aggregation for Automatic Schema Matching," *IEEE Trans. Knowledge and Data Eng.*, vol. 19, no. 4, pp. 538-553, Apr. 2007.

[13] R. dos Santos Mello, S. Castano, and C.A. Heuser, "A Method for the Unification of {XML} Schemata," *Information & Software Technology*, vol. 44, no. 4, pp. 241-249, 2002.

[14] M. Ehrig and S. Staab, "QOM: Quick Ontology Mapping," *Proc. Int'l Conf. Semantic Web (ISWC '04)*, pp. 683-697, 2004.

[15] M. Ehrig, S. Staab, and Y. Sure, "Bootstrapping Ontology Alignment Methods with APFEL," *Proc. Int'l Conf. Semantic Web (ISWC '05)*, pp. 186-200, 2005.

[16] M. Ehrig and Y. Sure, "Foam—Framework for Ontology Alignment and Alignment; Results of the Ontology Alignment Initiative," *Proc. Workshop Integrating Ontologies*, vol. 156, pp. 72-76, 2005.

[17] J. Euzenat and P. Valtchev, "Similarity-Based Ontology Alignment in OWL-Lite," *Proc. European Conf. Artificial Intelligence (ECAI '04)*, pp. 333-337, 2004.

[18] J. Euzenat, "State of the Art on Ontology Alignment," http://www.inrialpes.fr/exmo/cooperation/kweb/heterogeneity/deli/, 2008.

[19] J. Euzenat, M. Mochol, P. Shvaiko, H. Stuckenschmidt, O. Šváb, V. Svátek, W.R. Hage, and M. Yatskevich, "First Results of the Ontology Alignment Evaluation Initiative 2006," *Proc. Int'l Workshop Ontology Matching (OM '06) collocated with the Fifth Int'l Conf. Semantic Web (ISWC '06)*, pp. 73-95, 2006.

[20] J. Euzenat, A. Isaac, C. Meilicke, P. Shvaiko, H. Stuckenschmidt, O. Šváb, V. Svátek, W.R. Hage, and M. Yatskevich, "Results of the Ontology Alignment Evaluation Initiative 2007," *Proc. Int'l Workshop Ontology Matching (OM '07) collocated with the Fifth Int'l Conf. Semantic Web (ISWC '07)*, pp. 96-132, 2007.

[21] J. Euzenat, "Semantic Precision and Recall for Ontology Alignment Evaluation," *Proc. Int'l Joint Conf. Artificial Intelligence (IJCAI '07)*, pp. 348-353, 2007.

[22] J. Euzenat and P. Shvaiko, *Ontology Matching.* Springer, 2007.

[23] F. Giunchiglia and P. Shvaiko, "Semantic Matching," *The Knowledge Eng. Rev.*, vol. 18, pp. 265-280, 2003.

[24] F. Giunchiglia, M. Yatskevich, and P. Shvaiko, "Semantic Matching: Algorithms and Implementation," *J. Data Semantics*, vol. 9, pp. 1-38, 2007.

[25] R.R. Gligorov, Z. Aleksovski, W.T. Kate, and F.V. Harmelen, "Using Google Distance to Weight Approximate Ontology Matches," *Proc. Int'l Conf. World Wide Web (WWW '07)*, pp. 767-776, 2007.

[26] T. Gruber, "A Translation Approach to Portable Ontologies," *Knowledge Acquisition*, vol. 5, no. 2, pp. 199-220, 1993.

[27] B. He and K. Chen-Chuan Chang, "Statistical Schema Matching across Web Query Interfaces," *Proc. ACM SIGMOD '03*, pp. 217-228, 2003.

[28] A. Issac, L.V. der Meij, S. Schlobach, and S. Wang, "An Empirical Study of Instance-Based Ontology Matching," *Proc. Int'l Conf. Semantic Web and Asian Semantic Web (ISWC '07+ASWC '07)*, pp. 253-266, 2007.

[29] N. Jian, W. Hu, G. Cheng, and Y. Qu, "Falcon-AO: Aligning Ontologies with Falcon," *Proc. K-CAP Workshop Integrating Ontologies (K-CAP '05)*, pp. 85-91, 2005.

[30] H.L. Johnson, K.B. Cohen, and L. Hunter, "A Fault Model for Ontology Mapping, Alignment, and Linking Systems," *Proc. Pacific Symp. Biocomputing (PSB '07)*, pp. 233-244, 2007.

[31] Y. Kalfoglou and M. Schorlemmer, "Ontology Alignment: The State of the Art," *Knowledge Eng. Rev.*, vol. 18, pp. 1-31, 2003.

[32] Y. Kalfoglou and M. Schorlemmer, "IF-Map: An Ontology Mapping Method Based on Information Flow Theory," *J. Data Semantics*, vol. 1, no. 1, pp. 98-127, 2003.

[33] J. Kang and J.F. Naughton, "On Schema Matching with Opaque Column Names and Data Values," *Proc. ACM SIGMOD '03*, pp. 205-216, 2003.

[34] W. Kim and J. Seo, "Classifying Schematic and Data Heterogeneity in Multi-Database Systems," *Computer*, vol. 24, no. 12, pp. 12-18, 1991.

[35] P. Lambrix and H. Tan, "A Tool for Evaluating Ontology Alignment Strategies," *J. Data Semantics*, vol. 8, pp. 182-202, 2007.

[36] M.L. Lee, L.H. Yang, W. Hsu, and X. Yang, "XClust: Clustering XML Schemas for Effective Integration," *Proc. Int'l Conf. Information and Knowledge Management (CIKM '02)*, pp. 292-299, 2002.

[37] Y. Lee, M. Sayyadian, A. Doan, and A.S. Rosenthal, "eTuner: Tuning Schema Matching Software Using Synthetic Scenarios," *VLDB J.*, vol. 16, no. 1, pp. 97-122, 2007.

[38] Y. Li, J. Li, D. Zhang, and J. Tang, "Result of Ontology Alignment with RiMOM at OAEI '06," *Proc. Int'l Workshop Ontology Matching (OM '07) collocated with the Fifth Int'l Conf. Semantic Web (ISWC '07)*, pp. 181-190, 2006.

[39] Y. Li, J. Li, D. Zhang, and J. Tang, "Toward Strategy Selection for Ontology Alignment," *Proc. Int'l Conf. European Semantic Web Conf. (ESWC '07)*, (Poster), 2007.

[40] Y. Li, Q. Zhong, J. Li, and J. Tang, "Result of Ontology Alignment with RiMOM at OAEI '07," *Proc. Int'l Workshop Ontology Matching (OM '07) collocated with the Fifth Int'l Conf. Semantic Web (ISWC '07)*, pp. 227-235, 2007.

[41] J. Madhavan, P. Bernstein, and E. Rahm, "Generic Schema Matching with Cupid," *Proc. Int'l Conf. Very Large Database (VLDB '01)*, pp. 48-58, 2001.

[42] J. Madhavan, P. Bernstein, K. Chen, A. Halevy, and P. Shenoy, "Corpus Based Schema Matching," *Proc. Int'l Conf. IJCAI Workshop Information Integration on the Web (IIWeb '03)*, pp. 1567-1572, 2003.

[43] A. Maedche, B. Moltik, N. Silva, and R. Volz, "MAFRA-An Ontology Alignment FRAmework for Distributed Ontologies," *Proc. Int'l Conf. Knowledge Eng. and Knowledge Management (EKAW '02)*, pp. 235-250, 2002.

[44] C. Meilicke, H. Stuckenschmidt, and A. Tamilin, "Repairing Ontology Mappings," *Proc. Int'l 32nd Conf. Artificial Intelligence (AAAI '07)*, pp. 1408-1413, 2007.

[45] S. Melnik, A. Adya, and P.A. Bernstein, "Compiling Mappings to Bridge Applications and Databases," *Proc. ACM SIGMOD '07*, pp. 461-472, 2007.

[46] S. Melnik, H.G. Molina, and E. Rahm, "Similarity Flooding: A Versatile Graph Matching Algorithm," *Proc. 18th Int'l Conf. Data Eng. (ICDE '02)*, pp. 117-128, 2002.

[47] S. Melnik, E. Rahm, and P.A. Bernstein, "Rondo: A Programming Platform for Model Management," *Proc. ACM SIGMOD '03*, pp. 193-204, 2003.

[48] B. Motik, I. Horrocks, and U. Sattler, "Bridging the Gap between OWL and Relational Databases," *Proc. Int'l Conf. World Wide Web (WWW '07)*, pp. 807-816, 2007.

[49] N.F. Noy, "Semantic Integration: A Survey of Ontology Based Approaches," *SIGMOD Record*, special section on semantic integration, vol. 33, no. 4, pp. 65-70, 2004.

[50] OWL, *Ontology Web Language*, http://www.w3.org/TR/owl-features/, 2008.

[51] L. Palopoli, G. Terracina, and D. Ursino, "The System DIKE: Towards the Semi-Automatic Synthesis of Cooperative Information Systems and Data Warehouses," *Proc. ADBIS-DASFAA Symp. Advances in Databases and Information Systems*, pp. 108-117, 2000.

[52] L. Palopoli, D. Sacca, G. Terracina, and D. Ursino, "Uniform Techniques for Deriving Similarities of Objects and Subschemes in Heterogeneous Databases," *IEEE Trans. Knowledge and Data Eng.,* vol. 15, no. 2, pp. 271-294, Mar./Apr. 2003.

[53] Y. Qu, W. Hu, and G. Cheng, "Constructing Virtual Documents for Ontology Matching," *Proc. Int'l Conf. World Wide Web (WWW '06),* pp. 23-31, 2006.

[54] C. Quix, D. Kensche, and X. Li, "Generic Schema Merging," *Proc. Int'l Conf. Advanced Information Systems Eng. (CAiSE '07),* pp. 132-148, 2007.

[55] E. Rahm and P.A. Bernstein, "A Survey of Approaches to Automatic Schema Matching," *VLDB J.,* vol. 10, pp. 334-350, 2001.

[56] P. Rodriguez-Gianolli and J. Mylopoulos, "A Semantic Approach to XML-Based Data Integration," *Proc. Int'l Conf. Conceptual Modeling (ER '01),* pp. 117-132, 2001.

[57] P. Shvaiko and J. Euzenat, "A Survey of Schema-Based Matching Approaches," *J. Data Semantics,* vol. 4, pp. 146-171, 2005.

[58] G. Stoilos, G. Stamou, and S. Kollias, "A String Metric for Ontology Alignment," *Proc. Int'l Conf. Semantic Web (ISWC '05),* pp. 624-637, 2005.

[59] O. Šváb, V. Svátek, and H. Stuckenschmidt, "A Study in Empirical and "Casuistic" Analysis of Ontology Mapping Results," *Proc. Int'l European Conf. Semantic Web (ESWC '07),* pp. 655-669, 2007.

[60] J. Tang, J. Li, B. Liang, X. Huang, Y. Li, and K. Wang, "Using Bayesian Decision for Ontology Alignment," *J. Web Semantics,* vol. 4, no. 4, pp. 243-262, 2006.

[61] H. Wache, T. Vogele, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann, and S. Hubner, "Ontology-Based Integration of Information—A Survey of Existing Approaches," *Proc. Int'l Joint Conf. Artificial Intelligence Workshop Ontologies and Information Sharing,* pp. 108-117, 2001.

[62] X. Yang, M.L. Lee, and T.W. Ling, "Resolving Structural Conflicts in the Integration of XML Schemas: A Semantic Approach," *Proc. Int'l Conf. Conceptual Modeling (ER '03),* pp. 520-533, 2003.

Juanzi Li is an associate professor in the Department of Computer Science and Technology, Tsinghua University. Her research interests include semantic Web and knowledge discovery.

Jie Tang is an assistant professor in the Department of Computer Science and Technology, Tsinghua University. His research interests are data mining and semantic Web. He is a member of the IEEE.

Yi Li received the MSc degree from the Tsinghua University.

Qiong Luo is an assistant professor in the Department of Computer Science and Engineering, Hong Kong University of Science and Technology (HKUST). Her research interests are in database systems.