A Structure-based Similarity Spreading Approach for Ontology Matching

Ying Wang, Weiru Liu, and David A Bell

School of Electronics, Electrical Engineering and Computer Science,
Queen's University Belfast, Belfast, BT7 1NN, UK
{ywang14, w.liu, da.bell}@qub.ac.uk

Abstract. Most of the frequently used ontology mapping methods to date are based on linguistic information implied in ontologies. However, same concepts in different ontologies can represent different semantics under the context of different ontologies, so relationships on mapping cannot be solely recognized by applying linguistic information. Discovering and utilizing structural information in ontology is also very important. In this paper, we propose a structure-based similarity spreading method for ontology matching which consists of three steps. We first select centroid concepts from both ontologies using similarities between entities based on their linguistic information. Second, we partition each ontology based on the set of centroid concepts recognized in it using clustering method. Third, we utilize a similarity spreading method to update the similarities between entities from two ontologies and apply a greedy matching method to establish the final mapping results. The experimental results demonstrate that our approach is very effective and can obtain much better results comparing to other similarity based and similarity flooding based algorithms.

1 Introduction

Ontology mapping is a solution to the semantic heterogeneity problem in information integration and sharing. It establishes correspondences between semantically related entities in different ontologies [1]. Virtually any application that involves multiple ontologies must establish semantic mappings among them, to ensure interoperability [2]. Different applications arise in myriad domains [3]: ontology engineering, information integration, web services, and multi-agent communication etc.

Most of the ontology mapping approaches use the elementary-level matching techniques [4–6] (e.g., string-based methods, linguistic-based methods) which map elements by analyzing entities in isolation, ignoring their relationships with other entities [7]. But the determination of the true semantics of an entity is often difficult without a context, so structural information of an ontology plays an important role for ontology mapping.

When an ontology is viewed as a graph, an entity of an ontology (a node in a graph) inherits its parents semantics, and also passes on its own semantics to its children. Therefore, considering structural information is a natural way for enhancing ontology mapping as illustrated by the following two examples. Example 1 shows that two apparently different entities from two ontologies are similar when their neighboring concepts are similar. On the other hand, in Example 2, even when two entities of two ontologies are similar, if their neighboring concepts are not sufficiently similar, then these two entities are very likely not matched.

Example 1. Figure 1 is a mapping snippet between two ontologies which describes bibliographic references from OAEI benchmark. In this mapping example, concept **Proceedings** and concept **Proc** have different labels and seemingly not very similar semantically. But if they are embedded into the ontologies they belong, the neighbors of **Proceedings**: **Book, Monograph, Collection** are very similar to the neighbors of **Proc**: **Book, Monograph, Collection**, so **Proceedings** and **Proc** are highly possible to be mapped together.

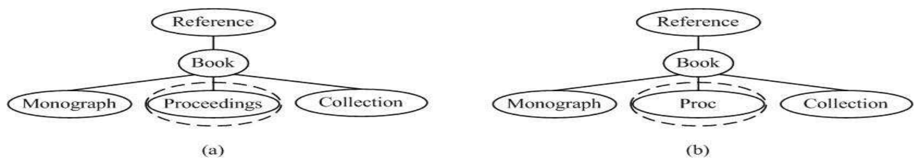

Fig. 1. An example of different concepts with the same meaning.

Example 2. Figure 2 shows an example of mapping snippet between two concept hierarchies which are extracted from Google Directory[1] and Yahoo Directory[2] separately. We observe that concept **History** in Figure 2(a) has exactly the same label as the concept **History** in Figure 2(b). When using traditional methods to compare the similarity between these two concepts, such as *edit-distance based or lexical-based methods*, a high similarity value will be obtained which means they should be mapped together. However this is incorrect, because we observe that the neighboring concepts of these two entities are not similar: one talks about the history of music and the other describes the history of architecture. So these two entities should not be mapped.

The idea of semantic propagation was explored in [6] on schema mapping in data integration, called *similarity flooding* which utilize graphs to compute structure similarities between data elements. In this paper, we extend this idea to ontology mapping. The difference between ontology mapping and schema mapping is that ontologies normally have richer semantics and more complex structures than schemas. If we apply the schema mapping similarity flooding algorithm to mapping ontolgies directly, the computational cost for storing graphs will be too high to be practically applicable.

[1] http://www.google.com/dirhp
[2] http://dir.yahoo.com/

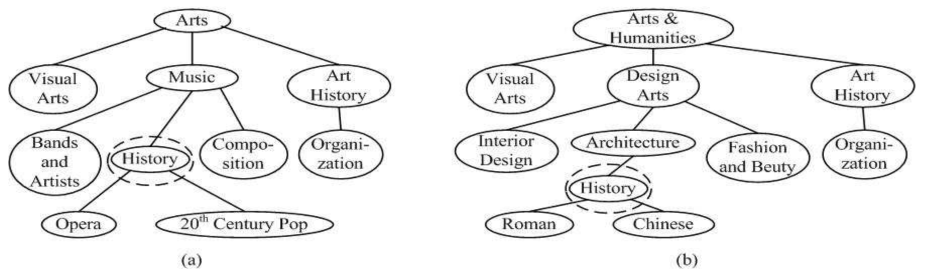

Fig. 2. An example of same concepts with different meaning.

Although there have been several extensions of similarity flooding to ontology mapping, such as [8, 9], they all suffer from some drawbacks. The main drawback in [8] is that the similarity propagation is done each time for a whole ontology graph. However, similarities can only be preserved to entities that are not too far from a given entity, that is, propagation should be done *locally* in relation to a given entity. The approach proposed in [9] tried to overcome this limitation by adding some restrictions when generating the pairwise connectivity graphs, so that propagations are done through these local graphs. However, the condition for generating the connectivity graphs may eliminate some useful links that should have been used for propagating similarities.

In this paper, we propose another extension to the similarity flooding approach, aiming to explore only locally related neighboring entities through a partition technique. This approach can reduce the computational complexity encountered in [8] on the one hand and avoid the danger of ignoring any useful entities on the other hand as may happen in [9], because only those neighboring concepts which are very close to the given entity structurally and highly related to this concept semantically should be considered. For example, as shown in Figure 2(a), concept **Organization** is not close to concept **History** and their semantics are not closely related, so the interaction (similarity propagation) between them can be ignored.

To propagate similarities locally, we need first to determine how to partition an ontology graph. To facilitate this, a set of centroid concepts (entities) are selected from two given ontologies. From these centroid concepts, some partition sets can be built. Our idea of selecting centroid concepts is built on the recognition that for two ontologies in the same domain, most of the concepts are repeated frequently, for instance, concepts **Reference, Book** of the first ontology also appear in the second ontology. Therefore, it is possible to start with highly matched concepts, selected as centroid concepts. This idea was used in [10] to find *anchors* (a pair of look-alike concepts) for flooding alignments to the neighbors of anchors. The difference between their approach and the idea here is that [10] floods alignments but our approach spreads similarities and revises the similarity value of a nodes when the similarity values of the nodes in its partition are changed.The idea of using *anchors* was also discussed in [11]. The difference between our approach of finding centroid and that in [11] is that the computation of structural similarities between entities are totally different.

Our mapping approach is realized by the following three main steps.

Step one: Establish preliminary mappings between any two entities from two given ontologies based on their descriptive (semantic) information.

Step two: Select centroid concepts from the two ontologies based on similarities between entities where the similarity value is 1. Each centroid is taken as the starting point of a partition set, and every remaining concept in each ontology is assigned to the partition set where the similarity value between the centroid of this partition set and this concept is the highest.

Step three: A structural-based similarity propagation method is performed to update the similarities between entities and a greedy matching method is used to find the final mapping results.

The main contributions of our proposed approach are:

- We propose a structure-based mapping method based on the similarity flooding algorithm.
- We select centroid concepts for determining the similarity propagation scope by deploying several computing similarity methods.
- We utilize a partition method to partition the entities in the same ontology into different similarity propagation scopes.
- Experimental results show that our method performs very well comparing with other similar approaches.

The rest of the paper is organized as follows. Section 2 introduces the basic methods used for computing the similarities between entities from two ontologies and then presents an algorithm to select the centroid concepts for each ontology. Section 3 illustrates our clustering-based method to partition ontologies. Section 4 presents the similarity spreading method. Section 5 describes the experimental datasets and the results of our experiments. Section 6 discusses related work and concludes the paper.

2 Centroid concepts selection

In this paper, ontologies are described by OWL and an entity in an ontology is defined as: $e \in C \cup P$ where C and P are the sets of concepts and properties in an ontology respectively.

We first compute the initial similarities between entities and then select the centroid concepts from these two ontologies.

When calculating similarities between entities, we aim to maximize the descriptive (or semantic) information of an entity, such as its ID, its label and its comment to cover diverse situations. The descriptive information of an entity is can be one of the following two forms:

- The descriptive information of a concept:
  $DI(e) = \{ID, label, comment\}$

– The descriptive information of a property:
  $DI(e) = \{ID, label, comment, domain, range\}$

where ID is the name of an entity, *label* provides an optional human readable name of an entity, and *comment* is often expressed in natural language describing an entity. *Domain* and *range* are the basic but important ways to restrict a property. The *domain* of a property limits the individuals to which the property can be applied and the *range* of a property limits the individuals that the property may have as its value[3].

Given two entities e_i from O_1 and e_j from O_2, we first apply the string-based and WordNet-based methods to compute the similarities between words, where these words are from the ID, label, or domain and range of entities. After this, we compute the similarities between entities based on the similarities of words. We also use the Vector Space Model (VSM)-based method to compute the similarities between entities based on their comments. Finally, we combine these similarities obtained from above methods as the final similarities between entities. The details of these methods are given below.

2.1 String-based method and WordNet-based method for computing similarities between entities

String-based method It consider strings as sequences of letters in an alphabet. They are typically based on the following intuition: the more similar the strings, the more likely they are to denote the same concepts [7]. In this paper, we apply the method proposed by Stoilos etc in [12] where the similarity measure between words w_i and w_j is defined as:

$$sim_{Str}(w_i, w_j) = comm(w_i, w_j) - diff(w_i, w_j) + winkler(w_i, w_j) \tag{1}$$

where $comm(w_i, w_j)$ stands for the commonality between w_i and w_j, $diff(w_i, w_j)$ for the difference between w_i and w_j, and $winkler(w_i, w_j)$ for the improvement of the result using the method introduced by Winkler in [13].

WordNet-based method It uses common knowledge or domain specific thesauri to match words. This kind of matcher has been used in many studies [14–16]. In this paper, we use an electronic lexicon, WordNet, for calculating the similarity values between words. WordNet is a lexical database developed by Princeton University which is now commonly viewed as an ontology for natural language concepts.

WordNet can be taken as a hierarchical structure and the idea of the path length method [17] is to find the sum of the shortest path from two concepts (words) to their common hypernym. The similarity between two words w_i and w_j is measured by using the inverse of the sum length of the shortest paths:

$$sim_{WN}(w_i, w_j) = \frac{1}{llength + rlength} \tag{2}$$

[3] http://www.w3.org/TR/owl-features/

where *llength* is the shortest path from word node w_i to its common hypernym with word node w_j and *rlength* denotes the shortest path from w_j to its common hypernym with w_i.

Calculating similarities from words to entities We have computed similarities between pairs of words according to two methods stated above, next we calculate similarities of entities based on the results obtained from the two methods separately. Let us assume that each entity is composed of several words and these individual words are grouped into a set. For example, entity *HistoryBook* is used to generate a set of words {*History, Book*}. Calculating the similarity between two entities is translated into calculating the similarity between two sets of words:

1. First, for each word w_i in one set of words, compute the similarity values between w_i and every word w_j from the other set of words and then pick out the largest similarity value.
2. Second, attach this value to word w_i. Repeat this step until all of the words in both sets have their own values.
3. Third, the final similarity value of a pair of entities is the sum of similarity values of all the words from the two sets divided by the total number of all the words in the two sets.

2.2 VSM-based method for computing similarities between entities

In vector space model [18], documents are represented by vectors of words and the similarity between two documents are computed by using the cosine similarity equation. To apply this method, we first regard each comment attached to an entity as a *document* and all of the comments belonging to two ontologies are regarded as the *collection of documents.* Then, we deploy the vector space model based method to compute the similarity between entities.

$$sim_{VSM}(e_i, e_j) = cossim(d_i, d_j) = \frac{\overrightarrow{d_i} \cdot \overrightarrow{d_j}}{|d_i| \cdot |d_j|} \tag{3}$$

where for each document d in a document collection D, a weighted vector can be constructed as $\overrightarrow{d} = (w_1, w_2, ..., w_n)$ where w_i is the weight of word i in document d, and $w_i = tf_i \cdot idf_i = tf_i \cdot \log \frac{|D|}{|d_i|}$ where tf_i is the frequency of word i in document d, $|D|$ is the total number of documents and $|d_i|$ is the number of documents that contains word i

2.3 Combination of similarities between entities

We have obtained the similarities of entities from three different methods by using descriptive information of entities, now we combine these similarities together. If the similarity value of one of three matchers based on descriptive information is 1, then the similarity value between these two entities is set to 1, since there is a method which is very sure about the equivalence between them. Otherwise, the similarity value is set to be the average the three similarities values obtained, as used in [19].

2.4 Centroid concepts selection

The selection of centroid concepts ($e \in C$) is based on the similarities between concepts from ontologies. As shown in Algorithm 1, entities (concepts) from two ontologies are selected as centroid concepts if each of them has a perfect match in another ontology.

Algorithm 1 Selecting Centroid Concepts from Ontologies

```
Input:  Ontologies $O_1$ and $O_2$
Output: Concept sets $C_1$ and $C_2$
1: $C_1 \leftarrow \emptyset, C_2 \leftarrow \emptyset$
2: for all $e_i \in O_1, e_j \in O_2$ do
3:     if $sim(e_i, e_j) = 1$ then
4:         $C_1 \leftarrow C_1 \cup \{e_i\}, C_2 \leftarrow C_2 \cup \{e_j\}$
5:     end if
6: end for
```

Example 3. After computing the similarities between concepts in Figure 2, we can obtain four centroid concepts for both ontologies: **Visual Arts**, **History**, **Art History**, **Organization**.

3 Ontology partition

Now, we describe the process of partitioning ontologies based on the centroid concepts selected from ontologies. The intuition of partitioning is that objects in the same partition set should be *close* or related to each other, both semantically and structurally.

3.1 Similarities between entities in one ontology

Structure similarity Wu and Palmer [20] proposed a method to measure the similarity between concepts within one conceptual domain. The conceptual domain is constructed as a hierarchical structure, so this method only considers the structure of the domain and does not use any other extra information.

$$sim_{Stru}(e_i, e_j) = \frac{2 * N_3}{N_1 + N_2 + 2 * N_3} \tag{4}$$

where e_i and e_j are two concepts in the same ontology and we assume that e_p is their most specific, common parent node; N_1 is the number of concepts on the path from e_i to e_p and N_2 is the number of concepts on the path from e_j to e_p; N_3 is the number of nodes on the path from e_p to root.

Semantic similarity The calculation of semantic similarity between entities in one ontology is the same as for calculating similarities between entities form two different ontologies as seen in Section 2.

3.2 Ontology partition

The algorithm for partitioning each ontology is illustrated below. We partition an ontology into a partition by considering the sum of the structural based similarity and semantic based similarity outlined in Section 3.1.

Algorithm 2 Partitioning Ontology

```
Input:  Ontology $O_1$ and centroid concept set $C_1$ where $C_1 = \{c_1, ...c_k\}$.
Output:  A partition $P = \{G_1, ...G_k\}$.
1: $G_i = \{c_i\}$ for $i = 1, ...., k$.
2: if $|C_1| > 1$ then
3:     for every entity $e_r$ in $O_1$ where $e_r \in C$ and $e_r \notin G_1 \cup ... \cup G_k$ do
4:         choose a centroid concepts $c_i \in C_1$ which has the maximum similarity with $e_r$
           (if there are several such centroid concepts, arbitrarily choose one).
5:         $G_i = G_i \cup \{e_r\}$.
6:     end for
7: end if
```

In this algorithm, if there is only one centroid concept, there is no need to do the partition, so the partition algorithm is applied when there is more than one centroid concept.

Example 4. Continue Example 3. We can partition the ontologies in Figure 2 as shown below:

– For ontology one demonstrated by Figure 2(a): *P*={{*Visual Arts, Arts, Music, Bands and Artists, Composition*},{*History, Opera, 20th Century Pop*}, {*Art History*}, {*Organization*}}
– For ontology two illustrated by Figure 2(b): *P*={{*Visual Arts, Arts& Humanities, Design Arts, Interior Design, Architecture, Fashion and Beauty*}, {*History, Roman, Chinese*}, {*Art History*},{*Organization*}}

4 Similarity propagation method

In order to apply the similarity flooding idea in our paper, we first introduce the similarity flooding (SF) algorithm proposed in [6]. In this method, two schemas *A* and *B* which are described by SQL DDL statements are first translated into graphs by using an import filter SQL2Graph that understands the definitions of relational schemas, in each of these graphs, different components in relational schema which have relationships are connected by edges. For example, there is an edge between table *Personnel* and column *Pno*. Each connection in a graph is represented as a triple (s, p, o) where s and o denote the source node and target node respectively, and p is the label of the edge. Then the three main processes in SF are:

– Construction of pairwise connectivity graph (PCG) between A and B. The PCG is defined as:

$$((x, y), p, (x', y')) \in PCG(A, B) \Leftrightarrow (x, p, x') \in A \text{ and } (y, p, y') \in B$$

Each node in the PCG is an element from $A \times B$, i.e. a possible candidate mapping pairs between two graphs.

– Construction of induced propagation graph (IPG). The induced propagation graph for A and B is constructed from PCG, where for every edge in the PCG, the IPG contains an additional edge with the same source and target nodes but in the opposite direction. The weights placed on the edges of the propagation graph indicate the coefficients of the similarity of a mapping candidate pair to its direct neighbors and back.

– Fixpoint computation. At the very beginning, the algorithm assigns similarities between nodes of two graphs using some traditional similarity calculation methods. Then it runs an propagation of similarities between connected nodes in the IPG iteratively until satisfy some stop conditions.

4.1 Construction of PCG and IPG from ontologies

Given two ontologies O_1 and O_2, we have partitioned them into partition sets. Here, we need to build a directed label graph for every partition set. In every partition set, the nodes are the concepts and the edges are from the structure information including *SubClassof*, *HasProperty*, *HasRange*, *SubPropertyof*. For each pair of corresponding partition sets from the two ontologies based on matched centroid concepts, we try to build an PCG. In this PCG, any pair of concepts nodes from O_1 and O_2 is merged into a single node when this pair of concepts have the same relationships with their neighbouring concepts, and in turn, their corresponding neighbouring concepts are merged too. For example, in Figure 3, {*history,history*} is merged into one node in Figure 4 because they have similar relationships with their child nodes. The IPG is then constructed based on this PCG as discussed above.

Example 5. Figure 3 shows two connected graphs, each is from one ontology from one partition set.

Figure 4 shows the PCG constructed from the two connected graphs in Figure 3, and its corresponding IPG.

4.2 Fixpoint computation

In SF, $\sigma(x, y)$ denotes the similarity between $x \in O_1$ and $y \in O_2$. The SF is based on the iterative computation of σ-values. In every iteration, the σ-value for a pair (x, y) is incremented by the σ-values of its neighbor pairs in the propagation

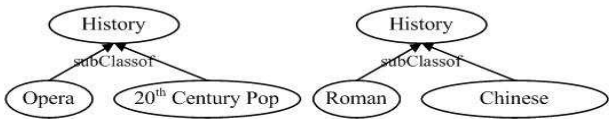

Fig. 3. An example of two connected graphs and each is from one partition set in one ontology.

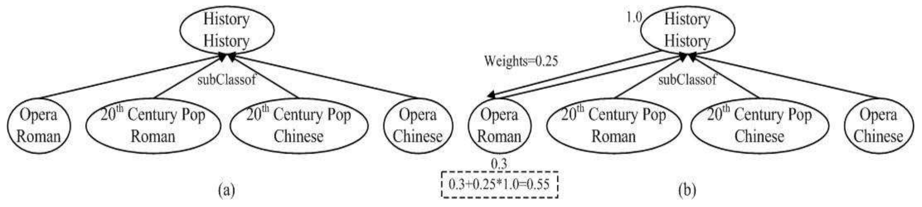

Fig. 4. An example of PCG and IPG.

graph multiplied by the weights on the edges going from the neighbor pairs to (x, y) [6]. The fixpoint formula of iteration for similarity propagation is:

$$\sigma^{i+1} = normalize(\sigma^i + \varphi^i) \tag{5}$$

$$\varphi^i = \Sigma_{j=1}^{m} \sigma_j^i w_j \tag{6}$$

where σ^i and σ^{i+1} are the similarities at the ith and $(i+1)th$ iteration, function φ^i is used to compute the increase of the similarities where m is the total number of neighbouring nodes connected. The value $(\sigma^i + \varphi^i)$ is finally normalized by the maximal σ-value of the current iteration ($(i+1)th$ iteration). The above computation is repeated until certain conditions are met, that is when there is no changes in similarities produced. To avoid a large number of iterations, a maximum number of propagation can be set to terminate the calculation.

4.3 Greedy matching

In order to finalize the matching, a greedy matching method in [21] is applied for choosing the best match candidates from the list of ranked matched pairs returned by the similarity spreading approach stated above. Once a pair has the maximum similarity, the entities in the pair are removed from the ontoliges, and the algorithm is applied to match the next pair until no more such pairs can be found.

5 Experiments

5.1 Datasets

We now present the experimental results that demonstrate the performance of different mapping methods on the OAEI 2009 Benchmark Tests. In our experiments, we only focus on classes and properties in ontologies.

Generally, almost all the benchmark tests in OAEI 2009 describe Bibliographic references except Test 102 which is about wine and they can be divided into five groups [22] in terms of their characteristics: Test 101-104, Test 201-210, Test 221-247, Test 248-266 and Test 301-304. A brief description is given below.

– **Test 101-104**: These tests contain classes and properties with either exactly the same or totally different names.
– **Test 201-210**: The tests in this group change some linguistic features compared to Test 101-104. For example, some of the ontologies in this group have no comments or names, names of some ontology have been replaced with synonyms.
– **Test 221-247**: The structures of the ontologies have been changed but the linguistic features have been maintained.
– **Test 248-266**: Both the structures and names of ontologies have been changed and the tests in this group are the most difficult cases in all the benchmark tests.
– **Test 301-304**: Four real-life ontologies about BibTeX.

5.2 Experimental evaluation metrics

To evaluate the performance of mapping, like many other papers that use retrieval metrics, *Precision*, *Recall* and *f-measure* to measure a mapping method, we use these measures to evaluate our methods as well. *Precision* describes the number of correctly identified mappings versus the number of all mappings discovered by the three approaches. *Recall* measures the number of correctly identified mappings versus the number of possible existing mappings discovered by hand. *f-measure* is defined as a combination of the *Precision* and *Recall*. Its score is in the range $[0, 1]$. $precision = \frac{|m_m \cap m_a|}{|m_a|}$, $recall = \frac{|m_m \cap m_a|}{|m_m|}$, $f - measure = \frac{2*precision*recall}{precision+recall}$ where m_m and m_a represent the mappings results discovered manually and by our method proposed in this paper respectively.

5.3 Comparison of experimental results

We now compare the outputs from our system (denoted as A-SP) to the results obtained from the similarity flooding algorithm (denoted as A-SF), and the traditional similarity based methods without using flooding technique (denoted as B-SP). The details are given in Figure 5, which compares the f-measure of three approaches. The benchmark tests are identified as numbers 101 to 304 on the x-axis.

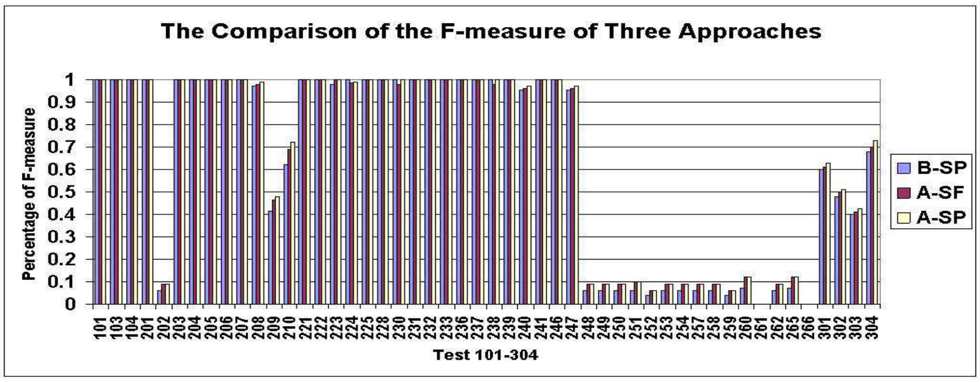

Fig. 5. The comparison of the f-measure of the three approaches.

We observe that the overall experimental results of $A - SF$ and $A - SP$ are better than $B - SP$, and the results of $A - SP$ are better than $A - SF$.

From Test 101 to Test 247, the two ontologies to be matched almost contain classes and properties with exactly the same names and comments, so every approach that deploys the computation of similarities of names and comments of entities can get good results. However, there still have three special cases do not obtain good results compared to other tests from Test 101 to Test 247. They are Test 205, 209 and 210. These three tests describe the same kind of information as other ontologies, i.e. publications, however, the class names and comments in them are very different from those in the reference ontology Test 101 so the three approaches does not obtain good results. We think $A - SP$ presents better than $A - SF$ on these tests because in the process of partitioning ontologies and constructing the PCG, $A - SP$ has got the best mapping pairs in each spreading scale. However, $A - SF$ has to construct PCG for the whole ontolgies, and some wrong similarities may generated during this process.

From Test 248 to Test 266, the names of the entities in ontologies are scrambled, meaningless and there are no comments attached to each entity, so string-based similarity method becomes the only useful method for computing similarity, but the similarity results still not very satisfied. These three approaches cannot obtain good results on this group of tests.

Test 301-304 are real-life BibTeX ontologies which also include different words compared to Test 101 describing publications so the results are similar to Test 205, so we do not get quite good similarity results from this data set.

6 Related work and Conclusion

Related work Many structure-based ontology mapping approaches have been proposed [11, 6, 8–10, 23–25]. Anchor-PROMPT [11] takes a set of anchors (pairs of related terms) as input from the source ontologies and traverses the paths between the anchors in the source ontologies. It compares the terms along these paths to identify similar terms and generates a set of new pairs of semantically similar terms. Similarity flooding [6] builds a pairwise connectivity graph and uses the structure features to spread similarity between elements in this graph. RiMOM [8] integrates multiple strategies for ontology alignment. It utilizes a strategy selection module to dynamically determine which strategies should be used in the alignment for different tasks. In its similarity propagation step, it uses the similarity flooding algorithm to generate structure similarities. The similarity propagation method [9] computes the similarity between entities based on the idea of similarity flooding. It first defines some conditions to limit the generation of pairwise connectivity graphs n the process of similarity propagation, it needs to build subgraphs for each element. Anchor-flood [10] starts off with an anchor (a pair of "look-alike") concepts form ontology and collects two blocks of neighboring concepts. The concepts of the pair of blocks are aligned and the process is repeated for finding new alignments. The differences between these approaches and our method proposed here have been discussed in the Introduction and our approach overcomes some weaknesses in these methods.

Conclusion In this paper, we propose a method for computing similarities based on both the semantic and structural information in ontologies. We particularly integrated the semantic flooding idea into our method in order to reflect structural information. Our methods consists of three steps to find the final mapping results. As a future work, investigation on how to partition ontologies will be carried out to see how different partitions will affect similarity spreading.

References

1. P. Shvaiko, J. Euzenat: Ten Challenges for Ontology Matching. In Proceedings of the 7th International Conference on Ontologies, DataBases, and Applications of Semantics (ODBASE'08). (2008) 300–313
2. A. Doan, J. Madhavan, P. Domingos, and A. Halevy: Ontology Matching: A Machine Learning Approach. Handbook on Ontologies in Information Systems, S. Staab and R. Studer (eds.), Springer-Velag, 2004. Invited paper, 397–416
3. J. Euzenat, P. Shvaiko: Ontology Matching. Springer, 2007.
4. H.H. Do, E. Rahm: COMA - A System for Flexible Combination of Schema Matching Approaches. In Proceedings of the 27th International Conference on Very Large Data Bases (VLDB'01). (2001) 610–621
5. J. Madhavan, P. Bernstein, E. Rahm: Generic Schema Matching with Cupid. In Proceedings of the 27th International Conference on Very Large Data Bases (VLDB'01). (2001) 49–58
6. S. Melnik, H. Garcia-Molina, and E. Rahm: Similarity Flooding: A Versatile Graph Matching Algorithm and Its Application to Schema Matching. In Proceedings of the 18th International Conference on Data Engineering (ICDE'02). (2002) 117–128
7. P. Shvaiko, J. Euzenat: A survey of schema-based matching approaches. Journal of Data Semantics. **4** (2005) 146–171
8. J. Li, J. Tang, Y. Li, Q. Luo: RiMOM: A Dynamic Multistrategy Ontology Alignment Framework. IEEE Transactions on Knowledge and Data Engineering. **21**(8) (2009) 1218–1232

9. P. Wang, B. Xu: An Effective Similarity Propagation Method for Matching Ontologies without Sufficient or Regular Linguistic Information. In Proceedings of the 4th the Semantic Web, Asian Conference (ASWC'09). (2009) 105–119
10. M.S. Hanif, M. Aono: An Efficient and Scalable Algorithm for Segmented Alignment of Ontologies of Arbitrary Size. Journal of Web Semantics. **7**(4) (2009) 344-356
11. N.F. Noy, M.A. Musen: Anchor-prompt: Using non-local context for semantic matching. In Workshop on Ontologies and Information Sharing at the 17th International Joint Conference on Articial Intelligence (IJCAI'01). (2001)
12. G. Stoilos, G.B. Stamou, S.D. Kollias: A string metric for ontology alignment. In Proceedings of the 4th International Semantic Web Conference (ISWC'05) (ISWC'05). (2005) 624-637
13. W. Winkler: The state record linkage and current research problems. Technical report, Statistics of Income Division, Internal Revenue Service Publication. (1999)
14. J. Tang, B. Liang, Z. Li: Multiple strategies detection in ontology mapping. In Proceedings of the 14th international conference on World Wide Web (WWW'05) (Special interest tracks and posters). (2005) 1040–1041
15. J. Madhavan, P.A. Bernstein, E. Rahm: Generic schema matching with cupid. In Proceedings of the 27th International Conference on Very Large Data Bases (VLDB'01). (2001) 49–58
16. P. Bouquet, L. Serafini, S. Zanobini: Semantic coordination: A new approach and an application. In Proceedings of the 2nd International Semantic Web Conference (ISWC'03). (2003) 130-145
17. P. Resnik: Using information content to evaluate semantic similarity in a taxonomy. In Proceedings of the 14th International Joint Conference for Artificial Intelligence (IJCAI'95). (1995) 448–453
18. G. Salton, A. Wong and C.S. Yang: A Vector Space Model for Automatic Indexing. Communications of the ACM. **18** (1975) 613–620
19. Q. Zhong, H. Li, J. Li, G. Xie, J. Tang, L. Zhou, Y. Pan: A Gauss Function based Approach for Unbalanced Ontology Matching. In Proceeding of the ACM SIGMOD International Conference on Management of Data (SIGMOD'09). (2009) 669–680
20. Z. Wu, M.S. Palmer: Verb Semantics and Lexical Selection. In Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics (ACL'94). (1994) 133–138
21. W. Wu, C.T. Yu, A. Doan, W. Meng: An Interactive Clustering-based Approach to Integrating Source Query interfaces on the Deep Web In Proceedings of the 30th ACM SIGMOD International Conference on Management of Data (SIGMOD'04). (2004) 95–106
22. Y. Qu, W. Hu, G. Cheng: Constructing virtual documents for ontology matching. In Proceedings of the 15th international conference on World Wide Web (WWW'06). (2006) 23–31
23. G. Jeh, J. Widom: SimRank: a measure of structural-context similarity. In Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'02). (2002) 538–543
24. W. Hu, N. Jian, Y. Qu, Y. Wang: GMO: A Graph Matching for Ontologies. In Proceedings of the K-CAP 2005 Workshop on Integrating Ontologies, (IO'05). (2005)
25. J. Euzenat, P. Guégan, P. Valtchev: OLA in the OAEI 2005 Alignment Contest. In Proceedings of the K-CAP 2005 Workshop on Integrating Ontologies, (IO'05). (2005)