An Effective Similarity Propagation Method for Matching Ontologies without Sufficient or Regular Linguistic Information

Peng Wang[1,2] and Baowen Xu[2,3]

[1] College of Software Engineering, Southeast University, China
[2] State Key Laboratory for Novel Software Technology, Nanjing University, China
[3] Department of Computer Science and Technology, Nanjing University, China
pwang@seu.edu.cn, bwxu@nju.edu.cn

Abstract. Most existing ontology matching methods are based on the linguistic information. However, some ontologies have not sufficient or regular linguistic information such as natural words and comments, so the linguistic-based methods can not work. Structure-based methods are more practical for this situation. Similarity propagation is a feasible idea to realize the structure-based matching. But traditional propagation does not take into consideration the ontology features and will be faced with effectiveness and performance problems. This paper analyzes the classical similarity propagation algorithm *Similarity Flood* and proposes a new structure-based ontology matching method. This method has two features: (1) It has more strict but reasonable propagation conditions which make matching process become more efficient and alignments become better. (2) A series of propagation strategies are used to improve the matching quality. Our method has been implemented in ontology matching system Lily. Experimental results demonstrate that this method performs well on the OAEI benchmark dataset.

1 Introduction

Currently more and more ontologies are used distributedly and built by different communities. Many ontologies describe similar domains but use different terminologies. Such ontologies are referred to as heterogeneous ontologies. It is the major obstacle to realize semantic interoperation. Ontology matching, which captures relations between ontologies, aims to provide a common layer from which heterogeneous ontologies could exchange information in semantically sound manners.

Some ontology matching methods have been proposed in recent years. In these methods, calculating linguistic similarity is the most popular way to discover alignments. However, not all ontologies provide sufficient and regular linguistic information. For example, the adult mouse anatomy ontology[1] uses codes like MA_0000436 to name the concepts. Some ontologies have few comments

[1] http://webrum.uni-mannheim.de/math/lski/anatomy09/mouse_anatomy_2008.owl

and labels to help the readers to understand their elements, and the 248-266 ontologies in the OAEI benchmark dataset are such extreme cases. For this situation, linguistic-based methods would miss a lot of alignments. Therefore, a practical matching system should consider the structure similarity to compensate for the disadvantages of linguistic-based methods.

The structure-based ontology matching is different from the geometrical graph matching because the latter can not reflect the semantic matching. So the traditional graph matching algorithms [1] are not suitable here. For ontology matching, the structure-based methods usually employ the similarity propagation idea "*similar objects are related to similar objects*". Several similarity propagation matching algorithms have been used for database or XML schema matching [2,3]. However, in our practice, we find these traditional similarity propagation matching algorithms can not be used for ontology matching directly. For example, we implement Blondel's graph matching algorithm [4] for ontology matching and can not obtain good results, but a modified method [5] performs well on the OAEI benchmark.

This paper analyzes the classical similarity propagation matching algorithm *Similarity Flood* [2], then proposes an effective similarity propagation method according to the ontology features. The new method can avoid some disadvantages of the previous one has, and can solve the ontology structure matching problem efficiently. The original contributions of this paper include: (1) We propose an effective similarity propagation method for matching ontologies, especially for the ontologies without sufficient and regular linguistic information; (2) The new method has more strict but reasonable propagation condition which makes matching process become more efficient and alignments become better; (3) A series of similarity propagation strategies are used in the method to improve the matching quality; (4) We implement the new method and the experimental results show the method is effective for ontology matching.

The remainder of this paper is organized as follows: Section 2 discusses the structure similarity problem in ontology matching. Section 3 presents the new similarity propagation method. Section 4 describes the propagation strategies. Some experimental results and discussions are presented in Section 5. Section 6 is a brief overview of related work and section 7 is the conclusion.

2 Structure Similarity Problem in Ontology Matching

Usually, an ontology contains concepts, relations, instances and axioms. The ontology matching can be defined as follows:

Definition 1 (*Ontology Matching*). *The matching between two ontologies* O_1 *and* O_2 *is a set of quadruples:* $\mathcal{M} = \{m_k | m_k = <se_i, te_j, r, s>\}$*, where* m_k *denotes an alignment,* se_i *and* te_j *represent the expressions which are composed of elements from* O_1 *and* O_2 *respectively;* r *is the semantic relation between* se_i *and* te_j*, and* r *could be equivalence*$(=)$*, generic/specific*$(\sqsupseteq/\sqsubseteq)$*, disjoint*$(\perp)$ *and overlap* $(\sqcap)$*, etc.;* s *is the confidence about an alignment and typically in the* $[0,1]$ *range.*

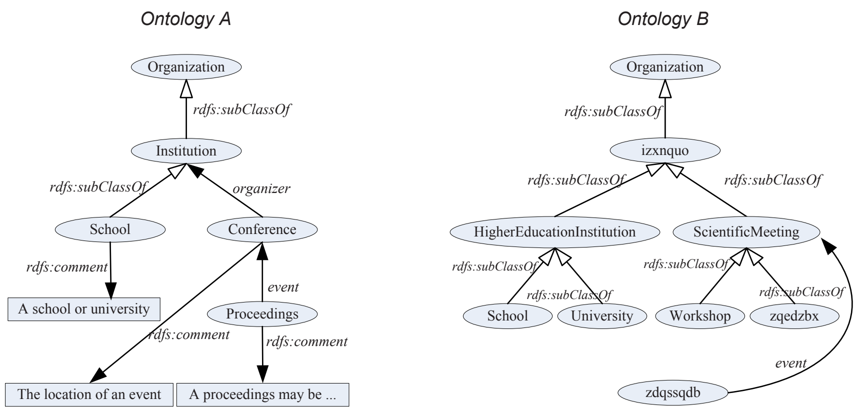

Fig. 1. An ontology matching example

This paper focuses on the matching about $concept - concept$ and $relation - relation$ with equivalence semantic relation.

Usually if we have enough and regular linguistic information about the elements, the alignments can be discovered easily. But the real world ontologies can not always provide sufficient and regular linguistic information. For example, Fig.1 is a matching snapshot in OAEI benchmark. ontology A has necessary comments for the concepts. The labels and comments in Ontology A are the regular natural words. But in ontology B some concepts have meaningless label and there is no any comments to explain these concepts. In our view, two reasons may cause this phenomenon. Firstly, some ontology engineers do not provide sufficient annotation for each element. Secondly, the ontology engineers would use some particular labels to name the elements. For instance, concept "Address" may be nameds as "Add" , "Adr" or "Dizhi" (in Chinese spelling). Therefore, it is necessary to find a way to discover the alignments for the ontologies without sufficient and regular linguistic information.

When the ontologies lack of linguistic information, matching methods usually utilize ontology structure information to find alignments. Although an ontology can be described as a graph, ontology matching is not equal to a graph matching problem. In graph matching, two elements are matched means they are similar in geometrical view other than they have semantic relation. Moreover, graph matching is a N–P problem [1], so it can not process ontology matching efficiently. For example, when we try to use the graph matching API provided by $SOQA - SimPack$ [6] to match ontology graphs, we find it needs more than several days or even several weeks for a normal matching task. Most importantly, only the graph topology can not represent the semantic information in ontology, so the geometrical graph similarity can not imply the corresponding elements are semantically similar. Therefore, it is not suitable to treat ontology matching as a simple graph matching problem.

Currently most structure-based ontology matching methods are inspired by the simple idea: "similar objects are related to similar objects". This idea also derives some heuristic rules, such as "concepts may be similar when their super/sub concepts are similar" and "concepts may be similar when they have similar instances". These rules have been used by some matching system [7,8]. But when the ontologies have little regular linguistic information, the heuristic rules usually can not work. It is because that the similarity between elements' neighbors can not be determined by linguistic information, the similarity between elements can not be determined too.

A feasible way is similarity propagation, namely, current similarity can propagate to neighbors in the graph to get more similarity results. After each propagation, the similarity results are normalized. The propagation process is terminated until the similarity results is convergent. Based on such similarity propagation idea, researchers have proposed some similarity propagation models [2,3,4,5,9]. Among these models, similarity flood is the most influential one. This paper will modify the similarity flood to solve the matching problem for the ontologies without sufficient or regular linguistic information.

The similarity flood includes three steps: (1) constructing pairwise connectivity graph; (2) constructing induced propagation graph; (3) computing fixpoint values for matching. Similarity flood is a versatile matching algorithm and can be implemented easily, but it is not sensitive for the initial similarity. Similarity flood algorithm has been used for schema matching in database and XML data. However, similarity flood is not a perfect algorithm. Melnik and his colleagues summarize six disadvantages [2], such as the neighbors have similarity is the necessary precondition of this algorithm. After we try to use similarity flood to match ontologies directly, we also find the algorithm can not work smoothly for ontology matching. First, similarity flood does not consider the similarity between edges, so the edge matching between ontologies can not be determined. Secondly, the maximum pairwise connectivity graph is $N_A * N_B$ (N_A and N_B are the numbers of edges in two ontologies), and it will greatly increase the time complexity for fixpoint computing and space complexity for storing the pairwise connectivity graphs. In real world matching tasks, the ontology graph may be thousands scale, so the corresponding pairwise connectivity graphs would become very large. For the above reasons, the similarity flood algorithm can not be used directly for ontology matching.

3 Similarity Propagation Method with Strong Constraint Condition

Ontology graph consists of the triples like $<s_i, p_i, o_i>$. The propagation condition is the core for a similarity propagation method. In ontology graph matching, a reasonable similarity propagation should consider both vertexes (s_i and o_i) and edges (p_i) in the triples. As far as similarity flood, the propagation condition presumes that all edge pairs (p_x, p_y) have 1.0 similarity value, and the similarity of one vertex pair (s_x, s_y) will be propagated to another vertex pair (o_x, o_y). This propagation condition obviously has the disadvantages: (1) It would produce a large number of alignment candidates and generate a large scale pairwise connectivity graph; (2) The propagation condition would produce many incorrect alignment candidates.

To provide a new similarity propagation method for dealing with ontology matching, this paper proposes a new propagation condition for ontology triples as definition 2, namely, the strong constraint condition for similarity propagation.

Definition 2 (*Strong Constraint Condition for Similarity Propagation in Triples*). *Given two triples $t_i =< s_i, p_i, o_i >$ and $t_j =< s_j, p_j, o_j >$, and let S_s,S_p and S_o denote the corresponding similarities of (s_i, s_j), (p_i, p_j) and (o_i, o_j) for the two triples. The similarity can be propagated iff t_i and t_j satisfy the following three conditions:*

(1) *In S_s, S_p and S_o, at least two similarities must be large than threshold θ;*
(2) *If t_i includes ontology language primitives, the corresponding positions of t_j must be the same primitives;*
(3) *t_i or t_j has at most one ontology language primitive.*

Condition (1) ensures the final similarity result is creditable after propagating. We set $\theta = 0.005$ in the implementation. The ontology language primitives refer to RDF vocabularies and OWL vocabularies. Condition (2) ensures two triples use same ontology language primitive to describe the facts. For example, $<Conference\_Paper, rdfs{:}subClassOf, Paper>$ and $<Paper, rdfs{:}subClassOf, Document>$ use the RDF primitive $rdfs{:}subClassOf$ as predicate, so the similarity can be propagated between them. Condition (3) ensures there is no definition and declaration triples during propagating, because such triples may cause incorrect matching results. For example, two triples $<PhDStu, rdf{:}type, rdfs : Class >$ and $< Paper, rdf : type, rdfs : Class >$ will cause wrong alignment: $PhDStu = Paper$.

After once propagation, the similarity of one element pair will be increased by the amount of other two pairs. Taking the similarity S_s as an example after i^{th} propagation, its new similarity is:

$$S_s^i = S_s^{i-1} + w_{po} \times S_p^{i-1} \times S_o^{i-1} \tag{1}$$

Analogously, the S_p^i and S_o^i are:

$$S_p^i = S_p^{i-1} + w_{so} \times S_s^{i-1} \times S_o^{i-1} \tag{2}$$

$$S_o^i = S_o^{i-1} + w_{sp} \times S_s^{i-1} \times S_p^{i-1} \tag{3}$$

The w_{po}, w_{so} and w_{sp} are propagation factors, and we will discuss them later.

All similarities will be normalized after each similarity propagation.

Based on the strong constraint condition, the new similarity propagation method still can be divided into three steps as follows:

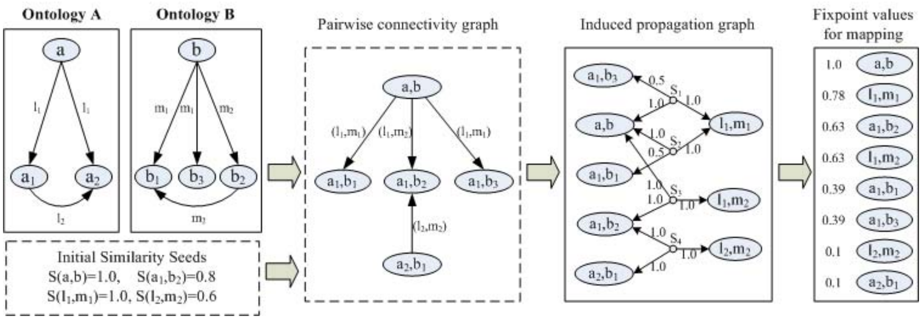

Fig. 2. Similarity propagation method with strong constraint condition

(1) *Constructing pairwise connectivity graph*
Traditional similarity flood is not sensitive to the initial similarity seeds, so all initial similarity values can be set to 1.0. However, in ontology matching, similarity propagation can not use the same setting, because it would not only cause very large pairwise connectivity graph but also generate many wrong alignments. In our view, the quality of the initial similarity seeds is very important for matching ontologies. We believe correct alignments would generate more correct alignments during similarity propagation, but wrong alignments would be noise in similarity propagation. Therefore, this paper try to use some high quality alignments as the initial similarity seed.

The seeds can be calculated by other linguistic-based matching methods or provided manually. This paper uses a method called *semantic description document* to produce the initial seeds. The detail about this linguistic-based matching method can be referred to our other work [10].

According to the initial similarity seed and the strong constraint condition, the pairwise connectivity graph can be constructed as Fig. 2 shows. Obviously, the pairwise connectivity graph is influenced by similarity seeds. Different seeds would cause different pairwise connectivity graphs.

(2) *Constructing propagation graph*
According to formula (1)-(3), the similarity from two element pairs always be propagated to the third pair. The propagation factor measures how many similarity can be propagated. There are three kinds of propagation factor: w_{sp}, w_{so} and w_{po}. Take w_{sp} as an example, it denotes how many similarity come from S_s and S_p can be propagated to S_o. Let f_{sp} denote the number of the triple pairs having $(s_i, s_j) \xrightarrow{(p_i,p_j)} (o_x, o_y)$ style in pairwise connectivity graph, then $w_{sp} = 1/f_{sp}$. w_{so} and w_{po} can be defined and calculated similarly.

Propagation graph can be represented by a bipartite graph as Fig. 2 shows. The weights of edges denotes the propagation factors. In the implementation, for the reason that the propagation factors can be directly obtained according to the pairwise connectivity graph, we can just record the propagation factors but need not to store the propagation graph.

(3) *Computing fixpoint*
The similarity propagation between ontology graphs can be computed iteratively until the final similarity matrix converges. Under the strong constraint condition, the fixpoint can be computed by formula (4), where normalization is omitted for clarity.

Actually, formula (4) is the synthesized style for formula (1)-(3). For each element pair (x, y), it would be subject pair, predicate pair or object pair, so its new similarity in the $(i+1)^{th}$ propagation would come from four parts: (1) the similarity in i^{th} propagation; (2) the propagating similarity when (x, y) is object pair; (3) the propagating similarity when (x, y) is subject pair; (4) the propagating similarity when (x, y) is predicate pair.

$$\begin{aligned} s^{i+1}(x,y) = s^i(x,y) &+ \sum_{\substack{<a_u,p_u,x>\in A \\ <b_u,q_u,y>\in B}} s^i(a_u,b_u)\cdot s^i(p_u,q_u)\cdot w_{sp} \\ &+ \sum_{\substack{<x,p_v,a_v>\in A \\ <y,q_v,b_v>\in B}} s^i(a_v,b_v)\cdot s^i(p_v,q_v)\cdot w_{po} \\ &+ \sum_{\substack{<a_t,x,c_t>\in A \\ <b_t,y,d_t>\in B}} s^i(a_t,b_t)\cdot s^i(c_t,d_t)\cdot w_{so} \end{aligned} \tag{4}$$

4 Propagation Strategies

To improve the efficiency of similarity propagation and the quality of propagation results, some reasonable strategies are adopted in the propagation. In general, this paper uses six strategies to make the matching results better and to improve the matching efficiency.

I. Propagation Scale Strategy
We should select the right parts in ontologies for similarity propagation. This paper studies four propagation scale strategies as follows:

(1) *Full graph propagation*
In similarity propagation, full ontology graph is the most direct propagation scale. It can assure that there is no ontology information to be missed. However, this propagation scale strategy also has obvious disadvantages: (a) For the large scale ontology graph, it is possible to cause large pairwise connectivity graph. (b) More triples do not mean better propagation results. Some triples are not important for describing the semantics. So too many triples may increase the uncertainty in propagation and bring negative affection for matching results.

(2) *Independent semantic subgraph propagation*
In an ontology, a semantic subgraph of an element is used to describe the element's meaning precisely. The definition and extracting algorithm of the semantic subgraph can be found in our other work [10]. If we constrain the propagation scale in the semantic subgraphs, the propagation can avoid the triples without important semantics. Therefore, the similarity propagation result would be

determined by the semantic subgraphs. Given two elements a and b and the corresponding semantic subgraphs G_a^s and G_b^s, the similarity $S(a,b)$ is obtained by the similarity propagating between G_a^s and G_b^s. If two ontologies have n and m elements respectively, this strategy needs $n \times m$ times propagation.

(3) *Combined semantic subgraph propagation*
The semantic subgraphs are combined in this strategy. We implement two combining ways: (a) combine all semantic subgraphs; (b) combine all semantic subgraphs of concepts to a graph G_C^c, and then combine semantic subgraphs of relations to another graph G_R^c. When we match concepts, we just consider G_C^c. Similarly, the matching between relations just uses G_R^c.

(4) *Hybrid semantic subgraph propagation*
This strategy is a mix of strategy (2) and strategy (3). In the propagation, one side is a semantic subgraph of an element e, and another side is the combined graph G_C^c or G_R^c. After a propagation, we can get the similarity about e to all the elements in another ontology. Obviously, this strategy needs n times propagation.

II. Incremental Updating for Pairwise Connectivity Graph
The strong constraint condition greatly reduces the scale of pairwise connectivity graph. After once similarity propagation, the similarity matrix would change and new similarity values would appear. Therefore, we need to construct a new pairwise connectivity graph for the next propagation. It is a time consuming process.

To reduce the constructing cost, we adopt an incremental updating way. After a propagation, the new pairwise connectivity graph need not to be reconstructed, but it can be extended based on the previous one. Namely, we just update the parts in the pairwise connectivity graph whose similarities have been changed.

III. Trust the Credible Seeds
In the initial similarity seeds, we regards the alignments having high similarity value as right alignments. Therefore, we keep these alignments during the propagation. The first advantage of this strategy is assuring some correct alignment can not be changed. Another advantage is avoiding some unnecessary similarity propagation computing. If $S(a_i, b_j)$ is a credible seed, then all similarity propagation like $S(a_i, b_x)$ and $S(a_y, b_j)$ can be skipped. In short, credible seeds not only can reduce the propagation cost, but also decrease the negative affection in propagation.

IV. Cross Validation for Propagation Result
This strategy only works for hybrid semantic subgraph propagation scale strategy. Given an element a_i, we can get a set of similarities $\{S(a_i, b_x)\}(x = 1, ..., n)$. Given another element b_j in the opponent ontology, we also can get another similarity set $\{S(a_y, b_j)\}(y = 1, ..., m)$. Therefore, we will have two similarity matrices. The similarity value at $S(a_i, b_j)$ may be different. This paper calculates the average of two similarity matrices as the final propagation result. The two similarity matrices have the function of validating crossly, so it can improve the quality of propagation result.

V. Penalty for Propagation

For an ideal similarity matrix, correct alignments should have higher confidence values and incorrect alignments should have lower confidence values. The real world similarity matrix is far away from the perfect one. So it is necessary to penalize the propagation result. The penalty should make little influence for alignments having high confidence value and make the potential misalignments have lower confidence value.

We provide two penalty factors p_a and p_b as follows:

$$p_a = \frac{s(a_i, b_j)}{max(s_{max}(a_i, b_x), s_{max}(a_y, b_j))} \tag{5}$$

$$p_b = \frac{1}{1 + e^{-\alpha t}}, t = (\frac{N+1}{n_i+1}/log(N+1)), \alpha \geq 1 \tag{6}$$

N is the sum of columns and rows of similarity matrix; n_i is the number of the alignments whose confidence values are large than 0 in i^{th} column and j^{th} row. After being penalized, the new similarity value is:

$$S^{'}(a_i, b_j) = S(a_i, b_j) \cdot p_a \cdot p_b \tag{7}$$

p_a penalizes the alignments having low similarity values, and p_b penalizes the alignments whose column and row have too many alignments with $S(x, y) > 0$. We set $\alpha = 3$ in the implementation.

VI. Termination Condition

Our propagation should satisfy two termination conditions: (1) The cosine between two sequential similarity matrices is not bigger than the given threshold. Propagation should assure the final similarity matrix is convergent. Melnik and his colleagues have proved that fixpoint computing can be convergent if the pairwise connectivity graph is a strongly connected graph [2]. (2) There is no updating for the pairwise connectivity graph. Besides the two conditions, to avoid the matrix needs too many times propagation to converge, we also set the maximum propagation times as 8 in the implementation.

5 Experimental Evaluation

We have implemented the new similarity propagation method in ontology mapping system Lily. Lily is implemented by Java and C++. More information about Lily can be found at http://ontomappinglab.googlepages.com/lily.htm.

The dataset is OAEI benchmark[2]. It includes more than 50 matching tasks having non-sequential number from 101 to 304. According to the dataset feature, we divide the dataset into 5 groups: (1) 101-104: this group contains same, irrelevant, language generalized and restricted ontologies. (2) 201-210: the ontology structure is preserved, but the labels and identifiers are replaced

[2] http://oaei.ontologymatching.org

by random names, misspellings, synonyms and foreign names. The comments have been suppressed in some cases. (3) 221-247: This group can be divided into two subgroups: 221-231 and 232-247. The first subgroup contains 11 kinds of modifications, such as the hierarchy is flattened or expanded, and individuals, restrictions and data types are suppressed. In the second subgroup, the modifications are the combinations of the ones used in 221-231. (4) 248-266: This is the most difficult test set. All labels and identifiers are replaced by random names, and the comments are also suppressed. (5) 301-304: This group contains 4 real matching tasks.

This paper uses the classical criterion: precision, recall and F-measure to evaluate the matching results. Let Q is the real matching result and T is the reference result, then the precision, recall and F-measure are:

$$P = \frac{|Q \cap T|}{|Q|}, R = \frac{|Q \cap T|}{|T|}, F\text{–}measure = \frac{2PR}{P+R} \tag{8}$$

5.1 Evaluating the Propagation Scale Strategies

This experiment aims to compare different propagation scale strategies. The dataset is 248 task. The experimental result (F-measure) is showed in Table 1. *Size* is the semantic subgraph size. *Seed* denotes the initial similarity seeds obtained by the linguistic matching method. C1, C2, C3 and C4 represent the four kinds of propagation scale strategies. Notice that C3A denotes all semantic subgraphs are combined; C3B denotes concept semantic subgraphs and relation semantic subgraphs are combined independently. C4A and C4B are both hybrid semantic subgraph propagation scale strategy, but in C4A the ontology has been enriched. The last row of Table 1 provides the average values for the semantic subgraphs from size 5 to 35.

Comparing with the seeds, Table 1 shows the similarity propagation can improve the quality of matching results. For all propagation scale strategies, their matching qualities can rank as: $C4B > C4A > C3B > C3A > C1 > C2$. Through

Table 1. Comparison of different propagation scale strategies

Size	Seed	C1	C2	C3A	C3B	C4A	C4B
0	0.020	0.020	0.020	0.020	0.020	0.020	0.020
1	0.371	0.603	0.604	0.422	0.246	0.537	0.496
2	0.431	0.653	0.604	0.547	0.352	0.570	0.552
3	0.418	0.531	0.476	0.541	0.400	0.715	0.736
5	0.493	0.608	0.529	0.658	0.607	0.761	0.802
10	0.536	0.587	0.592	0.675	0.693	0.828	0.828
15	0.586	0.643	0.586	0.662	0.658	0.849	0.837
20	0.557	0.610	0.630	0.671	0.675	0.822	0.785
25	0.561	0.629	0.598	0.662	0.731	0.789	0.832
30	0.561	0.648	0.690	0.658	0.706	0.753	0.879
35	0.561	0.620	0.651	0.621	0.653	0.716	0.826
Avg.	0.531	0.621	0.611	0.658	0.675	0.788	0.827

analyzing the experimental data, we have the conclusions: (1) Full graph strategy does not produce good results as we expect. (2) Independent semantic subgraph strategy causes the worst results. The reason is that the misalignments would have high similarity values when the similarity matrix is normalized after the propagation. So it is difficult to determine the correct alignments. (3) The results of C3A and C3B are very close; (4) C4A and C4B produce the best results. Surprisingly, the C4B without ontology enrich preprocess performs well than C4A with original ontology. The fact implies that some ontology preprocess may cause negative affection for similarity propagation.

5.2 Initial Similarity Seeds

We need validate whether and how our similarity propagation method is sensitive to the initial similarity seeds. In this experiment, we modified the seeds manually to keep the seeds always have the feature: $Precision = Recall = F\!-\!measure$. The seed quality F-measure decreases from 1.0 to 0 with step 0.1. The dataset is 248 task too. For an experiment at $F\!-\!measure\!=\!x$, we execute the matching three times. In each time, we modifies the seed randomly. We treat the average of the three results as the final $F\!-\!measure$ value.

The experimental result is showed in Fig. 3, where line B denotes the seed quality; C1, C3A and C4B are F-measure lines for the corresponding propagation scale strategies. We can draw the conclusions: (1) The initial seed influences the matching result greatly. With the change of seed quality, the matching result quality changes monotonously. (2) After propagating, the result is usually better than the initial seed. (3) C4B scale strategy is influenced by the seed slightly, so C4B is the preferred propagation scale strategy in the implementation.

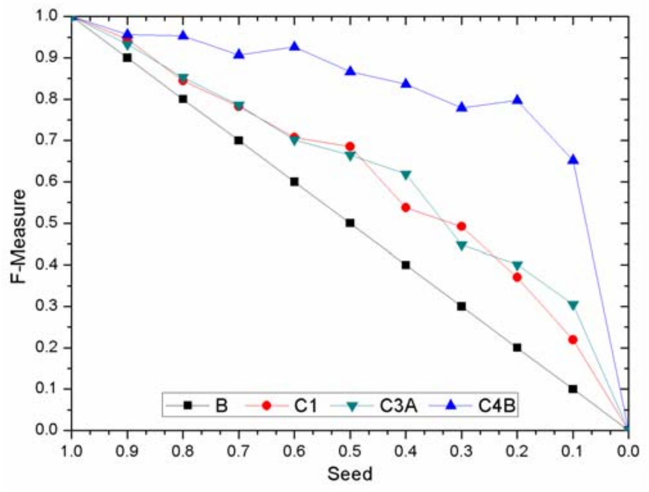

Fig. 3. Influence of initial seed to matching results

5.3 Overall Matching Results

Fig. 4 compares the matching results with similarity propagation and the results without similarity propagation. P1, R1 and F1 denote the precision, recall and F-measure for the method without similarity propagation. P2, R2 and F2 are the quality criterions about the method in this paper.

According to Fig. 4: (1) The similarity propagation method proposed in this paper improves the matching result quality, especially for the 248-266 dataset. (2) When the ontologies have not sufficient or regular linguistic information, we also find that our similarity propagation method can increase the recall greatly. Therefore, the similarity propagation method can discover more matching results using limited linguistic information.

Ontology mapping system Lily has implemented the similarity propagation methods in this paper, and Lily is one of the best systems in the OAEI benchmark evaluation in recent years. Table 2 shows results of some matching systems in OAEI-2008 benchmark evaluation [11, 12]. The evaluation divides the dataset

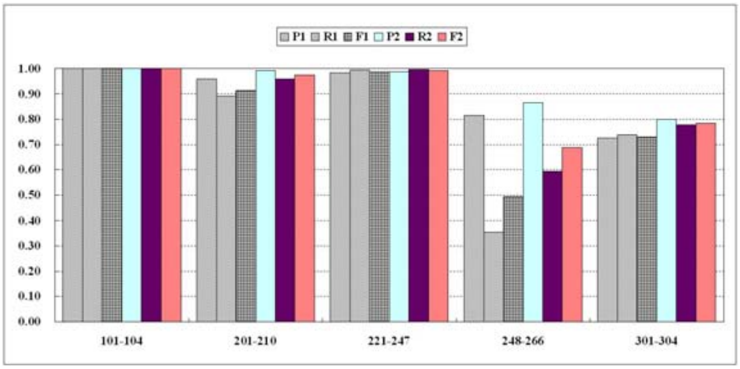

Fig. 4. Matching without similarity propagation VS with similarity propagation

Table 2. Overall matching results of some systems (OAEI 2008)

Systems	ASMOV		DSSim		Anchor-Flood		RiMOM		AROMA	
	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.
1xx	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
2xx	0.95	0.85	0.97	0.64	0.98	0.59	0.96	0.82	0.96	0.70
3xx	0.81	0.77	0.90	0.71	0.95	0.31	0.80	0.81	0.82	0.71
H-mean	0.95	0.86	0.97	0.67	0.98	0.62	0.96	0.84	0.95	0.70
Systems	CIDER		GeRoMe		SPIDER		SAMBO		Lily	
	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.
1xx	0.99	0.99	0.96	0.79	0.99	0.99	1.00	0.98	1.00	1.00
2xx	0.97	0.57	0.56	0.52	0.97	0.57	0.98	0.54	0.97	0.86
3xx	0.90	0.75	0.61	0.40	0.15	0.81	0.95	0.80	0.87	0.81
H-mean	0.97	0.62	0.60	0.58	0.81	0.63	0.99	0.58	0.97	0.88

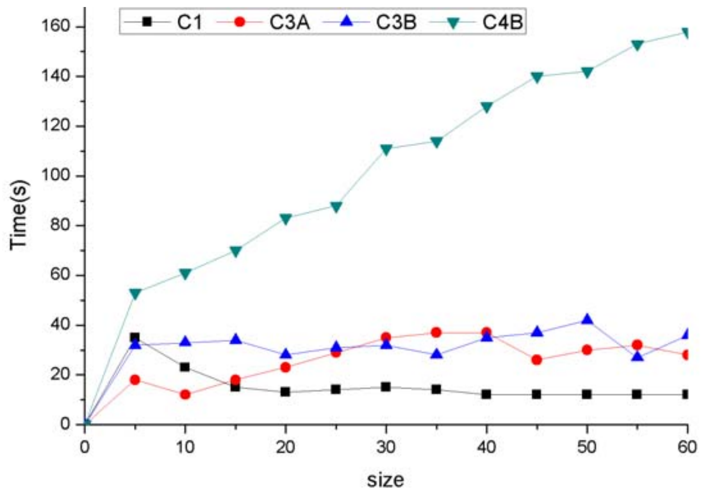

Fig. 5. Performance under different propagation scale strategies

into three groups: 1XX, 2XX and 3XX. The evaluation results show our system performs well on the benchmark dataset.

5.4 Performance

In Lily, we find the similarity propagation process occupies about 30%-50% matching time. It is necessary to analyze the major performance factors in propagation. In practical matching tasks, small semantic subgraphs would cause small pairwise connectivity graph, and the iteration process for calculating fixpoint would also terminate quickly. On the contrary, big semantic subgraphs would increase the burden in the propagation. So we believe the semantic subgraph size is a key factor for the performance. Fig. 5 demonstrates the running time with various subgraph size under different propagation scale strategies.

According to Fig. 5, (1) For full graph or combined subgraph scale strategies, the running time of propagation has no direct relevance to the semantic subgraph size. So C1, C3A and C3B are almost steady. When the semantic subgraph size is large than 5, the size of pairwise connectivity graph will keep stable. So the running time of C1, C3A and C3B would have little correlation with semantic subgraph size. (2) But for C4 propagation scale strategy, the running time increases with the semantic subgraph size quickly. It means we should set suitable semantic subgraph size for hybrid semantic subgraph scale strategy.

6 Related Work

Many ontology matching approaches are proposed in recent years. Some researchers have gave several excellent reviews for this topic [13, 14, 15]. This paper mainly focuses on the matching problem for the ontologies without sufficient or regular linguistic information. Structure-based matching method is a feasible way to deal with this special matching situation. For the reason that the ontology graph topology can not represent the semantics reasonably, the way using classical graph matching algorithm can not obtain good alignments, and it also have the serious performance problem due to high time complex of graph matching. Therefore, the method based on similarity propagation idea is the feasible way to solve the problem.

Blondel and his colleagues proposed a iteration equation for measuring the similarity between directed graphs [4]. This method is based on the $Hub-Authority$ idea. Based on the similar idea, in [9] the authors proposed a more universal measurement for the vertex similarity in network, and they also pointed out that Blondel's method is a special case of their method. These graph matching algorithms can only calculate the vertex similarity in graph, but they can not deal with the edge similarity. To overcome the problem, the bipartite graph is used to represent the ontology graph [5]. Tous and Delgado represent the ontology graph as the vector space method [16]. Similarity flood [2] is the most popular algorithm inspired by the similarity propagation, but it can not be directly used for ontology matching. In ontology mapping system RiMOM, three propagation strategies are used for structure matching: (1) propagation between concepts; (2) propagation between relations; (3) propagation between concepts and relations [17]. In another matching system PROMPT [8], a similarity propagation algorithm called AnchorPROMPT is used to find alignment.

The structure-based matching method based on our similarity propagation method is an effective solution for the ontologies without sufficient or regular linguistic information. Especially, our propagation method is reasonable for the ontology. We also propose some strategies to improve the propagation results and accelerate the matching process such as the propagation scale is constrained in the semantic subgraphs.

7 Conclusion

This paper proposes an effective similarity propagation method for matching the ontologies without sufficient or regular linguistic information. The new method is based on the strong constrained condition, and it is reasonable for ontology model. Some useful propagation strategies are also adopted to improve the matching results. Experiments shows that this structure-based matching method performs well on OAEI benchmark dataset.

References

1. Conte, D., Foggia, P., Sansone, C., Vento, M.: Thirty years of graph matching in pattern recognition. International Journal of Pattern Recognition and Articial Intelligence 18(3), 265–298 (2004)

2. Melnik, S., Garcia-Molina, H., Rahm, E.: Similarity flooding: A versatile graph matching algorithm and its application to schema matching. In: Proceeding of the 18th International Conference on Data Engineering (ICDE), San Jose, CA (2002)
3. Jeh, G., Widom, J.: Simrank: A measure of structural-context similarity. In: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Edmonton, Alberta, Canada (2002)
4. Blondel, V.D., Gajardo, A., Heymans, M., Senellart, P., Van Dooren, P.: A measure of similarity between graph vertices: Applications to synonym extraction and web searching. SIAM Review 46(4), 647–666 (2004)
5. Hu, W., Jian, N., Qu, Y., Wang, Y.: Gmo: A graph matching for ontologies. In: Integrating Ontologies Workshop, Banff, Alberta, Canada (2005)
6. Ziegler, P., Kiefer, C., Sturm, C., Dittrich, K.R., Bernstein, A.: Generic similarity detection in ontologies with the soqa-simpack toolkit. In: Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD 2006), Chicago, Illinois, USA (2006)
7. Ehrig, M., Staab, S.: QOM – quick ontology mapping. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) ISWC 2004. LNCS, vol. 3298, pp. 683–697. Springer, Heidelberg (2004)
8. Noy, N.F., Musen, M.A.: The prompt suite: Interactive tools for ontology merging and mapping. International Journal of Human-Computer Studies 59(6), 983–1024 (2003)
9. Leicht, E.A., Holme, P., Newman, M.E.J.: Vertex similarity in networks. Physical Review E 73 (2006)
10. Wang, P.: Research on the Key Issues in Ontology Mapping. PhD thesis, Southeast University, China (2009)
11. Caracciolo, C., Euzenat, J., Hollink, L., Ichise, R., et al.: Results of the ontology alignment evaluation initiative 2008. In: The Third International Workshop on Ontology Matching (OM 2008), Karlsruhe, Germany (2008)
12. Wang, P., Xu, B.: Lily: Ontology alignment results for oaei 2008. In: The Third International Workshop on Ontology Matching, OM 2008 (2008)
13. Kalfoglou, Y., Schorlemmer, M.: Ontology mapping: the state of the art. The Knowledge Engineering Review 18(1), 1–31 (2003)
14. Shvaiko, P., Euzenat, J.: A survey of schema-based matching approaches. In: Spaccapietra, S. (ed.) Journal on Data Semantics IV. LNCS, vol. 3730, pp. 146–171. Springer, Heidelberg (2005)
15. Rahm, E., Bernstein, P.A.: A survey of approaches to automatic schema matching. The VLDB Journal 10, 334–350 (2001)
16. Tous, R., Delgado, J.: A vector space model for semantic similarity calculation and OWL ontology alignment. In: Bressan, S., Küng, J., Wagner, R. (eds.) DEXA 2006. LNCS, vol. 4080, pp. 307–316. Springer, Heidelberg (2006)
17. Li, J., Tang, J., Li, Y., Luo, Q.: Rimom: A dynamic multistrategy ontology alignment framework. IEEE Transactions on Knowledge and Data Engineering 21(8), 1218–1232 (2009)