Journal of Southeast University (English Edition) Vol. 23, No. 3, pp. 338 – 342 Sept. 2007 ISSN 1003—7985

Ontology based approach of semantic information integration

Yang Xiandi[1,2] He Ning[1,2] Wu Libing[1,2] Liu Junqiang[3]

([1] School of Computer, Wuhan University, Wuhan 430072, China)
([2] Computer Center, Wuhan University, Wuhan 430072, China)
([3] State Key Laboratory of Software Engineering, Wuhan University, Wuhan 430072, China)

Abstract: In order to solve the problem of semantic heterogeneity in information integration, an ontology based semantic information integration (OSII) model and its logical framework are proposed. The OSII adopts the hybrid ontology approach and uses OWL (web ontology language) as the ontology language. It obtains unified views from multiple sources by building mappings between local ontologies and the global ontology. A tree-based multi-strategy ontology mapping algorithm is proposed. The algorithm is achieved by the following four steps: pre-processing, name mapping, subtree mapping and remedy mapping. The advantages of this algorithm are: mapping in the compatible datatype categories and using heuristic rules can improve mapping efficiency; both linguistic and structural similarity are used to improve the accuracy of the similarity calculation; an iterative remedy is adopted to obtain correct and complete mappings. A challenging example is used to illustrate the validity of the algorithm. The OSII is realized to effectively solve the problem of semantic heterogeneity in information integration and to implement interoperability of multiple information sources.

Key words: information integration; semantic heterogeneity; ontology; ontology mapping

With the development of information technologies, people demand complete access to available information, which is often heterogeneous and distributed. The technology of bringing together heterogeneous and distributed computer systems and establishing efficient information sharing is known as information integration. Therefore, the main aim of information integration is to improve system interoperability by resolving heterogeneity. Semantic heterogeneity refers to differences or similarities in the meaning of local data, which is currently causing great problems in information integration. Explicit definitions of terms used in schemas are a solution to this problem, so it led researchers to apply ontology as a promising solution to semantic heterogeneity.

Ontologies have been used in information integration systems in the following ways: ① Single ontology approach: a typical example of this approach is SIMS[1]. ② Multiple ontology approach: the OBSERVER system[2] is an example of this approach. ③ Hybrid ontology approach: a combination of the two preceding approaches is used. A local ontology is built for each source schema, which is not mapped to other local ontologies, but to a global shared ontology. New sources can be easily added with no need for modifying existing mappings. It avoids the disadvantages of single ontology or multiple ontology approaches.

Ontology mapping is one of the major bottlenecks in semantic integration. So how to build ontology mappings automatically or semi-automatically, thus reducing the work of the experts and speeding up this process is a hot-point for researchers. A number of ontology mapping methods exist in the research area of semantic information integration. GLUE[3] is an instance-based approach to match taxonomies using machine learning techniques. It uses multiple learners exploiting information in concept instances and a taxonomic structure of ontologies. But it needs a large number of examples for training, which is difficult to obtain. The PROMPT system[4] uses a mixture of lexical and structural features, as well as input from the user during an interactive merging session to find the mappings. Other newer methods such as QOM[5] (which is optimized for speed) and OLA[6], which combine a variety of different similarity measures. Especially, the Falcon[7] algorithm from Southeast University has done a good job.

This paper proposes an OSII (ontology based semantic information integration) model to solve the problem of semantic heterogeneity. First, we give the formal foundations of OSII, and then present a frame-

Received 2007-05-18.

Biography: Yang Xiandi (1974—), female, graduate, lecturer, yxiandi@126.com.

work for it. As shown in the framework, ontology plays a key role in OSII, thus a tree-based multi-strategy ontology mapping algorithm is described. A challenging example is used to illustrate the validity of the algorithm.

1 Foundations

As a foundation, we give some definitions for ontology, ontology mapping and alignment.

Definition 1 (ontology) Initially, ontology is introduced as an "explicit specification of a conceptualization"[8]. Our ontologies are built on OWL (web ontology language), so an ontology O can be expressed by a 5-tupel: $O = \{C, P, H_{C,P}, I, A\}$, where C is the set of concepts (examples of owl: Class); P is a set of data properties for concepts C (examples of owl: Datatype Property); $H_{C,P}$ is the corresponding hierarchy of C, P (examples of rdfs: subClassOf, rdfs: subPropertyOf, which denotes a directed relation called concept hierarchy or taxonomy); I is the instance set of concepts C; A is a set of axioms, expressed in a logical language, and it can be used to infer knowledge from an existing one.

Definition 2 (ontology mapping) Ontology mapping is the task of finding semantic relationships between entities (i. e. concept, attribute, and relation) of two ontologies. The similarity among ontologies is defined as a similarity function: $\mathrm{sim}(e_{1i}, e_{2j}) \in [0, 1]$ (0 indicates e_{1i} and e_{2j} are different, 1 shows they are the same), where e_{1i}, e_{2j} are two entities in ontology O_1 and O_2 separately. If the similarity $\mathrm{sim}(e_{1i}, e_{2j})$ exceeds a threshold $\mu \in [0, 1]$, we call e_{2j} the matching candidate of e_{1i}. Furthermore, if there are more than one matching candidates in O_2 for e_{1i}, the one with the highest similarity is selected as its matched entity.

Definition 3 (alignment) The result of mapping, called an alignment, is a set of pairs of entities $\langle e_{1i}, e_{2j} \rangle$ from two ontologies O_1 and O_2 that are supposed to be similar.

2 Framework

The framework of the OSII model is shown in Fig. 1, which is divided into three layers. They are the information resource layer, the ontology mediator and the user interface layer, respectively. The information resource layer gives a set of local information sources, which potentially store their data in different formats, e. g. , SQL, XML or RDF. The ontology mediator provides users with a uniform query interface via the global ontology to all the local information sources. This allows users to avoid querying the local information sources one by one, and to obtain a result from them just by querying the global ontology. The user interface layer should be friendly to the users as well as being highly efficient.

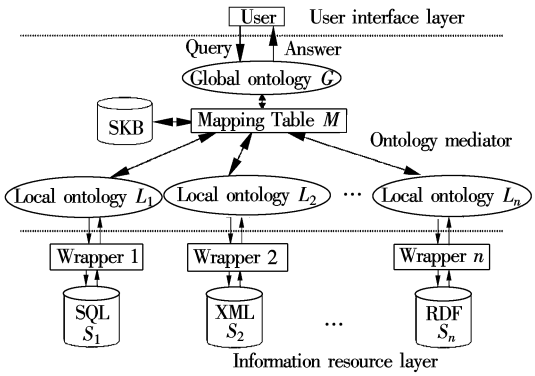

Fig. 1 Framework of the OSII model

We generate for each local source a local ontology, which represents the source schema. These local ontologies then map to the global ontology, which provides an overview of all the local ontologies and mediation between each pair of sources. A mapping table is produced to contain all the mappings, which are correspondences between the global ontology and local ontologies. A semantic knowledge base (SKB) is established to provide semantic mapping standards, which includes semantic dictionary (WordNet, HowNet), heuristic rules, etc. With the help of SKB we can achieve ontology mapping effectively and automatically.

A logical framework of OSII can be set up as follows. The ontology-based information integration framework I can be formalized as a 4-tuple $\langle G, S, f, M \rangle$, where G is the global ontology expressed in ontology language (such as RDFS, OWL) over the alphabet A_G. The alphabet comprises the names of the classes and properties of G. S is the source schema expressed in a language L_S over the alphabet A_S, which comprises the source (such as SQL, XML) element names in S. f is a schema transformation function, which generates a local ontology L for S. M is the mapping table consisting of a set of mappings between the global ontology G and a set of sources S_i, where $i \in [1 \ldots n]$. Each entry in M is of the form $(G, S_1, \ldots, S_n)$, where $G \in A_G$ and $S_i \in A_{Si} \cup \{\varepsilon\}$ for $i \in [1 \ldots n]$. Note that ε is used when a source schema has no corresponding elements to an element of G.

3 Design and Implementation

3.1 Ontology representation

In order to eliminate syntax heterogeneity, metadata (i. e., source schemas) in each data source should be explicitly represented by a local ontology, using a single language. The familiar ontology languages are XML schema, RDF(S), OWL, DAML + OIL and so on. Among all these ontology languages, we are most interested in OWL[9] for its particular roles in information integration and the semantic web. More specifically, OWL makes use of external data types (in particular it relies on XML schema datatypes). Additionally, a range of existing tools such as Protégé and Jena can be used to manipulate OWL-based ontologies.

3.2 Ontology mapping

Our approach begins with the belief that the combination of linguistic analysis and graph theory will lead to successful mapping. So a tree-based approach is provided. This paper only considers the 1∶1 mappings between concept-to-concept and property-to-property. The first is to discover concept mappings, and the latter is to discover property/relation mappings for the concept mapping. In order to improve mapping efficiency, OSII uses OWL's data type clustering similar elements into categories. We need only compare those that belong to compatible categories, so the number of element-to-element comparisons can be reduced. OSII also uses the following mapping rules:

Rule 1 If two entities have the same URI, they are identical, so the similarity is 1.

Rule 2 In OWL, there are properties such as equivalentClass, equivalentProperty or sameAs. They explicitly state that two entities are the same, so the similarity is 1.

Rule 3 In OWL, the properties such as differentFrom and AllDifferent explicitly state that two entities are different, so their similarity is 0.

Rule 4 If concepts/properties have similar parents and similar siblings (i. e. children of parents), and most of their children are similar, the compared concepts/properties are similar.

Rule 5 Concepts that have the same instances are the same.

Rule 6 If all properties of two concepts are similar, the concepts are also similar.

By rules 1 and 2, we obtain the exact mappings and can directly add them into the mapping table.

We give a running example to explain the algorithm. We want to match the two ontologies, PO and PurchaseOrder, in Fig. 2. The ontologies have parsed into node trees, where nodes without an oval represent concepts (classes) and nodes with an oval represent properties (attributes). Although the ontologies are very similar, there is much variation in naming and structure that makes algorithmic matching quite challenging. There are four steps as follows:

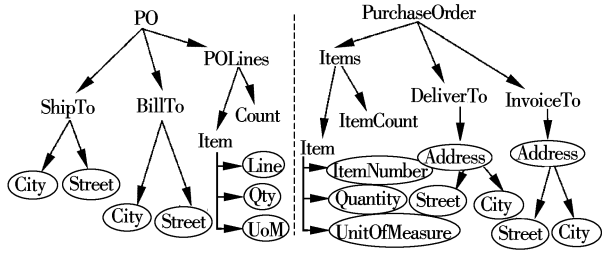

Fig. 2 Purchase order ontologies

Step 1 Pre-processing

Three processes are prepared for the mapping: ① Convert the class hierarchies in ontologies to node trees. We parse OWL ontology into a node tree (we call it a semantic tree) by Protégé. ② Add some helpful node features. For example, the features of a concept include: the number of its sub concepts, the number of properties it has, and the depth of the concept in the semantic tree. ③ Categorization by datatype. We make use of OWL's data types clustering similar elements into categories, e. g. string, numeral (include integer, float etc.), Boolean and date. We only compare those that belong to compatible categories. There are two categories in the example: 〈{PO, Count, Line, Qty}, {PurchaseOrder, ItemNumber, Quantity}〉 are numeral; others belong to string.

Step 2 Name mapping

We compare node names belonging to the compatible datatype categories and start with comparisons from the top down of the semantic tree. We choose the candidate by referring to its node features. For example, Item matches Item rather than Items, because their node features are similar. This paper adopts an edit distance based and a semantic dictionary based approach to achieve name mapping.

We first normalize node names into token sets. Similar schema elements in different schemas often have names that differ due to the use of abbreviations (Qty for Quantity), acronyms (UoM for UnitOfMeasure), punctuation, etc. So, we perform tokenization (parsing names into tokens based on punctuation), expansion (identifying abbreviations and acronyms) and

elimination (discarding prepositions, articles, etc.) using SKB's dictionary, e. g. "PO" results in {Purchase, Order}.

Based on Levenshtein's edit distance[10], the similarity of the two words on a scale from 0 to 1 is given according to the following equation.

$$\mathrm{sim}_{\mathrm{ed}}(e_1, e_2) = \max\left(0, \frac{\min(|e_1|, |e_2|) - \mathrm{ed}(e_1, e_2)}{\min(|e_1|, |e_2|)}\right)$$

The above method is simple and efficient in finding similarity. But sometimes it may be deceptive when two words resemble each other though there is no meaningful relationship between them, e. g. "power" and "tower". So, next we will refer to SKB's semantic dictionary to correct it and find synonymous terms (Bill and Invoice, Ship and Deliver):

$$\mathrm{sim}_{\mathrm{dic}}(e_1, e_2) = \begin{cases} 1 & \text{if } e_1 \text{ and } e_2 \text{ are synonymous} \\ 0 & \text{otherwise} \end{cases}$$

Then, we get the uniform similarity measure of the two words:

$$\mathrm{sim}(e_1, e_2) = \frac{\mathrm{sim}_{\mathrm{ed}}(e_1, e_2) + \mathrm{sim}_{\mathrm{dic}}(e_1, e_2)}{2}$$

Finally, the name mapping computes the similarity matrix for the two token sets. Each value in the matrix denotes the similarity of each pairwise-word. Formally, by inputting two entity names, name_1 and name_2, they are pre-processed into two token sets $\{w_i\}$ and $\{w_j\}$. Then for each w_i, we select the highest similarity sim (w_i, name_2) as the similarity between w_i and name_2. The similarity of name_1 and name_2 is defined as

$$\mathrm{sim}(\mathrm{name}_1, \mathrm{name}_2) = \sum_{i=1\ldots n} \frac{\mathrm{sim}(w_i, \mathrm{name}_2)}{n}$$

where n is the word count in name_1.

Step 3 SubTree mapping

Similarity between the descendent nodes can usually determine the similarity of a pair of nodes. Moreover, leaf nodes are the entities that express content. Therefore, the subtree mapping is actually leading the mapping process in the right direction. Note that in most cases not all of the descendent nodes can map precisely. So, we define a threshold for subtree mapping aiming at different applications. The threshold has the ability of learning from feedback. We use the method of machine learning and manual estimation to adjust the threshold and repeat this process until the threshold fluctuates in a steady narrow field.

We use linguistic similarity which is more reliable than structural similarity, so, when the linguistic similarity is very low, the alignment will be ignored avoiding measuring the structural similarity. If the linguistic similarity is not very low, the structural similarity will help to make the decision. We start comparisons from the bottom up of the semantic tree, so that we can reuse the lower nodes' similarities. The pair of nodes' structural mapping algorithm is as follows:

Input: Local node tree $^*L$, global node tree $^*G$;

Output: 1 or 0//1 denotes that the pair of nodes is matching, 0 denotes not.

```
int StructuralMapping(*L, *G)
{if (L != Null && G != Null)
    {if (NameMapping(L, G) > μ1)//μ1 is the threshold of name mapping
    {subtreeNum = max(number of subtrees of L, number of subtrees of G);
    match = 0;
    for each Lsubtree_i in L, each Gsubtree_i in G do
    match = match + StructuralMapping(Lsubtree_i, Gsubtree_i);
    if (match/subtreeNum) > μ2 return 1; else return 0; //μ2 is the threshold of structural mapping}
    else return 0; }
else if (L == Null && G == Null) return 1; else return 0; }
```

If the output is 1, the pair of nodes would be added into the mapping table. After step 3, the mapping results of the given example is shown in Tab. 1. We find the result mappings are not complete. Some mappings are missed due to the categorization or name mapping.

Tab. 1 Result of step 3

PO		PurchaseOrder	
ShipTo	City	DeliverTo	City
	Street		Street
BillTo	City	InvoiceTo	City
	Street		Street
Item	Qty	Item	Quantity
	UoM		UnitOfMeasure

Step 4 Remedy mapping

In order to obtain correct and complete mappings, we utilize structural information to remedy the mapping results. We use heuristic rules 3 to 6 to remove the unbelievable mappings, correct mistaken mappings, and create new mappings. We start the remedy from the top down of the semantic tree. In this step, we also support user interactive determination to make a better decision. The remedy mapping step is an iterative process. Iteration may stop when no new mappings are proposed. For example, POLines is mapped to Items because their parents match and all of their siblings match respectively (rule 4). In the same way, Count is mapped to ItemCount, Line is mapped to ItemNumber. Now we obtain correct and complete mappings.

Eventually, the output is a mapping table including multiple entries of $\mathrm{Map}(e_{\mathrm{L}i}, e_{\mathrm{G}i})$ from O_{L} to O_{G}.

Following it, semantic correspondences are established between local ontologies and the global ontology.

4 Conclusion

This paper provides a model of OSII as well as its logic framework, and presents its design and implementation. So it gives an efficient approach to resolving interoperability of heterogeneity sources. A tree-based ontology mapping method is proposed to facilitate the OSII. We use a challenging example to illustrate the validity of the algorithm.

References

[1] Wache H, Vögele T, Visser U, et al. Ontology-based integration of information—a survey of existing approaches [C]//*Proc of IJCAI Workshop*: *Ontologies and Information Sharing*. Seattle, 2001: 108 - 117.

[2] Mena Eduardo, Kashyap Vipul, Sheth Amit P, et al. OBSERVER: an approach for query processing in global information systems based on interoperation across pre-existing ontologies [C]//*Proc of the* 1*st IFCIS International Conference on Cooperative Information Systems*. Brussels, Belgium, 1996: 14 - 25.

[3] Doan A, Madhavan J, Dhamankar R, et al. Learning to match ontologies on the semantic web [J]. *Very Large Data Bases Journal*, 2003, **12**(4): 303 - 319.

[4] Noy N F, Musen M A. The PROMPT suite: interactive tools for ontology merging and mapping [J]. *International Journal of Human-Computer Studies*, 2003, **59**(6): 983 - 1024.

[5] Marc Ehrig, Steffen Staab. QOM—quick ontology mapping [C]//*Proc of the* 3*rd International Semantic Web Conference*. Hiroshima, Japan, 2004: 683 - 697.

[6] Euzenat J, Loup David, Touzani Mohamed, et al. Ontology alignment with OLA [C]//*Proc of the* 3*rd International Workshop on Evaluation of Ontology based Tools* (*EON*). Hiroshima, Japan, 2004: 59 - 68.

[7] Jian Ningsheng, Hu Wei, Cheng Gong, et al. Falcon-AO: aligning ontologies with falcon [C]//*K-CAP* 2005 *Integrating Ontologies Workshop*. Banff, Alberta, Canada, 2005: 85 - 91.

[8] Gruber Tom. A translation approach to portable ontology specifications [J]. *Knowledge Acquisition*, 1993, **5**(2): 199 - 220.

[9] W3C. OWL web ontology language guide [EB/OL]. (2004-02-10) [2007-04-10]. http://www.w3.org/TR/owl-guide.

[10] Levenshtein V. Binary codes capable of correcting deletions, insertions, and reversals [J]. *Soviet Physics Doklady*, 1966, **10**(8): 707 - 710.

一种基于本体的语义信息集成方法

杨先娣[1,2] 何 宁[1,2] 吴黎兵[1,2] 刘君强[3]

([1] 武汉大学计算机学院,武汉 430072)
([2] 武汉大学计算中心,武汉 430072)
([3] 武汉大学软件工程国家重点实验室,武汉 430072)

摘要:针对信息集成中的语义异构问题,提出了一个基于本体的语义信息集成模型 OSII,并给出了逻辑框架. OSII 采用混和本体方式建模,以 OWL 描述本体,通过局部本体与全局本体之间的映射获得多源统一视图. 提出了一种基于树结构的多策略本体映射算法,该算法包含 4 个步骤,即预处理,名称映射,子树映射和映射矫正. 其特点在于:按照数据类型分类进行映射,并采用启发式规则,提高映射效率;同时考虑概念的语言相似性和结构相似性,提高相似度计算的准确性;采用迭代矫正,最终得到正确而完整的映射对. 通过一个挑战性的实例说明了算法的有效性. OSII 能很好地解决信息集成中的语义异构难点,实现多信息源之间的互操作.

关键词:信息集成;语义异构;本体;本体映射

中图分类号:TP391