

Combining Declarative and Procedural Knowledge to Automate and Represent Ontology Mapping

Li Xu¹, David W. Embley², and Yihong Ding²

¹ Department of Computer Science,
University of Arizona South, Sierra Vista, Arizona 85635, U.S.A.
{lxu}@email.arizona.edu

² Department of Computer Science,
Brigham Young University, Provo, Utah 84602, U.S.A.
{embley, ding}@cs.byu.edu

Abstract. Ontologies on the Semantic Web are by nature decentralized. From the body of ontology mapping approaches, we can draw a conclusion that an effective approach to automate ontology mapping requires both data and metadata in application domains. Most existing approaches usually represent data and metadata by ad-hoc data structures, which lack formalisms to capture the underlying semantics. Furthermore, to approach semantic interoperability, there is a need to represent mappings between ontologies with well-defined semantics that guarantee accurate exchange of information. To address these problems, we propose that domain ontologies attached with extraction procedures are capable of representing knowledge required to find direct and indirect matches between ontologies. And mapping ontologies attached with query procedures not only support equivalent inferences and computations on equivalent concepts and relations but also improve query performance by applying query procedures to derive target-specific views. We conclude that a combination of declarative and procedural representation based on ontologies favors the analysis and implementation for ontology mapping that promises accurate and efficient semantic interoperability.

1 Introduction

Ontologies on the Semantic Web, by nature, are decentralized and built independently by distinct groups. The research on *ontology mapping* is to compare ontological descriptions for finding and representing semantic affinities between two ontologies. By analyzing the body of ontology mapping approaches [2, 6–8, 12, 13, 15, 16, 18], a key conclusion is that an effective ontology mapping approach requires a principled combination of several base techniques such as linguistic matching of vocabulary terms of ontologies, detecting overlap in the choice of data types and representation of data values populated in ontologies, considering patterns of relationships between ontology concepts, and using domain knowledge [13].

To support knowledge sharing between base ontology-mapping techniques, a knowledge base that describes domain models is of great value. The knowledge bases in most existing approaches, however, are represented informally by ad-hoc data structures, in which it is difficult to effectively capture well defined semantics. To further facilitate interoperability between ontologies, there is a need to represent mappings such that the mapping representation guarantees successful exchange of information. The research

work that addressed this ontology-mapping representation problem is usually done separately from the research that focuses on finding semantic affinities [3, 10, 11, 14]. The separation results in lack of support for an efficient approach to achieve interoperability on the Semantic Web. To approach these problems within one knowledge-representation framework, we argue that a combination of declarative and procedural representation based on ontologies favors the analysis and implementation for ontology mapping and promises accurate and efficient semantic interoperability.

Our declarative representation for ontology mapping includes (1) domain ontologies that provide semantic bridges to establish communications among base techniques in order to find semantic affinities between ontologies; and (2) mapping ontologies that provide means to correctly exchange information. Declaratively, ontologies are usually expressed in a logic-based language so that detailed, accurate, consistent, sound, and meaningful distinctions are possible among concepts and relations. Their logic base promises proper reasoning and inference on ontologies.

Ontology mapping requires more than the declarative expressivity of ontologies. One reason is that ontologies have difficulties to effectively express exceptions and uncertainties [17]. Semantic heterogeneity among ontologies is caused by exceptions and uncertainties. Hence, the capability of handling exceptions and uncertainties is extremely important for ontology mapping since the goal of ontology mapping is to find and represent semantic affinities between semantically heterogeneous ontologies. Moreover, to support interoperability across ontologies, based on a debate on the mailing list of the IEEE Standard Upper Ontology working group,³ semantic interoperability is to use logic in order to guarantee that, after data are transmitted from a sender system to a receiver, all implications made by one system must hold and must be provable by the other, and that there should be a logical equivalence between those implications [11]. To express equivalent concepts and relations between two ontologies, queries have to be issued to compute views over ontologies since ontologies rarely match directly [18]. The associated set of logic inference rules with ontologies, however, support neither expressing complex queries nor reasoning about queries efficiently.

Procedural attachment is a common used technique to enforce the expressive power in cases where expression power is limited [17]. A *procedural attachment* is a method that is implemented by an external procedure. We employ two types of procedural attachments in our approach. A domain ontology shared by base ontology-mapping techniques is attached with *extraction procedures*. An extraction procedure is an encoded method with extraction patterns, which express both vocabulary terms and lexical instantiations of ontology concepts. A mapping ontology, on the other hand, is attached with *query procedures* to establish a communication across ontologies. Each mapping instance maps a *source* ontology to a *target* ontology, which is guaranteed to correctly transmit the source data to the target. A *query procedure* computes a target-specific view over the source so that the view data satisfies all implications made by the target.

In this paper, we offer the following contributions: (1) attaching extraction procedures with domain ontologies to represent knowledge shared by base techniques to find semantic affinities between ontologies; and (2) attaching query procedures with mapping ontologies to efficiently interoperate among heterogeneous ontologies based on

³ Message thread on the SUO mailing list initiated at <http://suo.ieee.org/email/msg07542html>.

mapping results produced by base techniques. We present the details of our contribution as follows. Section 2 describes elements in input and domain ontologies and how to apply domain ontologies to find semantic affinities between ontologies. Section 3 describes source-to-target mappings as mapping ontologies and how the representation supports accurate and efficient semantic interoperability. Section 4 gives an experimental result to demonstrate the contribution of applying domain ontologies to ontology mapping. Finally, we summarize and draw conclusions in Section 5.

2 Domain Model Representations

2.1 Input Ontology

An ontology include classes, slots, slot restrictions, and instances [4]. A class is a collection of entities. Each entity of the class is said to be an instance of that class. With “IS-A” and “PART-OF” relationships, classes constitute a hierarchy. Slots attached to a class describe properties of objects in the class. Each slot has a set of restrictions on its values, such as cardinalities and ranges. By adopting an algebra approach to represent ontologies as logical theories [11], we use the following definition for input ontologies.

Definition 1. An *input ontology* $O = (S, A, F)$, where S is the *signature* that describes the vocabulary for classes and slots, A is a set of *axioms* that specify the intended interpretation of the vocabulary in some domain of discourse, and F is a set of *ground facts* that classifying instances with class and slot symbols in the signature S . \square

For discussion convenience, in this paper we use rooted hypergraphs graphs to illustrate structure properties between classes and slots in ontological signatures. A hypergraph includes a set of nodes modeling classes and slots and a set of edges modeling relations between them. The root node is representing a designated class of primary interest. Figure 1, for example, shows two ontology hypergraphs (whose roots are *house* and *House*). In hypergraphs, we present a class or slot using either a solid box or a dashed one. A solid box indicates that there are object identifiers populated for the concept. And a dashed box indicates that there is lexical data populated for the concept. A functional relation using a line with an arrow from its domain to its range, and a nonfunctional relation using a line without arrowhead.

2.2 Domain Ontology

To accommodate knowledge requirements of ontology mapping, we define domain ontologies as follows.

Definition 2. A *domain ontology* $O = (S, A, P)$, where S is the ontological *signature*, A is a set of ontological *axioms*, and P is a set of *extraction procedures* that extract metadata and data from vocabulary terms and populated instances of input ontologies based on extraction rules⁴. \square

⁴ Ground facts are not part of a domain ontology since a domain ontology is not populated with instances.

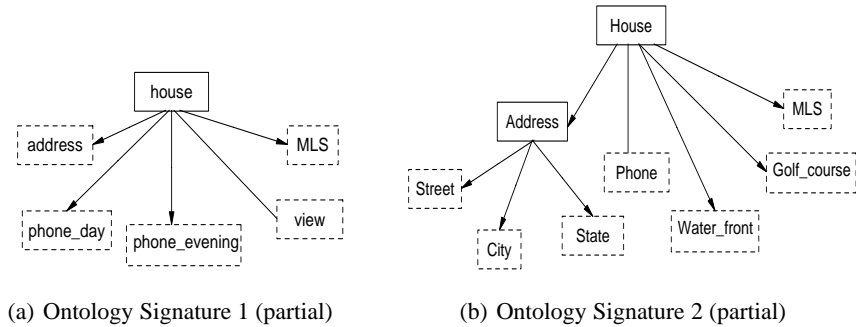


Fig. 1. Signatures of Input Ontologies

A domain ontology is used to establish a semantic bridge in order to find semantic affinities between ontologies. Extraction procedures attached with domain ontologies apply data extraction techniques [9] to retrieve data and metadata when matching two ontologies. Each extraction procedure is designed for either a class or slot in a domain ontology. When an extraction procedure is invoked, a recognizer does the extraction by applying a set of extraction rules specified using regular expressions. Figure 2 shows the regular expressions using the Perl syntax for slot *View* and *Phone* in a real-estate domain.

Each list of regular expressions include declarations for data values that can potentially populate a class or slot and keywords that can be used as vocabulary terms to name classes and slots. We describe data values using *extract* clauses and the keywords using *keyword* clauses. When applied to an input ontology, both the *extract* and *keyword* clauses causes a string matching a regular expression to be extracted, where the string can be a vocabulary term in the ontological signature or a data values classified by the ontological ground facts.

2.3 Application of Domain Ontology

Figure 3 shows three components in a real-estate domain ontology, which we used to automate the mapping between two ontologies in Figure 1 and also for mapping real-world ontologies in the real-estate domain in general. Each box in Figure 3 associates with an extraction procedure. Filled-in (black) triangles denote aggregation (“PART-OF” relationships). And open (white) triangles denote generalization/specialization (“IS-A” superclasses and subclasses).

Provided with the domain ontology described in Figure 3, we can discover many semantic affinities between Ontology 1 in Figure 1(a) and Ontology 2 in Figure 1(b) as follows.

1. *Merged/Split Values.* Based on the *Address* declared in the ontology in Figure 3(a), the attached extraction procedure detects that (1) the values of *address* in Ontology 1 match with extraction patterns for concept *Address*, and (2) the values of *Street*, *City*, and *State* in Ontology 2 match with extraction patterns for concepts *Street*,

```

View matches [15] case insensitive
constant
  { extract "\bmountain\sview\b"; },
  { extract "\bwater\sfront\b"; },
  { extract "\briver\sview\b"; },
  { extract "\bpool\sview\b"; },
  { extract "\bgolf\s*course\b"; },
  { extract "\bcoastline\sview\b"; },
  ...
  { extract "\bgreenbelt\sview\b"; };
keyword
  "\bview(s)?\b";
End;
Phone matches [15] case insensitive
constant
  { extract "\b\d{3}-\d{4}\b"; }, - nnn-nnnn
  { extract "\b(\d{3})\s*\d{3}-\d{4}\b"; }, - (nnn) nnn-nnnn
  { extract "\b\d{3}-\d{3}-\d{4}\b"; }, - nnn-nnn-nnnn
  { extract "\b\d{3}\\\d{3}-\d{4}\b"; }, -nnn\nnn-nnnn
  { extract "\b1-\d{3}-\d{3}-\d{4}\b"; }; - 1-nnn-nnn-nnnn
Keyword
  "\bcall\b","bphone\b";
End;

```

Fig. 2. Example of regular expressions in a real-estate domain

City, and *State* respectively. Based on “PART-OF” relationships in Figure 3(a), we can find the “PART-OF” relationships between *Street*, *City*, and *State* in Ontology 2 and *address* in Ontology 1.

2. *Superset/Subset*. By calling extraction procedures attached with classes in Figure 3(b), *phone_day* in Ontology 1 matches both keywords and data value patterns for *Day Phone* and *phone* in Ontology 2 matches with *Phone*. In Figure 3(b) the ontology explicitly declares *Phone* is a superset of *Day Phone* based on the “IS-A” relationship between *Day Phone* and *Phone*. Thus we can find the semantic affinity between *phone_day* in Ontology 1 and *Phone* in Ontology 2.
3. *Vocabulary Terms and Data Instances*. Extraction procedures apply extraction patterns to recognize keywords and value patterns over both ontology terms and populated data instances since it is difficult to distinguish boundaries between meta-data and populated data instances in complex knowledge representation systems. In Ontology 1, “*water front*” is instance data populated for *view*. In Ontology 2, *Water_front* is a vocabulary term. Boolean values “Yes” and “No” for *Water_front* in Ontology 2 indicate whether the values *Water_front* should be included as description values for *view* of *House* in Ontology 1 if we map the two ontologies. The extraction procedure for concept *View* in Figure 3(c) recognizes terms such as *Water_front* in Ontology 2 as values and the procedure for concept *Water Front* can also recognize keyword “*water front*” associated with *view* in Ontology 1. Since *Water Front* “IS-A” *View* in Figure 3(c), by derivation, we

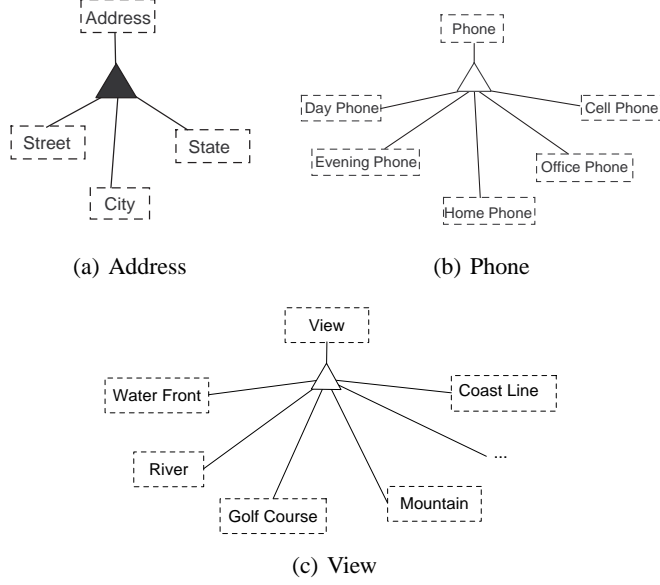


Fig. 3. Real-estate domain ontology (partial)

can detect that *view* in Ontology 1 has a semantic affinity with *Water_front* in Ontology 2.

3 Mapping Result Representation

3.1 Source-to-target Mapping

We adopt an ontology mapping definition [11] as follows.

Definition 3. A *source-to-target mapping* M_{ST} from $O_S = (S_S, A_S, F_S)$ to $O_T = (S_T, A_T, F_T)$ is a morphism $f(S'_S) = S'_T$ such that $A'_T \models f(A'_S)$, i.e. all interpretations that satisfy O'_T axioms also satisfy O'_S translated axioms if there exist two sub-ontologies $O'_S = (S'_S, A'_S, F'_S)$ ($S'_S \subseteq S_S, A'_S \subseteq A_S, F'_S \subseteq F_S$) and $O'_T = (S'_T, A'_T, F'_T)$ ($S'_T \subseteq S_T, A'_T \subseteq A_T, F'_T \subseteq F_T$). \square

Our representation solution for source-to-target mapping allows a variety of source derived data based on the discovered semantic affinities between two input ontologies. These source derived data include generalizations and specializations, merged and split values, translations between vocabulary terms and data instances, and etc. Therefore, our solution “extends” elements in an ontological signature S_S of a source ontology O_S by including views computed via queries, each of which we call a *view* element. We let V_S denote the extension of S_S with derived, source view elements.

Every source-to-target mapping M_{ST} is composed of a set of triples. Each triple $t = (e_t, e_s, q_e)$ is a *mapping element*, where $e_t \in S_T$, $e_s \in V_S$, q_e is either empty or a *mapping expression*. We call a triple $t = (e_t, e_s, q_e)$ a *direct match* which binds

$e_s \in S_S$ to $e_t \in S_T$, or an *indirect match* which binds a view element $e_s \in V_S - S_S$ to $e_t \in S_T$. When a mapping element t is an indirect match, q_e is a *mapping expression* to illustrate how to compute the view element e_s over the source ontology O_S .

To represent source-to-target mapping as logic theories, we specify source-to-target mappings as populated instances of a mapping ontology, which we define as follows.

Definition 4. A mapping ontology $O = (S, A, F, P)$, where S is the ontological signature, A is the set of ontological axioms, F is a set of *ground facts* presenting source-to-target mappings, and P is a set of *query procedures* that describe designed query behaviors to compute views over ontologies. \square

If a mapping element $t = (e_t, e_s, q_e)$ in a source-to-target mapping M_{ST} is an indirect match, i.e. e_s is a source view element, a query procedure is attached with t to compute e_s by applying the mapping expression q_e .

3.2 Mapping Expressions

We can view each class and class slot (including view elements corresponding to either classes or slots) in ontologies as single-attribute or multiple-attribute relations. Relational algebra is therefore ready to be applied to describe procedural behaviors for query procedures. Since traditional operators in relational algebra do not cover the ones required to address problems such as *Merged/Split values* and *Vocabulary Terms and Data Instances*, we present mapping expressions by an extended relational algebra.

For example, to address *Merged/Split Values*, we design two operations *Composition* and *Decomposition* in the extended relational algebra, which we describe as follows. In the notation, a relation r has a set of attributes; $attr(r)$ denotes the set of attributes in r ; and $|r|$ denotes the number of tuples in r .

- *Composition* λ . The λ operator has the form $\lambda_{(A_1, \dots, A_n), A} r$ where each A_i , $1 \leq i \leq n$, is either an attribute of r or a string, and A is a new attribute. Applying this operation forms a new relation r' , where $attr(r') = attr(r) \cup \{A\}$ and $|r'| = |r|$. The value of A for tuple t on row l in r' is the concatenation, in the order specified, of the strings among the A_i 's and the string values for attributes among the A_i 's for tuple t' on row l in r .
- *Decomposition* γ . The γ operator has the form $\gamma_{A, A'}^R r$ where A is an attribute of r , and A' is a new attribute whose values are obtained from A values by applying a routine R . Applying this operation forms a new relation r' , where $attr(r') = attr(r) \cup \{A'\}$ and $|r'| = |r|$. The value of A' for tuple t on row l in r' is obtained by applying the routine R on the value of A for tuple t' on row l in r .

Assuming that Ontology 1 in Figure 1(a) is the target and Ontology 2 in Figure 1(b) is the source, the following lists the derivation of a view element *House – address'* in Ontology 2 that matches with *house – address* in Ontology 1.

$$\begin{aligned} \text{Address} - \text{Address}' &\leftarrow \pi_{\text{Address}, \text{Address}'} \lambda_{(\text{Street}, ", ", \text{City}, ", ", \text{State}), \text{Address}'} (\\ &\quad \text{Address} - \text{Street} \bowtie \text{Address} - \text{City} \bowtie \text{Address} - \text{State}) \\ \text{House} - \text{address}' &\leftarrow \rho_{\text{Address}' \leftarrow \text{address}'} \pi_{\text{House}, \text{Address}'} (\text{House} - \text{Address} \\ &\quad \bowtie \text{Address} - \text{Address}') \end{aligned}$$

The λ operator denotes the *Composition* operation in the relational algebra. The Composition operation merges values in *Street*, *City* and *State* for a new concept *Address'*. The derivation also applies standard operations including *Natural Join* \bowtie , *Projection* π , and *Rename* ρ .

3.3 Semantic Interoperability

We define a semantic interoperable system as follows.

Definition 5. A *semantic interoperable system* $I = (O_T, \{O_{S_i}\}, \{M_{S_i T}\})$, where O_T is a target ontology, $\{O_{S_i}\}$ is a set of n source ontologies, and $\{M_{S_i T}\}$ is a set of n source-to-target mappings, such that for each source ontology O_{S_i} there is a mapping $M_{S_i T}$ from O_{S_i} to O_T , $1 \leq i \leq n$. \square

Note that data instances $F_{O_{S_i} \rightarrow O_T}$ flowing from any source O_{S_i} to the target O_T based on $M_{S_i T}$ hold classifications to either signature or view elements in O_{S_i} . Since a source-to-target mapping defines a morphism $f(S'_{O_{S_i}}) = S'_{O_T}$, the data instances $F_{O_{S_i} \rightarrow O_T}$ hence hold the classifications to the signature elements in O_T that correspond source elements in O_{S_i} .

The following theorem provides that accurate information exchange between ontologies is guaranteed by derived source-to-target mappings.

Theorem 1. Given a *semantic interoperable system* $I = (O_T, \{O_{S_i}\}, \{M_{S_i T}\})$ where $1 \leq i \leq n$, data instances $F_{O_{S_i} \rightarrow O_T}$ flowing from O_{S_i} to O_T by $M_{S_i T}$ hold and are provable by O_T . \square

Assume that user queries issued over I are Select-Project-Join queries and we also assume that they do not contain comparison predicates such as \leq and \neq . We use the following standard notation for conjunctive queries.

$$Q(\overline{X}) : -P_1(\overline{X}_1), \dots, P_n(\overline{X}_n)$$

Where \overline{X} , $\overline{X}_1, \dots, \overline{X}_n$ are tuples of variables, $\overline{X} \subseteq \overline{X}_1 \cup \dots \cup \overline{X}_n$, and predicate P_i ($1 \leq i \leq n$) is a target signature element.

When evaluating query answers for a user query Q , the semantic interoperable system I transparently reformulates Q as Q^{ext} , a query over the target and source ontologies in I . The source-to-target mapping instances lead automatically to a rewriting of every target element as a union of the target element and corresponding source elements. Query reformulation thus reduces to rule unfolding by applying the view definition expressions for the target elements in the same way database systems apply view definitions.

With query reformulation in place, we can now prove that query answers are *sound*—every answer to a user query Q is an entailed fact according to the source(s) and the target—and that query answers contain all the entailed facts for Q that the sources and the target have to offer—*maximal* for the query reformulation.

Theorem 2. Let Q_I^{ext} be the query answers obtained by evaluating Q^{Ext} over I . Given a user query Q over I , a tuple $\langle a_1, a_2, \dots, a_M \rangle$ in Q_I^{Ext} is a sound answer for Q . \square

Theorem 3. Let Q_I^{ext} be the query answers obtained by evaluating Q^{Ext} over I . Given a user query Q over I , Q_I^{Ext} is maximal for Q with respect to I . \square

4 Experimental Result

We used a real-world application, *Real Estate*, to evaluate applications of a domain ontology shared by a set of matching technique [18]. The *Real Estate* application has five ontologies. We decided to let any one of the ontologies be the target and let any other ontology be the source. In summary, we tested 20 pairs of ontologies for the *Real Estate* application. In the test, *Merged/Split Values* appear four times, *Superset/Subset* appear 48 times, and *Vocabulary Terms and Data Instances* appear 10 times. With all other indirect and direct matches, there are a total of 876 matches. We evaluated the performance of our approach based on three measures: precision, recall and the F-measure, a standard measure for recall and precision together [1]. By exploiting knowledge specified in the domain ontologies attached with extraction procedures, the performance reached 94% recall, 90% precision, and an F-measure of 92%⁵.

One obvious limitation to our approach is the need to manually construct an application-specific domain ontology with extraction procedures⁶. To facilitate the knowledge acquiring process to build domain ontologies, we can reuse existing ontologies. Machine learning techniques can also be applied to facilitate the construction of extraction patterns for extraction procedures. Since we predefine a domain ontology for a particular application, we can compare any two ontologies for the application using the same domain ontology. Therefore, the work of creating a domain ontology is amortized over repeated usage.

5 Conclusions

We have proposed an approach to automate and represent ontology mappings by combining both declarative and procedural representations. With experiments we have shown that a set of base techniques is able to establish communications via domain ontologies attached with extraction procedures. By sharing domain ontologies, the base techniques detected indirect matches related to problems such as *Superset/Subset*, *Merged/Split values*, as well as *Vocabulary Terms and Data Instances*. To approach semantic interoperability across ontologies, we present source-to-target mappings as mapping ontologies attached with query procedures, which not only support equivalent inferences and

⁵ See a detailed explanation about the experiment in [18]

⁶ Experience has shown that computer science students can build a domain ontology with extraction rules for a data-rich, narrow domain of interest in a few dozen person hours. Students have built ontologies for a wide variety of applications including such diverse areas such as digital cameras, prescription drugs, campgrounds, and computer jobs [5].

computations on equivalent concepts and relations but also improve query performance by applying query procedures.

References

1. R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, Menlo Park, California, 1999.
2. J. Berlin and A. Motro. Database schema matching using machine learning with feature selection. In *Proceedings of the International Conference on Advanced Information Systems Engineering (CAISE 2002)*, pages 452–466, Toronto Canada, 2002.
3. D. Calvanese, G. De Giacomo, and M. Lenzerini. A framework for ontology integration. In *Proceedings of the 1st Internationally Semantic Web Working Symposium (SWWS)*, pages 303–317, 2001.
4. V.K. Chaudhri, A. Farquhar, R. Fikes, P.D. Karp, and J.P. Rice. OKBC: a programmatic foundation for knowledge base interoperability. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98)*, Madison, Wisconsin, 1998.
5. Demos page for BYU data extraction group. <http://www.deg.byu.edu/multidemos.html>.
6. R. Dhamankar, Y. Lee, A. Doan, A. Halevy, and P. Domingos. iMAP: Discovering complex semantic matches between database schemas. In *Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data (SIGMOD 2004)*, pages 283–294, Paris, France, June 2004.
7. H. Do and E. Rahm. COMA - a system for flexible combination of schema matching approaches. In *Proceedings of the 28th International Conference on Very Large Databases (VLDB)*, pages 610–621, Hong Kong, China, August 2002.
8. A. Doan, J. Madhavan, R. Dhamankar, P. Domingos, and A. Halevy. Learning to match ontologies on the semantic web. *VLDB Journal*, 12:303–319, 2003.
9. D.W. Embley, D.M. Campbell, Y.S. Jiang, S.W. Liddle, D.W. Lonsdale, Y.-K. Ng, and R.D. Smith. Conceptual-model-based data extraction from multiple-record Web pages. *Data & Knowledge Engineering*, 31(3):227–251, November 1999.
10. A.Y. Halevy. Answering queries using views: A survey. *The VLDB Journal*, 10(4):270–294, December 2001.
11. Y. Kalfoglou and M. Schorlemmer. Ontology mapping: the state of the art. *The Knowledge Engineering Review*, 18(1):1–31, 2003.
12. W. Li and C. Clifton. SEMINT: A tool for identifying attribute correspondences in heterogeneous databases using neural networks. *Data & Knowledge Engineering*, 33(1):49–84, April 2000.
13. J. Madhavan, P.A. Bernstein, A. Doan, and A. Halevy. Corpus-based schema matching. In *ICDT'05*, y, 2005.
14. J. Madhavan, P.A. Bernstein, P. Domingos, and A. Halevy. Representing and reasoning about mappings between domain models. In *Proceedings of the 18th National Conference on Artificial Intelligence (AAAI'02)*, 2002.
15. A. Maedche, B. Motic, N. Silva, and R. Volz. Mafra - an ontology mapping framework in the semantic web. In *Proceedings of the ECAI Workshop on Knowledge Transformation*, Lyon, France, July 2002.
16. E. Rahm and P.A. Bernstein. A survey of approaches to automatic schema matching. *The VLDB Journal*, 10(4):334–350, December 2001.
17. S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education, Inc., second edition edition, 2003.
18. L. Xu and D.W. Embley. A composite approach to automating direct and indirect schema mappings. *Information Systems*, available online April 2005.