A Dichotomy Approach for Ontology Mapping

Zhi Huilai[1],Huo Zhanqiang[2],Xu Bin[3]

[1] School of Computer Science and Technology, Henan Polytechnic University
Jiaozuo, Henan, China

[2] School of Computer Science and Technology, Henan Polytechnic University
Jiaozuo, Henan, China

[3]Jiaozuo Municipal Center for Disease Control and Prevention
Jiaozuo, Henan, China

Abstract

Ontology mapping is the key challenge in the construction of semantic web, and a hinder in the handling of conflicts among heterogeneous data. We propose an efficient ontology mapping method which adopts a dichotomy approach. In this method, we divide original ontology into several small ones, and establish mapping between them, and at the end combine the local mapping results to get the final result. Compared with the method which doesn't use dichotomy approach, at the best case it will be reduced from n*n to n*logn. Experiments show that the accuracy of ontology mapping is also improved when the ontology has a graphical structure.

Keywords: *Ontology Mapping, Semantic Web, Dichotomy Approach, Concept Similarity.*

1. Introduction

In the last few years, a lot of effort has been put in the development of techniques that aim at the "Semantic Web". A lot of those newly developed techniques requires and enables the specification of ontologies [1] on the web. With the grown availability of large and specialized online ontologies, the questions about the combined use of independently developed ontologies have become even more important. Reuse of existing ontologies is often not possible without considerable effort [2].

When one wants to reuse different ontologies together, those ontologies have to be combined in some way. This can be done by integrating the ontologies, or the ontologies can be kept separately. In both cases, the ontologies have to be aligned, and we must know the semantic correspondences between their elements [3-4]. Ontology mapping, one that has received relatively little attention, becomes the key challenge in building semantic web. Although there is already a lot of research done in this area, there are still many open questions [5].

Currently, the major method to carry out ontology mapping is based on heuristic or matching learning technique. GLUE [6] uses probabilistic model, while Yin's method [7] is based on hidden Markov model. PROMPT [8] exploits lexical and structure information. FCA-merge [9] utilizes the idea that similar concepts classify documents will get similar result and concept lattice is used. And nearly all the above methods use hybrid matching strategy, especially Cupid [10], to enhance mapping precision.

Various kinds of methods have been used to deal with ontology mapping, every method has its merits, but no one can solve the problem perfectly. In this paper, a new method is proposed. In the conclusion of this paper, a simplified compare study for these methods is presented. In this paper, our method holds the semantic net view that meaning is inseparable from structure. Influenced by this standpoint, concept distribution and ontology structure are thoroughly surveyed which gives our method a solid theory base.

2. Ontology Mapping Process

There are three steps in the process of ontology mapping, firstly we divide ontology into sub-ontoloies, secondarily we establish mapping between sub-ontologies, finally we combine the mapping between sub-ontologies to form the final result.

In the first step of ontology mapping, there are still two issues to be considered. The first one is how the divide one ontology into smaller ones, which is constrained by the structure of the ontology, and the other one is how to compute concept similarity. The main frame of this step in shown in fig.1.

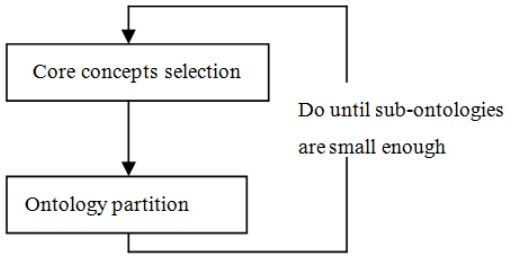

Fig.1 ontology partition

2.1 Core Concept Selection

Core concepts pick-up is the first step in ontology mapping. This step aims at find the most general concepts in ontology, from which others concepts evolve or derive from. In order to do this, the structure of ontology must be surveyed.

WordNet[11] can be viewed as two light weight ontology. Mark Steyvers and Joshua B. Tenenbaum's research show that although the processes to generate these two semantic networks surely differ in many important ways, the results are similar, WordNet has the distinctive statistical small-world and scale-free structures: very short average distances among nodes, high local clustering, and a perfectly power-law degree distribution [12]. It is the power-law degree distribution that gives a solid base of our mapping method.

The high-connectivity words at the tail of the power-law distribution can be thought of as the "hubs" of the semantic network [12]. In this paper, these "hubs" are called "core concept". In our method, "hubs" are mapping first, which are most likely to corresponding to each other. It is because that: core concept is usually confirmed by domain expert and represents the most general concepts, and is the premise for the other concepts to derive and evolve from.

Contrast to core concepts, we have "marginal concepts". If we use graph structure to present ontology, marginal concepts are the concepts that locate at the verge of the graph, which have the least number of neighbors and denote concrete concepts.

In this paper, ontology uses graph structure. The graph structure can be formalized as a couple G=(V,E), V is composed of nodes which denote concepts, namely, $v_1,v_2,\ldots,v_n$, $|V|=n$; E is composed of arcs which denote relations. Then G can be depicted by an $n\times n$ matrix $A=(a_{ij}),1\le i,j\le n$, $a_{ij}=1$,if $(i,j)\in E$; $a_{ij}=0$, if $(i,j)\notin E$, and this matrix is called adjacent matrix.

Now we can define core concept and marginal concept formally by the degree of a concept in a graph.

Defiition1: the concept which has the largest degree locally is a Core Concept; on the contrary, the concept which has the relatively smallest degree is a Marginal Concept.

The basic idea is to find the concept nodes which have largest degree, and then check if whether it links to a more general concept node. If not, it is a core concept; if yes, the more general concept node which it links by "is-a" relationship, is a core concept.

Core concepts selection algorithm is shown in Fig2. In this algorithm, we have to traverse the matrix to get the degree of every node, so the complex of this algorithm is o(n*n). After running algorithm 1, we will get several core concepts according to the scale or structure of the given ontology.

```
Algorithm 1: core concepts selection
Input: adjacent matrix
Output: one or two core concepts
Compute the degree of each node;
Set up an empty queue;
Push n/m nodes in-queue with the priority of their degree, node of larger degree in queue first;  /* According to the scale of ontology, we can adjust m's value */
Pop the head and do the following until queue is empty or already get two core concepts
{
For the head node #i
Decide_Core_Concept(node #i) /*decide whether node #i is a core concept*/
}

Method decide_Core_Concept(node #i)
{
If node #i doesn't has a adjacent node with relation is-a;
Then #i is a core concept;
Else Decide_Core_Concept(ADJ(#i));  /* recursive call to decide whether node #i is a core concept.  ADJ(#i) represents #i's adjacent nodes.*/
}
```

Fig2. Algorithm 1: core concepts selection

2.2 Ontology partition

This step is aims at reduce the scale of ontology, so as to reduce mapping complexity and improve mapping accuracy.

In the process of ontology process, concept similarity will be calculated. Concept similarity calculation plays a vital importance in pattern matching and neighbor searching [13]. In different application due to different the representation of the concept, the method of calculating concept similarity is different. Concepts in OWL have two features: Firstly, as son concepts inherit their father concepts' attributes, so all concepts form a hierarchical taxonomy tree; Secondly, concept's attributes are defined by asserted conditions including the ones inherit form their father concepts. The computing model also has two parts: First part is using recursive algorithm to compare concepts' attribute similarity; the second part is to calculate concept similarity by using the amount of same attributes. Fig. 3 depicts a concept similarity calculation algorithm.

```
Algorithm 2: calculate-similarity of two concepts
Input: two concepts
Output: similarity degree between 0 and 1, similarity of two concepts increases as the degree increases
Global m;/*define global variable, record the max assertion constraint number of c1 and c2 */
int Calculate-SameProperty(concept c1;concept c2)
{
 int sameProperty=0;/*record number of same assertion constraint */
p=c1; q=c2;
whlie (p.father!=SUP) /*SUP is root*/
{
   p.restriction union p.father.assertCondition;
   p piont at p.father;
}/*collect all the assertion constraint of c1 */
whlie (q.father!=SUP)
{
    q.restriction union .father.assertCondition;
    q piont at q.father;
}/* ollect all the assertion constraint of c2 */
m=max(|p.restriction|, |q.restriction|);/* record the max assertion constraint number of c1 and c2 */
For every property p in p.restriction
{ /*deal with every assertion constraint */
   if restriction is ( ∋ p d1)
    Then if there exist ( ∋ p d1) in q.restriction
    sameProperty=sameProperty+1;/*ad up same assertion constraint */
   if restriction is ( ≥ p x)
   then if there exist ( ≥ p y) in q.restriction and (y ≥ x)
      or there exist (=p x) in q.restriction
      or number of all filter in q.restriction more than x
      sameProperty=sameProperty+1;
   if restriction is ( ≤ p x)
   then if there exist ( ≤ p x) in q.restriction and (y ≤ x)
      or there exist (=p x) in q.restriction
      or number of all filter in q.restriction less than x
      sameProperty=sameProperty+1;
    if restriction is (=p x)
      then if there exit (=p x) in q.restriction
      or number of all filter in q.restriction equal x
      sameProperty=sameProperty+1;
   if restriction is ( ∀ p c) or ( ∃ p c)
    then if there exist ( ∀ p d )or ( ∃ p d)
     if c disjoint d
then return 0;
/*c disjoint d，then c1disjoint c2, return 0*/
          else then
 sameProperty=sameProperty+ Calculate-SameProperty(c,d)/m;
/*recursive compute the number of same assertion
```

Fig3. Algorithm 2: calculate-similarity of two concepts

Ontology partition method is based on the evaluation of concept similarity. If the similarity of two concepts is lower than the given threshold, these two concepts will be divided into two sub-ontologies. Fig. 4 depicts a ontology partition algorithm.

```
Algorithm 3: ontology partition
Input: ontology G=(V,E), two core concepts c and c', similarity threshold r
Output: two sub-ontologies
Select one core concept c randomly, do the following to generate sub-ontology Oc:
{
 Depth-first traverse (G, c); /*depth first traverse graph G from node c, c' is the currently visited concept */
 If calculate-similarity (c,c')> r then append concept c' to Oc ;
}
The rest concepts make up of sub-ontology Oc' ;
```

Fig4. Algorithm 3: ontology partition

2.3 Tools and strategies used in the process of ontology mapping

After we divide original ontology into sub-ontologies, we have to establish mapping between them. We need thesaurus or dictionaries to overcome naming difference.There are three kinds of naming difference. One is due to use of abbreviations, acronyms, punctuation, etc [14]. The other is due to language's exuberance. Still the other is the coding difference. The ways to solve these problems is illustrated thoroughly in reference [14]. These ways are not novel enough, so we don't discuss it in detail. In solving the above problems, thesaurus or dictionaries are used to identify similar concepts.

Besides thesaurus or dictionaries, we still adopt some strategies in ontology mapping, such as establish mapping between core concepts first, establish mapping between marginal concepts first, establish ontology mapping in a small scale, and establish mapping in double direction, one is from core concept to marginal concept; the other is from marginal concept to core concept.

2.4 Sub-ontology mapping result merging

Sub-ontologies mapping result merging is the final step of our method. In this step, the results of sub-ontologies mapping is merged to generate the final result. The most important thing is the merging order. This order must strictly obey the reverse order of ontology partition, which resembles the process of merge-Sort on arrays.

3. Efficiency Estimation

That we partition ontology into two part very time other than three or more parts is for the sake of convenience. Because if we can find a core concept and its neighbors with similarity larger than threshold, then these concepts as well as core concept will form a group which resembles a community, the rest will form another group naturally.

The reason why we partition ontology based on semantic similarity other than randomly partition is that: in ontology mapping, if we partition ontology randomly, similar concepts will not always lie in one sub-ontology, when establish mapping from one ontology to the other, it will make a lot of trouble, partition and conquer strategy will lose its meaning.

Dealing with semantic meaning is extremely a hard work. By dividing original ontology into two sub-ontologies, so the work reduces to half of the whole. By using divide and conquer strategy, we consider sub-ontology instead of the original ontology, so the complexity reduces n/logn times.

At the worst case, every time after ontology partition, only one concept could be separated from the others as shown in Fig5. After k times ontology partition, the original otology is divided into k+1 sub-ontologies, and there are one sub-ontology contains n-k concepts and the left sub-ontologies has only one concept and the complexity of ontology mapping is not reduced.

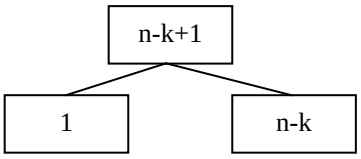

Fig5. The worst case of ontology partition

At the best case, every time after ontology partition, ontology will be divided into two sections which contain equal number of concepts. After k times ontology partition, the original otology is divided into 2^k sub-ontologies, and in every sub-ontology there are $n/2^k$ concepts. At this time, the complexity of ontology mapping is reduced to $1/2^k$ compare with the original problem.

4. Accuracy Estimation

4.1 Experiments on ontology which has taxonomy structure

We have evaluated our method on several real world domains. We test the effectiveness of our method on EON2004 benchmark test cases (see Table 1). These ontologies are about book or magazines, and have taxonomy structure, which resembles a tree.

Compared with other method [7-10], our method doesn't show significant improvement of precision, and shoe no advantages. To study carefully, ontology which is about the field of animal classification, administration, university, document classification and so on has tree structure. When using divide and conquer strategy, every branch is a sub-ontology naturally, and doesn't needs to be sorted, on the contrary cost extra time, so our method doesn't show its merits.

Table 1: EON2004 benchmark test
CASE(HTTP://CO4.INRIALPES.FR/ALIGN/CONTEST/)

Ontology	Concept	Attribute	Instance	Precision
Reference ontology	33	59	76	
101	33	61	111	100
103	33	61	111	97
104	33	61	111	100
201	34	62	111	91
202	34	62	111	68
204	33	61	111	93
205	34	61	111	82
221	34	61	111	98
222	29	61	111	97
223	68	61	111	98
225	33	61	111	98
230	25	54	83	87

4.2 Experiments on ontology which has graph structure

In this experiment, we establish ontology mapping in the domain of traffic accident. This kind of ontology is hard to find online, we establish them by ourselves.

First, we collect 100 reports about traffic accident online and randomly select 30 of them every time to establish ontology. We do 6 times to establish 6 ontologies for experiment. Ontology is semi-automatically established. First, extract high frequency used words; discard preposition, conjunction, auxiliary word, pronoun, pronoun, and adverb, only left noun and verb. Then use these noun and verb to establish ontology manually.

We chose one ontology as reference ontology. In this ontology, there are 29 concepts, namely "traffic accident", "loss", "death", "injured", "salvage", "ambulance", "overspend" , "overload" , "medical staff" , "Police" , "insurance company" and so on.

TABLE 2: ontology mapping in field of traffic accident

Ontology	Concept	Attribute	Use divide conquer strategy (precision)	Not use divide conquer strategy (precision)
Reference ontology	29	57		
1	32	59	89	75
2	30	57	90	78
3	32	58	83	70
4	28	55	87	72
5	28	55	91	79

There are five kind of relationship between concepts, namely "compose of", "is-a", "caused-by", "followed by" and "is-done-by".

Because of the complex relationship, ontology has a graph structure other than a tree. For There are "hubs" in it, for example "traffic accident" "loss" "salvage". Using these

"hubs" we divide ontology into smaller ones. Then, we establish mapping on sub-ontology instead of the original ones.

From the above table, we can see that using divide conquer strategy mapping accuracy is improved

Comparing our method with the other methods, it has no advantage on such certain taxonomically organized concepts, such as classes of animals or other natural kinds (most ontology mapping contest, such as Ontology Alignment Evaluation Initiative, is of this kind). But in the field, such as working flow, event describing, and cooperate network, our method can perform better than the methods. To contemplate on the reason, we find in these fields, "core concepts" exits explicitly and are easy to find, and the boundary of sub-ontology is clear. So the ontology is easy to partition and can get better result. To be more formally, these fields comply with power-law distribution perfectly and share the statistical small-world feature.

5. Conclusions

Establishing ontology mapping is the key technology in the reuse of existing ontology. Besides this, ontology mapping can be used in other potential applications, such as identical concept discovery, ontology database maintenance, people information retrieval and so on.

In this paper, we put forward divide and conquer, include core concepts pick-up and ontology partition. Besides, we also adopt several strategies to improve mapping accuracy, such as carry mapping in two directions (core to marginal and marginal to core), deal with core concepts and marginal concepts first. Experiments and analysis both show our method is reasonable and has its advantages in some important fields.

Time complexity: By using core concepts pick-up and ontology portion, the complex of our method is reduced form O(n*n) to O(n*logn). So, our method should ranks first. References [7-10] don't adopt this strategy; on the contrary, carry out mapping on the initial ontologies directly. Yin's Hidden Markov model [7] is most complex, for it doesn't have a decided mapping process, and solely depends on the learning process.

Versatility: Yin's Hidden Markov model [7] ranks the top, because it considers attributes, relations, hierarchies, sibling and rules all together, and adopts self-learning method to adjust the mapping result. GLUE [7] depends too much on instance of concepts. FCA-merge [9] depends on too much on the selected documents. The competence of methods [8,10] and our method is between [7] and [9].

Accuracy: Comparing our method with the other methods, it has no advantage on such certain taxonomically organized concepts, such as classes of animals or other natural kinds (most ontology mapping contest to test, such as Ontology Alignment Evaluation Initiative, is of this kind, this is also the reason we don't give comparison). But in the field, such as working flow, event describing, and cooperate network, our method can perform better than the methods. To contemplate on the reason, we find in these fields, "core concepts" exits explicitly and are easy to find, and the boundary of sub-ontology is clear. So the ontology is easy to partition and can get better result. To be more formally, these fields comply with power-law distribution perfectly and share the statistical small-world feature.

Acknowledgments

This paper is supported by National Science Foundation of China (60975033) and Innovative Foundation for Graduates of Shanghai University (A.16-0108-08-002, SHUCX091010) and Henan Polytechnic University Foundation for Doctor(B2011-102).

References

[1] Thomsa R. Gruber, A Translation Approach to Portable Ontology Specifications, Knowledge Acquisition, Vol.5, No. 2, 1993,pp.199-220.

[2] Simperl Elena , Sarasua Cristina, Ungrangsi, Rachanee, Bürger Tobias. Ontology Metadata for Ontology Reuse, International Journal of Metadata, Semantics and Ontologies, Vol.6, No.2,2011, pp 126-145.

[3] T. Berners-Lee, J. Hendler, O. Lassila, The Semantic Web, Scientific American, 2001.

[4] M. Uschold, Where are the Semantics in the Semantic Web? AI Magazine, Vol.24, No.3,2003, pp25-36.

[5] Natalya F. Noy, Michel Klein, Ontology Evolution: Not the Same as Schema Evolution, Knowledge and Information Systems, Vol.6, No.4,2004, pp.428-440.

[6] Doan, Pedro Domingos, Alon Halevy, Learning to Match Ontologies on the Semantic Web. Very Large Data Base Journal, Vol.12, No.4,2003, pp.303-319.

[7] Yin Kangyin, Song Zilin, Xu Ping, Ontology Mapping based on Hidden Markov Model. Journal of Southeast University (English Edition), Vol.23, No.3, 2007,pp.389-391.

[8] Natalya F N, Mark A M, The Prompt Suite: Interactive Tools for Ontology Merging and Mapping, International Journal of Human-Computer Studies, Vol.59, No.6, 2003,pp.983-1024.

[9] Gerd S, Alexander M. FCA-Merge: Bottom-up Merging of Ontologies, in Proceedings of the Seventeenth International Joint Conference on Artificial intelligence, 2001, pp.225-230.

[10] Yuzhong Qu, Wei Hu, Gong Cheng. Constructing Virtual Documents for Ontology Matching. in Proceedings of the 15th international conference on World Wide Web, 2006,pp.23-31.

[11] Fellbaum, C. WordNet, An Electronic Lexical Database USA:MIT Press, 1998.

[12] Mark. Steyvers, Joshua B. Tenenbaum. T, The Large-Scale Structure of Semantic Networks: Statistical Analyses and a Model of Semantic Growth, Cognitive Science, Vol.29, No.1, 2005,pp.41-78.

[13] Michihiro Jinnai, Satoru Tsuge, Shingo Kuroiwa, et.al. New Similarity Scale to Measure the Difference in Like Patterns with Noise, International Journal of Advanced Intelligence, Vol.1, No.1, 2009,pp.59-88.
[14] Yang Xiandi, He Ning, Wu Libing, et al. Ontology Based Approach of Semantic Information Integration, Journal of Southeast University (English Edition),Vol.23, No.3,2007, pp.338-342.

Zhi Hui-Lai received Dr. Eng. degree in 2010 from School of Computer Science and Technology Shanghai University, China. Currently, he is a lecture in Henan Polytechnic University, China. He has published 20 papers in the field of ontology engineering and formal concept analysis. His current research field is knowledge representation and extended model of concept lattice.

Huo Zhan-Qiang received the Dr. Eng. degree in circuits and system from Yanshan University (China) in 2010. He is a teacher at School of Computer Science and Technology, Henan Polytechnic University. His research interests are performance evaluation for wireless network system and vacation queuing systems.

Xu Bin received Master degree in Henan University (China) in 2012. He is a engineer in Jiaozuo Municipal Center for Disease Control and Presentation. His research interests are data analysis and disease prevention.