Constructing Virtual Documents for Ontology Matching

Yuzhong Qu
Department of Computer Science and Engineering
Southeast University
Nanjing 210096, P. R. China
yzqu@seu.edu.cn

Wei Hu
Department of Computer Science and Engineering
Southeast University
Nanjing 210096, P. R. China
whu@seu.edu.cn

Gong Cheng
Department of Computer Science and Engineering
Southeast University
Nanjing 210096, P. R. China
gcheng@seu.edu.cn

ABSTRACT

On the investigation of linguistic techniques used in ontology matching, we propose a new idea of virtual documents to pursue a cost-effective approach to linguistic matching in this paper. Basically, as a collection of weighted words, the virtual document of a URIref declared in an ontology contains not only the local descriptions but also the neighboring information to reflect the intended meaning of the URIref. Document similarity can be computed by traditional vector space techniques, and then be used in the similarity-based approaches to ontology matching. In particular, the RDF graph structure is exploited to define the description formulations and the neighboring operations. Experimental results show that linguistic matching based on the virtual documents is dominant in average F-Measure as compared to other three approaches. It is also demonstrated by our experiments that the virtual documents approach is cost-effective as compared to other linguistic matching approaches.

Categories and Subject Descriptors

D.2.12 [**Software Engineering**]: Interoperability—*Data mapping*; I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods—*Representation languages*; I.2.7 [**Artificial Intelligence**]: Natural Language Processing—*Language generation, Language parsing and understanding*

General Terms

Algorithms, Experimentation, Measurement

Keywords

Ontology Matching, Linguistic Matching, Description Formulation, Neighboring Operation, Vector Space Model

1. INTRODUCTION

Web ontologies written in RDF [11] or OWL [20] play a crucial role in the emerging Semantic Web. However, there always exist multiple ontologies for overlapped domains and even for the same domain due to the decentralized nature of the Web. Therefore, ontology matching, or ontology alignment, is necessary to establish interoperation between Web applications using different ontologies.

Many researches have been taken to pursue good algorithms and tools for (semi-)automatic ontology matching. GLUE [4], QOM [5], OLA [6], S-Match [7], ASCO [14], PROMPT [19] and HCONE-merge [12] are such tools. Many experimental results of some tools on "standard" tests of pairwise ontologies are reported in OAEI 2005[1], EON 2004[2] and I3CON 2003[3]. Even before the emergence of the Semantic Web, there exist a number of algorithms and tools for schema matching in the field of database, e.g., Artemis [1], COMA [3], Cupid [16] and Similarity Flooding (SF) [17]. Two good surveys on the approaches to ontology or schema matching can be found in [18, 22].

In all the approaches mentioned above, linguistic information and structural information, and even domain knowledge in some cases, are exploited to find good mapping between URIrefs declared in OWL/RDF ontologies or elements in database schemas. In an ontology, linguistic information includes local name (fragment identifier) and descriptions (e.g., labels, comments and other annotations) of declared URIrefs; while structural information means the relationships among URIrefs. Except for very few ones (e.g., GLUE and S-Match), most approaches are based on similarity, i.e., they rely on some quantitative assessment of pairwise likeness between URIrefs or elements. Generally speaking, similarity-based approaches first find some similar entities by computing linguistic similarity, and then exploit the structural similarity based on the idea of "similarity propagation" [16, 17]. Although the formulation of structural similarity is a key feature of a matching approach, the fact that must be noticed is that ontology or schema matching should ground on linguistic matching.

To pursue a cost-effective approach to linguistic matching, in this paper, we investigate the state of the art of the linguistic characteristics used in ontology or schema matching, and we propose a new idea of virtual documents for linguistic matching. As stated in [28], a virtual document is simply a document for which no persistent state exists and for which some or all instances are generated at run time. Historically, virtual documents are mainly used for building dynamic Web pages or reusing existing information. Here, a virtual document, represented by a collection of weighted words, is generated for each URIref declared in an OWL/RDF ontology. Document similarity can be

WWW 2006, May 23–26, 2006, Edinburgh, Scotland.
ACM 1-59593-323-9/06/0005.

[1] http://oaei.inrialpes.fr/2005/
[2] http://km.aifb.uni-karlsruhe.de/ws/eon2004/
[3] http://www.atl.external.lmco.com/projects/ontology/i3con.html

computed by traditional vector space techniques and then be used in similarity-based approach to ontology matching. The novelty of the virtual documents proposed in this paper is as follows. The virtual document of a URIref declared in an ontology contains not only the local descriptions but also the neighboring information to reflect the intended meaning of the URIref, and a weighting scheme is incorporated to reflect the importance of information appeared in different categories. The construction of virtual documents and its usage in linguistic matching for ontologies have been primarily implemented in our tool, Falcon-AO [10]. Systematic experiments on the OAEI 2005 benchmark tests show that virtual documents together with TF/IDF [24] technique are cost-effective in linguistic matching for ontologies.

The rest of this paper is organized as follows. In Section 2, we survey the linguistic characteristics and related techniques used in ontology or schema matching. In Section 3, we propose the concept of virtual documents for linguistic matching, and present the formulation of virtual documents by making use of the RDF graph structure. The TF/IDF-based computation of similarities between virtual documents is presented in Section 4. Experimental results of different but related approaches on the OAEI 2005 benchmark tests are given and compared in Section 5. Some related work is presented in Section 6. Finally, conclusion and remarks on future work are given in Section 7.

2. LINGUISTIC CHARACTERISTICS USED IN ONTOLOGY MATCHING

We survey the linguistic techniques currently used in ontology or schema matching, with an emphasis on the linguistic characteristics adopted in existing algorithms or tools for ontology matching. Especially, we examine whether they utilize local name, descriptions and neighboring information in linguistic matching, and whether they look up synonymy and hyponymy relationships in thesaurus, and which specific techniques are employed in. The result of our investigation on representative approaches is sketched in Table 1.

Seven approaches are selected for comparison due to their distinctiveness in linguistic matching. Compared to COMA [3], SF [17] and some other approaches in schema matching, Cupid [16] exploits more sophisticated linguistic matching techniques, so we take it as a representative in the area of schema matching. In the field of ontology matching, PROMPT [19] and QOM [5] only use string comparison in linguistic matching, so they are not discussed further although they are famous in the ontology matching community. Some others [1, 23] are also omitted here because they are not distinctive in linguistic matching. More details of the seven representative approaches are described below.

Cupid is a similarity-based approach to schema matching. In [16], tokenization (parsing names into tokens based on punctuations, cases, etc.), expansion (identifying abbreviations and acronyms) and elimination (discarding prepositions, articles, etc.) are performed in a normalization step. Then, linguistic similarities are computed between schema elements by comparing the tokens normalized from their names. Beside the substring matching, a thesaurus is used to look up synonymy and hyponymy relationships.

It also points out that information retrieval (IR) techniques can be used to compare descriptions which annotate the schema elements, and that using schema annotations (textual descriptions of schema elements in the data dictionary) for linguistic matching is one of the challenges for the further work.

OLA is a similarity-based approach to ontology matching. In [6], both string distance and lexical distance are computed for the comparison between URIrefs. A variant of the substring distance is used to establish a basic similarity value. Lexical distance relies on an exploration of WordNet 2.0 with a quantitative assessment of the "relatedness" between two terms (possible multi-word). The relatedness of two WordNet entries is based on a depth-sensible measurement. The computation of multi-word similarity involves the similarity calculation between sets based on element similarities.

In the OAEI 2005 contest, as the requirement of a fixed parameter set for the entire testing families, OLA just relies on the string distance. Indeed, it achieves a better performance as well as much more efficient than the WordNet-based computation, as reported in [6].

ASCO is another similarity-based approach to ontology matching. The main idea behind ASCO [14] is to maximally use the available information contained in the ontology in the progress of discovering the mapping between classes. In ASCO, linguistic similarity is a linear combination of name similarity, label similarity and description similarity.

Jaro-Winkler metric is used for calculating string similarity between two tokens. Each name is transformed into tokens based on punctuations, upper cases, special symbols and so on. Name similarity is measured by the average of the best similarity of each token with another token in the other set. The calculation of label similarity is somewhat similar to the name similarity calculation. WordNet is used for looking up synonyms in the calculation of name similarity and label similarity.

The description similarity is measured by comparing "collections of words" in the descriptions, and the TF/IDF technique is applied. As pointed out in [14], the integration of WordNet in the calculation of description similarity may not be valuable and cost much calculating time because the descriptions are not merely a term or a short expression.

HCONE-merge is a Human Centered ONtology Engineering approach to merging ontologies [12]. Its contribution to ontology mapping is a way to map concepts specified in an ontology to word senses using Latent Semantic Analysis (LSA). Basically, for a given concept C, it finds word senses lexicalized by C or its variations by looking up WordNet, and expands the word senses by hyponymy. The semantic space for C is constructed by an $n * m$ matrix that comprises the n most frequently occurred terms in the vicinity of the m word senses. Based on the semantic space for C, LSA is used to find the best word sense associated with a query string using the terms in the vicinity of C. However, only subclasses and superclasses are included in the vicinity of the concept, and the descriptions of the concept is not considered in the query string.

GLUE is an instance-based approach to match taxonomies using machine learning techniques. In [4], the authors pro-

Table 1: Linguistics characteristics and techniques used in representative approaches

	Local Name	Descriptions	Neighboring Information	Thesaurus Look-up	Specific Techniques
Cupid	√			√	
OLA	√			√	relatedness of WordNet entries
ASCO	√	√		only synonym	
HCONE-merge	concepts		subclasses, superclasses	√	semantic space, LSA
GLUE	classes, instances	√	superclasses, attributes		machine learning
SCM	classes, instances	√	subclasses, instances		ranking, oblique coordinating
S-Match	concepts			√	semantic relation

pose a notion of concept similarity in terms of the joint probability distribution of instances of the concerned concepts. Multiple learning strategies are employed for computing concept similarities. It implements two base learners (Name Learner and Content Learner) and a meta-learner which combines the two base learner linearly. The Name Learner and the Content Learner learn from full names of instances and textual contents of instances. The full name of an instance is the concatenation of concept names leading from the root of the taxonomy to that instance. The textual content of an instance is the collection of tokens derived from its name and other attribute values.

SCM is a Semantic Category Matching approach to ontology alignment. In [8], SCM is based on the Vector Space Model [21]. However, it is limited to match class hierarchies. In that approach, words extracted from instances and subclasses of a given class are put into the statistics of word occurrences of the class, and only high-ranking words (e.g., the first 30 words) related to each class are selected as keywords (features) by a measurement similar to Kullback-Leibler's formulation. In addition, an oblique coordinate system that reflects the correlationship between the keywords can be adopted to make the vector space semantically more nature. Moreover, structural consistency analysis is used to evaluate and adjust the mapping.

S-Match is a semantic matcher for concept hierarchies. The key point of S-Match [7] is to compute the semantic relations holding between the concepts assigned to pair of nodes from the two concept trees. Possible semantic relations are: equivalence, more general, less general, mismatch and overlapping. It builds an atomic concept for each meaningful word based on WordNet, and then constructs complex concepts for labels using logical connectives. The concept at a node is built as the intersection of the concepts of labels of all the nodes above the node being analyzed. String manipulation (e.g., prefix, postfix analysis, n-grams analysis, edit distance) is employed to guess the semantic relation implicitly encoded in similar words. Finally, it tries to find the strongest relations holding between concepts of nodes by checking the satisfaction of logic formula.

Based on the investigation, we get some observation as follows.

Firstly, string comparison (e.g., edit distance) for name matching is a basic approach in linguistic matching. Most of the similarity-based approaches except HCONE-merge and SCM make use of string similarity to find initial mapping or to be combined with other similarities (e.g., structural similarity). Even in S-Match, i.e., so-called "Semantic Matching", string similarity is calculated and then used to guess the semantic relation implicitly encoded among similar words.

Secondly, not all the approaches exploit descriptions in linguistic matching. An ontology usually contains comments or other annotations to convey the intended meaning of the declared concepts. But not all the approaches exploit descriptions in linguistic matching with three exceptions ASCO, GLUE and SCM. And in HCONE-merge, annotations of concepts are ignored in constructing query string of a concept.

Thirdly, neighboring information has not been well exploited in these approaches. The presented approaches except HCONE-merge, GLUE and SCM, do not utilize neighboring information in linguistic matching. Even in SCM, only limited portion of neighboring information is utilized, which means only subclasses and instances are included in the neighborhood of concepts.

Finally, looking up thesaurus is time-consuming. As reported by the experience of OLA in OAEI 2005 contest, the approaches using the string distance have better performances and are also much more efficient than the ones using WordNet-based computation. As reported by the experience of ASCO, the integration of WordNet in the calculation of description similarity may not be valuable and cost much computation time. Our own experimental results, as will be presented in Section 5.4, also show that WordNet-based computation faces the problem of efficiency and even accuracy in some cases.

Based on the above observation, we propose a new idea of virtual documents to represent the intended meaning of URIrefs declared in an OWL/RDF ontology, by making use of not only local descriptions but also neighboring information. The virtual documents together with document similarity computation (e.g., TF/IDF technique) will be demonstrated to be a cost-effective approach to linguistic matching.

3. CONSTRUCTION OF VIRTUAL DOCUMENTS

In this section, we construct virtual documents for URIrefs declared in an OWL/RDF ontology. RDF graph model is the foundation of Semantic Web ontologies, and an OWL ontology can also be mapped to an RDF graph [20]. Therefore,

we use the RDF graph model to define description formulations and neighboring operations for constructing virtual documents.

As stated in [11], an RDF graph is a set of triples. An RDF triple (or a statement) is conventionally written in the order (subject, predicate, object). A node in an RDF graph may be a URI with an optional local name (URI reference, or URIref), a literal, or a blank node (having no separate form of identification). Note that predicates (or properties) are always URIrefs and a literal can not be a subject.

In the field of IR, the content of a document may be represented as a collection of tokens: words, stems, phrases, or other units derived or inferred from the text of the document. These tokens are usually weighted to indicate their importance within the document which can then be viewed as a vector in a N-dimensional space. In this paper, a virtual document is represented as a collection of weighted tokens, and the weights are rational numbers. To simplify the expression, we use the term "collection of words" instead of "collection of weighted tokens". Some terminologies, functions and symbols used in this paper are listed as follows.

URIref is short for Universal Resource Identifier Reference. In this paper, the URIrefs exclude the built-ins provided by ontology languages such as RDF or OWL. It is unnecessary to construct virtual documents for these built-ins because these ones are not the target of ontology matching and their semantics are fixed by W3C specifications.

***sub(s), pre(s) and obj(s)*:** the subject, predicate and object of the RDF triple (statement) s, respectively.

Words are normalized by a preprocessing step, e.g., parsing local names and descriptions into tokens based on punctuations, cases, etc., eliminating non-content-bearing "stopwords" and stemming.

Collection of words: "collection" is similar to the concept of "set", but allows the components repeating. In this paper, "collection of words" means "collection of weighted tokens" with the weights being rational numbers.

+: the operator stands for merging two collections of words together.

The description formulations will be given immediately. Briefly, for a literal node, the description is a collection of words derived from the literal itself; for a URIref (refers to *Definition 1*), it is a collection of words extracted from the local name, rdfs:label(s), rdfs:comment(s) and other possible annotations; for a blank node (refers to *Definition 2*), it is a collection of words extracted from the information originated from the forward neighbors. A weighting scheme is incorporated in the formation of the description. More formal definitions are given below.

Definition 1. (Descriptions of URIrefs) Let e be a URIref, the description of e is a collection of words defined by (1):

$$\begin{aligned} Des(e) \;=\; & \alpha_1 * \text{collection of words in the local name of } e \\ & + \alpha_2 * \text{collection of words in the rdfs:label of } e \\ & + \alpha_3 * \text{collection of words in the rdfs:comment of } e \\ & + \alpha_4 * \text{collection of words in other annotations of } e \end{aligned} \tag{1}$$

where $\alpha_1, \alpha_2, \alpha_3$ and α_4 are fixed rational numbers in $[0, 1]$, which indicate the weights for each category.

Definition 2. (Descriptions of Blank Nodes) Let B be the set of all the blank nodes, b be a blank node ($b \in B$) and s be any statements related to b. The description of b, written by $Des(b)$, is defined to be the convergence solution of the following iteration equations:

$$Des_1(b) \;=\; \beta * \sum_{sub(s)=b} Des(pre(s)) + Des(obj(s)) \tag{2}$$

$$Des_{k+1}(b) \;=\; Des_k(b) + \beta * \sum_{\substack{sub(s)=b \\ obj(s)\in B}} Des_k(obj(s)) \qquad (k \geq 1) \tag{3}$$

where β is a fixed rational number in $[0, 1)$. According to our experiments, five iterations of the computation of (3) is usually enough to get the convergence. Two special situations should be noticed: (i) we don't consider the cycle descriptions on blank nodes, and (ii) a transformation rule is introduced to avoid heterogeneous in expressing a List structure: all members of a list are collected, and the rdfs:member property is used to express the relation between the list and each of its members, instead of using RDF collection vocabularies (rdf:first, rdf:rest and rdf:nil).

To include descriptions of neighbors in virtual documents, we use three neighboring operations to capture different kinds of neighbors. For a URIref denoted by e, we use $SN(e)$ to denote the nodes that occur in triples with e as the subject, and we called it the subject neighboring operation. The definitions of the predicate neighboring operation ($PN(e)$) and the object neighboring operation ($ON(e)$) are analogous. The formal definition is given below.

Definition 3. (Neighbors of URIrefs) Let B be the set of all blank nodes and e be a URIref. The subject neighbors, written as $SN(e)$, is defined by (4). The predicate neighbors, written as $PN(e)$, is defined by (5). The object neighbors, written as $ON(e)$, is defined by (6).

$$SN(e) \;=\; \bigcup_{sub(s)=e} \{pre(s), obj(s)\} \tag{4}$$

$$PN(e) \;=\; \bigcup_{\substack{pre(s)=e \\ sub(s)\notin B}} \{sub(s), obj(s)\} \tag{5}$$

$$ON(e) \;=\; \bigcup_{\substack{obj(s)=e \\ sub(s)\notin B}} \{sub(s), pre(s)\} \tag{6}$$

Note that we exclude the statements that have a blank node as a subject in the computation of neighbors because the intended usage of this kind of statements is for describing the concerned blank nodes.

Based on the description formulations and the neighboring operations, we define the virtual documents of URIrefs as follows.

Definition 4. (Virtual Documents of URIrefs) Let e be a URIref. The virtual document of e, written as $VD(e)$, is defined by (7):

$$\begin{aligned} VD(e) &= Des(e) \\ &\quad + \gamma_1 * \sum_{e' \in SN(e)} Des(e') \\ &\quad + \gamma_2 * \sum_{e' \in PN(e)} Des(e') \\ &\quad + \gamma_3 * \sum_{e' \in ON(e)} Des(e') \end{aligned} \tag{7}$$

where γ_1, γ_2 and γ_3 are fixed rational numbers in $[0, 1]$.

3.1 A Simple Example

As an example to demonstrate the construction of virtual documents, we extract a segment from an OAEI 2005 benchmark test. The RDF graph structure of the segment is depicted in Fig. 1. There are three URIrefs (eg1:Article, eg1:Journal and eg1:articles) and a blank node (_:genid). After the preprocessing step, three words remain (magazine, article, journal). According to *Definition 1* and *Definition 2*, the descriptions of three URIrefs and a blank node are represented as follows.

$$\begin{aligned} Des(\text{eg1:Journal}) &= \{\alpha_1 * \text{“journal”}\} \\ Des(\text{eg1:articles}) &= \{\alpha_1 * \text{“article”}\} \\ Des(\text{eg1:Article}) &= \{\alpha_3 * \text{“magazine”}, \ \alpha_3 * \text{“journal”}, \\ &\qquad (\alpha_1 + \alpha_3) * \text{“article”}\} \end{aligned}$$

$$\begin{aligned} Des(\text{\_:genid}) &= \beta * (Des(\text{eg1:articles}) + Des(\text{eg1:Article})) \\ &= \{\alpha_3\beta * \text{“magazine”}, \ \alpha_3\beta * \text{“journal”}, \\ &\qquad (2\alpha_1 + \alpha_3)\beta * \text{“article”}\} \end{aligned}$$

The subject neighbor of eg1:Journal is _:genid according to *Definition 3*, and the predicate neighbor is empty and the object neighbor is eg1:articles.

The virtual document of eg1:Journal can be represented as follows according to *Definition 4*.

$$\begin{aligned} VD(\text{eg1:Journal}) &= Des(\text{eg1:Journal}) \\ &\quad + \gamma_1 * Des(\text{\_:genid}) \\ &\quad + \gamma_3 * Des(\text{eg1:articles}) \\ \\ &= \{(\alpha_1 + \alpha_3\beta\gamma_1) * \text{“journal”}, \\ &\qquad (2\alpha_1\beta\gamma_1 + \alpha_1\gamma_3 + \alpha_3\beta\gamma_1) * \text{“article”}, \\ &\qquad \alpha_3\beta\gamma_1 * \text{“magazine”}\} \end{aligned}$$

The virtual documents of eg1:articles and eg1:Article can be computed in a similar way.

4. SIMILARITY BETWEEN VIRTUAL DOCUMENTS

The similarity between virtual documents is calculated in the Vector Space Model (VSM) [21], combined with the prevalent TF/IDF [24] technique.

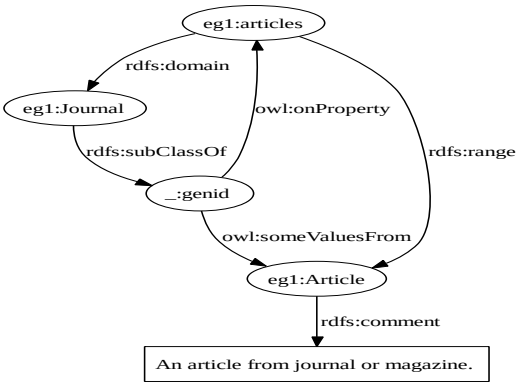

Figure 1: An example ontology

Each of the words occurring in either ontology forms a coordinate basis vector of the vector space; thus the dimension denotes the number of all the unique words. As mentioned in Section 3, each of the virtual documents of URIrefs consists of a collection of words according to the weighting scheme. Then each of the virtual documents can be represented as a vector in the vector space. The components of the vector are the scores assigned to corresponding words, which reflect the relatedness between words and virtual documents. The higher the score is, the more the word is related to the document. Therefore, in this Vector Space Model, the characteristics of virtual documents are described by weighted words.

In addition to the selection of words to represent virtual documents, it is common to associate a weight to each word in a document to reflect the importance of that word. Thus, TF/IDF technique is adopted in this approach to optimize the vector representation. Therefore, the final word score is calculated as follows.

$$WordScore = TF * IDF \tag{8}$$

$$TF = \frac{w}{W} \tag{9}$$

$$IDF = \frac{1}{2} * (1 + \log_2 \frac{N}{n}) \tag{10}$$

For each of the virtual documents, w denotes the refined word occurrence obtained from *Definition 4*; W denotes the total refined occurrence among all the words in a specific virtual document. For each of the words, n denotes the number of the virtual documents containing this word; N denotes the number of all the virtual documents. So the word score gives prominence to the words close related to the specific virtual documents, which somewhat exposes the latent features of the virtual documents.

Finally, the similarity between virtual documents is measured by the cosine value between the two vectors $\overrightarrow{N_i}$ and $\overrightarrow{N_j}$, representing the virtual documents in the Vector Space Model. The measure is as follows:

$$\cos(\overrightarrow{N_i}, \overrightarrow{N_j}) = \frac{\sum_{k=1}^{D} n_{ik} n_{jk}}{\sqrt{\sum_{k=1}^{D} n_{ik}^2 \sum_{k=1}^{D} n_{jk}^2}} \tag{11}$$

where D is the dimension of the vector space and $n_{ik}(n_{jk})$

is the components of the vectors. Thus, if the two virtual documents don't share any words, the similarity will be 0.0; if all the word scores equal completely, it will be 1.0.

5. EXPERIMENTAL RESULTS

In this section, we will present the experimental results that demonstrate the performance of the virtual documents approach on the OAEI 2005 benchmark tests. The tool accompanied this paper can be found at our website[4]. In fact, an early version of the virtual documents approach has been integrated with a graph-based approach named GMO [9] into an automatic ontology matching tool named Falcon-AO [10], which participated in the OAEI 2005 contest. According to the results validated by the organizers, our tool is largely dominant in benchmark tests and is one of the three best participants in directory tests.

5.1 Case Study

Generally, all the benchmark tests can be divided into five groups: Test 101-104, Test 201-210, Test 221-247, Test 248-266 and Test 301-304. A brief description is shown below.

Test 101-104 The two ontologies to be matched contain classes and properties with exactly the same or totally different names.

Test 201-210 The tests in this group are obtained by discarding some linguistic features of the initial ontologies leaving the structural features untouched, such as replacing local names of URIrefs with synonyms.

Test 221-247 The structures of the ontologies are changed, but the linguistic features remain. For example, properties and relations between objects have been completely suppressed.

Test 248-266 Some linguistic and structural features are both suppressed in this group. These test cases are the most difficult ones in all the benchmark tests.

Test 301-304 Four real-life ontologies of bibliographic references are found on the Web and left untouched.

As designed by the requirement, all the linguistic-based techniques should perform well in Test 101-104 and Test 221-247, but may fail to achieve good results on the other tests. The results of the experiments on these five groups are reported in correspondence.

5.2 Experimental Methodology

Three experiments are designed to evaluate the virtual documents approach, as compared to some other approaches.

Firstly, we validate the effectiveness of the neighboring operations as mentioned in *Definition 4*. According to (7), setting γ_1, γ_2, γ_3 to zero means we only consider the local descriptions; and setting them larger than zero means the neighboring information will be included in the virtual documents. The former is denoted by Simple V-Doc and the latter is denoted by V-Doc.

In our experiments, the parameters with V-Doc are configured as follows: (i) $\alpha_1, \alpha_2, \alpha_3$ and α_4 are set to 1.0, 0.5, 0.25 and 0, respectively; (ii) β is set to 0.5; and (iii) γ_1, γ_2 and γ_3 are set to 0.1, 0.1 and 0.1, respectively. In practice, α_i and β should be configured w.r.t. the cases. For example, in some ontologies, all the local names of the URIrefs are trivial serial numbers, then α_1 is recommended to be set to 0. Nevertheless, a slight change on γ_i does not effect remarkably.

[4]http://xobjects.seu.edu.cn/project/falcon/

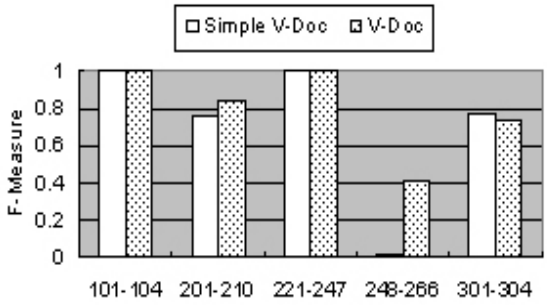

Figure 2: Simple V-Doc vs. V-Doc

Secondly, we design an exciting experiment to compare four different approaches: two string comparison approaches (one is based on edit distance, denoted by EditDist; the other is proposed by [25], denoted by I-Sub), V-Doc proposed in this paper, and a simple WordNet-based approach (denoted by WN-based).

Edit distance based string comparison approaches are one kind of the most commonly used approaches to gain the linguistic similarity in ontology matching. We implement a tool which is based on Levenshtein's edit distance [15], and measure the similarity of the two strings (here means local names) on a scale from 0 to 1 according to the following equation.

$$sim_{string}(c,d) = \max(0, \frac{\min(|c|,|d|) - ed(c,d)}{\min(|c|,|d|)}) \qquad (12)$$

I-Sub propose in [25] is a new metric for string comparison. Its novelity is that the similarity between two strings is related to their commonalities as well as to their differences. The experiments have shown that I-Sub performs better than other traditional string comparison techniques on average.

Some semantic lexicons (e.g., the best known WordNet) provide rich semantic relations, such as synonym and hyponym, to exploit the relatedness between words. As a comparison with our approach, we implement a simple WordNet-based algorithm according to the multidimensional scaling theory [2], which is widely used to calculate the similarity between sets.

The similarity between two URIrefs is also transformed to the similarity between their local names for simplification, which is calculated as follows:

$$sim_{set}(E,F) = \frac{\sum_{e \in E} \overrightarrow{e}}{|E|} \cdot \frac{\sum_{f \in F} \overrightarrow{f}}{|F|} \qquad (13)$$

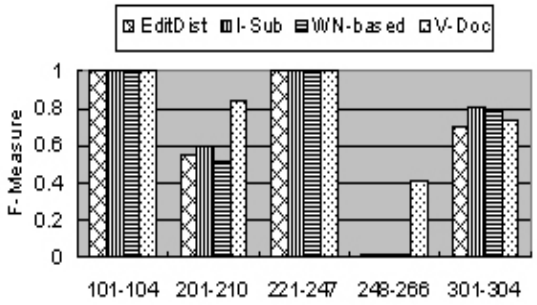

Figure 3: Comparison among four approaches

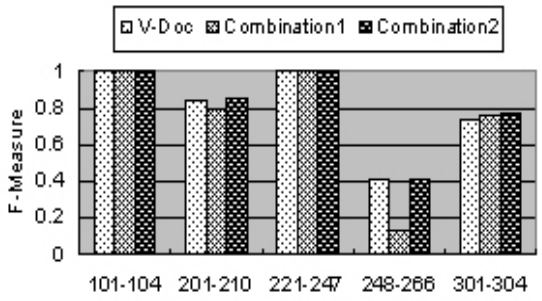

Figure 4: V-Doc vs. Combinations

$$\overrightarrow{e} = (sim(e,e_1), sim(e,e_2), \ldots, sim(e,f_1), sim(e,f_2), \ldots) \quad (14)$$

$$\overrightarrow{f} = (sim(f,e_1), sim(f,e_2), \ldots, sim(f,f_1), sim(f,f_2), \ldots) \quad (15)$$

where the two local names are regarded as two sets (E and F, respectively) of words (e_i and f_j, respectively); and $sim(x,y)$ is defined as follows.

$$sim(x,y) = \begin{cases} 1.0 & \text{if } x \text{ equals } y, \\ 0.8 & \text{if } x \text{ and } y \text{ co-occur in any synset,} \\ 0.0 & \text{otherwise.} \end{cases} \quad (16)$$

The comparison with SCM seems interesting. However, the developers of SCM haven't opened up the details of their tool but merely the experimental results on EON2004 tests (three groups of tests). So we can't compare V-Doc with SCM on the OAEI 2005 benchmark tests. Instead, we run V-Doc on EON2004 tests and find that V-Doc appears a distinct superiority especially on the group of real-life tests.

In the third experiment, we try to combine the string comparison approaches (EditDist and I-Sub) with V-Doc, and to see whether they could bring some improvement.

In all the experiments, we don't use any cutoff or threshold in our tests and it seems fair to all the approaches.

Please note that a complete comparison of these approaches is not an easy work due to the following three reasons: (a) matching is inherently a subjective operation and every test case also has its own characteristics (e.g., some test cases only have the subclass/superclass structure), (b) each approach to ontology matching is usually designed for a special goal, so some of its features are biased towards specific scenes, and (c) each approach usually has several different variations and it is impossible to cover all of them in our experiments. Still, we believe the experimental evaluation is essential to make progress in this difficult problem.

5.3 Measuring Matching Performance

We report the performance of each approach by using standard information retrieval metrics to assess the results of our tests. We focus on the average F-Measure of each approach w.r.t. each group of tests.

$$Precision = \frac{\sharp correct\_found\_matched\_pairs}{\sharp found\_matched\_paris} \quad (17)$$

$$Recall = \frac{\sharp correct\_found\_matched\_pairs}{\sharp existing\_matched\_paris} \quad (18)$$

$$F\text{-}Measure = \frac{2 * Precision * Recall}{Precision + Recall} \quad (19)$$

5.4 Discussion on Experimental Results

Firstly, the results of validating the effectiveness of the neighboring operations are depicted in Fig. 2. We can see that in most tests, when the neighboring information is utilized, the results are equal or better than that of only considering the local descriptions. In particular, in Test 248-266, there is little linguistic information in some local descriptions, so it is very effective to exploit the information from the neighbors. However, there is a slight fall in Test 301-304, since utilizing the neighboring information brings more found matched pairs. Therefore, an appropriate cutoff or threshold should be taken into account.

Secondly, the results of the comparison of the four different approaches are shown in Fig. 3. We can see that the performances of the four different approaches are very close in Test 101-104 and Test 221-247. In Test 301-304, I-Sub and WN-based seem a little better. But in Test 201-210 and 248-266, V-Doc is dominant. The average F-Measures of the four different approaches are depicted in Table 2 and we can see that V-Doc has more than 15% increasing in average F-Measure as compared to the other three approaches.

V-Doc is also cost-effective. In our environment (Intel Pentium 4 2.4 GHz processor and 512M memory, Windows XP), it takes 7 minutes to complete all the 51 test cases (including the time for parsing ontologies), while EditDist needs 48 seconds, I-Sub needs 50 seconds, but WN-based takes nearly 4 hours to complete all the tests. The average runtime per test of the four different approaches are also depicted in Table 2.

Finally, as shown in Fig. 4, we can see that there is little improvement in most tests when we combine V-Doc with EditDist or I-Sub (denoted by Combination1 and Combination2, respectively). The main reason is that some matched pairs found by V-Doc are rewritten by EditDist or I-Sub

Table 2: The summarization of average F-Measure and runtime per test

	101-104	201-210	221-247	248-266	301-304	Overall	Average Time
EditDist	**1.00**	0.55	**1.00**	0.01	0.70	0.60	**0.94(s)**
I-Sub	**1.00**	0.60	**1.00**	0.01	**0.81**	0.61	**1.0(s)**
WN-Based	**1.00**	0.51	**1.00**	0.01	0.78	0.59	282(s)
Simple V-Doc	**1.00**	0.76	**1.00**	0.01	0.77	0.64	4.3(s)
V-Doc	**1.00**	**0.84**	**1.00**	**0.41**	0.74	**0.77**	8.2(s)
Combination1	**1.00**	0.80	**1.00**	0.12	0.76	0.68	9.4(s)
Combination2	**1.00**	**0.85**	**1.00**	**0.41**	0.77	**0.78**	9.8(s)

but some of them are wrong. So we could make a empirical conclusion that simply combining these approaches cannot bring a much better result and just using V-Doc for linguistic matching is adequate in most cases.

The summarization of average F-Measure and runtime per test of all the experimental results is presented in Table 2.

6. RELATED WORK

As pointed out in Section 2, there are three related works in category of the similarity-based approaches to ontology matching. ASCO [14] makes use of local descriptions but not neighboring information, hence it can be considered as using the simple version of virtual documents in our perspective. Another case is HCONE-merge [12], where LSA is adopted. However, it is limited to match class hierarchies, and it does not utilize the descriptions of concepts. The third case is SCM [8], where specific techniques, such as keyword selecting and oblique coordinating, are adopted in the approach. It is also limited to matching class hierarchies, while it makes use of descriptions of instances and subclasses. Compared with them, the presented formulation of virtual documents is more comprehensive and well-founded: it bases on the RDF graph model, and it incorporates both descriptions and neighboring information with a weighting scheme. In practice, our implementation of V-Doc can be used to compare any two OWL/RDF ontologies.

Within the machine learning approaches to ontology matching, the training set is the classified documents or information contents. In [13], the classified documents are assumed to be existing beforehand. In GLUE [4], the information contents in the training set for Name Learner and Content Learner are full names and textual contents of instances, respectively. These information contents are generated at runtime and can be seen as virtual documents of instances in our perspective. Compared with the construction of information contents used in GLUE, we can see that our formulation of virtual documents is a more general way. In [27], the authors mainly focus on learning subclass relation between the concepts from two ontologies by using auxiliary knowledge sources (e.g., Google and domain-specific source) and information retrieval techniques. The methods discussed in [27] are targeted to mapping class hierarchies and at least one auxiliary knowledge source is required. In contrast, our approach is efficient and is not limited in these two aspects. However, we also believe that auxiliary knowledge source, e.g., domain-specific corpus, is worthy of being exploited for ontology matching.

7. CONCLUSION

In this paper, the main contributions are as follows:

Firstly, we have surveyed the linguistic characteristics and techniques currently used in ontology or schema matching. Based on the investigation, we observe that string comparison for matching names or local descriptions are adopted by many approaches, but neighboring information is not well exploited in ontology matching. Meanwhile, looking up thesaurus (even only synonym) is usually time-consuming.

Secondly, we have proposed a new idea of virtual documents to pursue a cost-effective approach to ontology matching. We present a way to construct virtual documents from an OWL/RDF ontology, including the description formulations and neighboring operations based on the RDF graph structure.

Finally, we have presented our experimental results of related approaches on the OAEI 2005 benchmark tests. The results demonstrate that linguistic matching based on the virtual documents has more than 15% increasing in average F-Measure and is also cost-effective as compared to other linguistic matching approaches.

As to the usage of the V-Doc, it can be integrated with other ontology matching approaches (e.g. I-Sub and GMO) in a matching system. Besides, just V-Doc can be used in some cases, especially when we want to slightly combine structural information but do not like to spend too much time. Because rigorous structural matching based on similarity propagation is time-consuming, the virtual documents approach gives another choice to the trade-off between efficiency and accuracy.

As future work, we will study intelligent neighboring operations and weighting scheme for neighbors. Currently, the neighboring operations involve only one-step neighbors without considering the importance of the involved relations (or properties). An intelligent neighboring operation should reach further neighbors along with the more important relationships. Of course, we need to find a suitable trade-off between the cost and the accuracy. Another future work is the incorporation of corpus-based distributional similarity among words. This kind of similarity can be incorporated in the computation of similarity between vectors of words. Finally, as we believe, the virtual documents can be extended to not only ontology matching but also other ontology engineering tasks, such as topic distilling and ontology partitioning.

Acknowledgments

The work is supported partially by the 973 Program of China under Grant 2003CB317004, and in part by the NSFC under Grant 60573083, and also in part by the JSNSF under Grant BK2003001. The first author of this paper is also supported by Ministry of Education of China under Grant NCET-04-0472. We would like to thank Ningsheng Jian, one of our team members, for his excellent work on related experiments. We are also grateful to Prof. Jianming Deng and Dr. Yanbing Wang for their valuable suggestions. In the end, we would like to thank anonymous reviewers for their helpful comments.

8. REFERENCES

[1] Castano, S., Antonellis, V.D., De Capitani di Vimercati, S. Global Viewing of Heterogeneous Data Sources. IEEE Trans. Knowl. Data Eng. 13(2) (2001), 277-297

[2] Cox, T., and Cox, M. Multidimensional Scaling. Chapman and Hall 1994

[3] Do, H.H., and Rahm, E. COMA - A System for Flexible Combination of Schema Matching Approaches. VLDB 2002, 610-621

[4] Doan, A., Madhavan, J., Dhamankar, R., Domingos, P., and Halevy, A.Y. Learning to Match Ontologies on the Semantic Web. VLDB J. 12(4) (2003), 303-319

[5] Ehrig, M., and Staab, S. QOM - Quick Ontology Mapping. International Semantic Web Conference 2004, 683-697

[6] Euzenat, J., Guegan, P., and Valtchev, P. OLA in the OAEI 2005 Alignment Contest. K-Cap 2005 Workshop on Integrating Ontologies 2005, 97-102

[7] Giunchiglia, F., Shvaiko, P., and Yatskevich, M. S-Match: An Algorithm and an Implementation of Semantic Matching. ESWS 2004, 61-75

[8] Hoshiai, T., Yamane, Y., Nakamura, D., and Tsuda, H. A Semantic Category Matching Approach to Ontology Alignment. Proceedings of the 3rd International Workshop on Evaluation of Ontology Based Tools (EON 2004). CEUR-WS Publication 2004

[9] Hu, W., Jian, N.S., Qu, Y.Z., and Wang, Y.B. GMO: A Graph Matching for Ontologies. K-Cap 2005 Workshop on Integrating Ontologies 2005, 43-50

[10] Jian, N.S., Hu, W., Cheng, G., and Qu, Y.Z. Falcon-AO: Aligning Ontologies with Falcon. K-Cap 2005 Workshop on Integrating Ontologies 2005, 87-93

[11] Klyne, G., and Carroll, J.J. (eds.). Resource Description Framework (RDF): Concepts and Abstract Syntax. W3C Recommendation 10 February 2004. Latest version is available at http://www.w3.org/TR/rdf-concepts/

[12] Kotis, K., Vouros, G.A., and Stergiou, K. Capturing Semantics Towards Automatic Coordination of Domain Ontologies. AIMSA 2004, 22-32

[13] Lacher, M.S., and Groh, G. Facilitating the Exchange of Explicit Knowledge through Ontology Mappings. FLAIRS Conference 2001, 305-309

[14] Le, B.T., Dieng-Kuntz, R., and Gandon, F. On Ontology Matching Problems - for Building a Corporate Semantic Web in a Multi-Communities Organization. ICEIS (4) 2004, 236-243

[15] Levenshtein, I.V. Binary Codes Capable of Correcting Deletions, Insertions, and Reversals. Soviet Physics - Doklady 10(8) (1966), 707-710

[16] Madhavan, J., Bernstein, P.A., and Rahm, E. Generic Schema Matching with Cupid. VLDB 2001, 49-58

[17] Melnik, S., Garcia-Molina, H., and Rahm, E. Similarity Flooding: A Versatile Graph Matching Algorithm and Its Application to Schema Matching. ICDE 2002, 117-128

[18] Noy, N.F. Semantic Integration: A Survey Of Ontology-Based Approaches. SIGMOD Record 33(4) 2004, 65-70

[19] Noy, N.F., and Musen, M.A. The PROMPT Suite: Interactive Tools for Ontology Merging and Mapping. International Journal of Human-Computer Studies 59 (2003), 983-1024

[20] Patel-Schneider, P.F., Hayes, P., and Horrocks, I. (eds.). OWL Web Ontology Language Semantics and Abstract Syntax. W3C Recommendation 10 February 2004. Latest version is available at http://www.w3.org/TR/owl-semantics/

[21] Raghavan, V.V. and Wong, S.K.M. A Critical Analysis of Vector Space Model for Information Retrieval. JASIS 37(5) 1986, 279-287

[22] Rahm, E., and Bernstein, P.A. A Survey of Approaches to Automatic Schema Matching. VLDB J. 10(4) (2001), 334-350

[23] Rodríguez, M.A., and Egenhofer, M.J. Determining Semantic Similarity among Entity Classes from Different Ontologies. IEEE Trans. Knowl. Data Eng. 15(2) (2003), 442-456

[24] Salton, G. and McGill, M.H. Introduction to Modern Information Retrieval. McGraw-Hill 1983

[25] Stoilos, G., Stamou, G., and Kollias, S. A String Metric for Ontology Alignment. International Semantic Web Conference 2005, 623-637

[26] Sure, Y., Corcho, O., Euzenat, J., and Hughes, T. (eds.). Proceedings of the 3rd International Workshop on Evaluation of Ontology Based Tools (EON 2004). CEUR-WS Publication 2004

[27] van Hage, W.R., Katrenko, S., and Schreiber, G. A Method to Combine Linguistic Ontology-Mapping Techniques. International Semantic Web Conference 2005, 732-744

[28] Watters, C. Information Retrieval and the Virtual Document. JASIS 50(11) (1999), 1028-1029