Reverse Ontology Matching

Jorge Martinez-Gil
University of Malaga
Dept. of Computing Sciences
Boulevard Louis Pasteur 35, 29071 Malaga
jorgemar@lcc.uma.es

Jose F. Aldana-Montes
University of Malaga
Dept. of Computing Sciences
Boulevard Louis Pasteur 35, 29071 Malaga
jfam@lcc.uma.es

ABSTRACT

Ontology Matching aims to find the semantic correspondences between ontologies that belong to a single domain but that have been developed separately. However, there are still some problem areas to be solved, because experts are still needed to supervise the matching processes and an efficient way to reuse the alignments has not yet been found. We propose a novel technique named Reverse Ontology Matching, which aims to find the matching functions that were used in the original process. The use of these functions is very useful for aspects such as modeling behavior from experts, performing matching-by-example, reverse engineering existing ontology matching tools or compressing ontology alignment repositories. Moreover, the results obtained from a widely used benchmark dataset provide evidence of the effectiveness of this approach.

1. INTRODUCTION

In the new approaches to develop information systems, the use of a type of formal schema called ontology is usual. Ontologies are considered to be semantically richer than schemas in general, and therefore, techniques for schema matching can be easily adapted to ontologies but not vice versa [12].

There are many ontologies available on the web currently. These ontologies are usually developed for different collections of information, and different kinds of applications. Nowadays, the Swoogle search engine[1] has indexed thousands of ontologies. There are several reasons for the quick proliferation of ontologies, but we consider mainly two:

- It is often easier to construct a new ontology, than find an existing one which is appropriate for a given task.
- There is often a desire for direct control over the ontology for a particular domain, rather than having the structure dictated by external forces.

[1]http://swoogle.umbc.edu

A direct consequence of having large numbers of ontologies available is that it is necessary to integrate knowledge which is represented in different ways. Ontology matching aims to produce alignments, that is, sets of semantic correspondences between elements from different ontologies. This task is very expensive in terms of time and resource consumption. The reason is that it is necessary a lot of work from domain experts to match ontologies or to supervise results from existing semiautomatic tools. Our approach is based on the extraction of the ontology matching functions used by the agents, experts or tools when matching ontologies, so it is a powerful way to reuse, store and understand their knowledge. Moreover, there are other collateral benefits as the ability to implement strategies for ontology matching by example, reverse engineering existing ontology matching tools or compress large ontology alignment repositories. In this way, we think that the main contributions of our work can be summarized as follow:

- We propose, for the first time to the best of our knowledge, a methodology for reverse engineering an ontology alignment which tries to find the matching function that have been used to generate an ontology alignment.
- We perform an empirical evaluation of our approach in order to show its practical viability in the real world.

The rest of this work is structured in the following way: Section 2 describes the problem statement related to Reverse Ontology Matching. Section 3 presents the related works regarding other reverse engineering proposals. Section 4 presents the core of our approach, a methodology for reverse engineer an ontology alignment, and some real examples. Section 5 contains an evaluation that shows the applicability of Reverse Ontology Matching in the practice. In Section 6, we describe the conclusions extracted from this work.

2. PROBLEM STATEMENT

An ontology is "*a specification of a conceptualization*" [9] that it is to say, an abstract representation of the world like a set of objects. In this work, we are going to use the intuitive notion of ontology as a set of classes with relations among them that serves primarily to represent formal knowledge in a way which is understandable by people and machines.

Definition 1 (Ontology Matching Function). *Ontology Matching Function is a function f where, given two input ontologies o and o', an (optional) input incomplete alignment A, a set of configuration parameters p and, a set of external resources r, an alignment y is returned.*

$$y = f(o, o', A, p, r) \tag{1}$$

Definition 2 (Ontology Alignment). *An ontology alignment is a set of mappings, thus, a set of tuples in the form (id, e, e', n, R). Where id is an unique identifier, e and e' are entities belonging to two different ontologies, R is the relation of correspondence between these entities and n is a real number between 0 and 1 representing the confidence for R.*

Definition 3 (Reverse Ontology Matching Function). *We define Reverse Ontology Matching as the function g that has been used for obtaining an alignment y using two input ontologies o and o', an (optional) input incomplete alignment A, a set of configuration parameters p and, a set of external resources r according to the following equation:*

$$y = f(o, o', A, p, r) \rightarrow \exists g,\ g(o, o', A, p, r, y) = f \tag{2}$$

The computation of f is far from being trivial. There are hundreds of algorithms to match ontologies, and everything indicates that more algorithms will appear. Moreover, matching algorithms can be compose so we obtain a solution space populated by many millions of possibilities. Lastly, in alignments created by humans it is possible to find mappings that have been found using heterogeneous rules [13]. Therefore, we cannot be sure that we are going to obtain the original function, but a function that is equivalent to the original one.

Definition 4 (Equivalent Ontology Matching Function). *Let f be an ontology matching function, then we define an Equivalent Ontology Matching Function f' as a function which return the same result that f for the same input. More formally*

$$f' \equiv f \leftrightarrow f'(o, o', A, p, r) = f(o, o', A, p, r) \tag{3}$$

These two basic ideas behind the notions of Reverse Ontology Matching Function and Equivalent Ontology Matching Function allow us to formulate the definition Equivalent Reverse Ontology Matching Function as follows.

Definition 5 (Equivalent Reverse Ontology Matching Function). *Let g be a reverse ontology matching function, then we define a Equivalent Reverse Ontology Matching Function g' as an function which return the same result that g for the same input. More formally*

$$g' \equiv g \leftrightarrow g'(o, o', A, p, r, y) = g(o, o', A, p, r, y) \tag{4}$$

Take into account, that given an alignment between two ontologies, we know nothing about the heuristic used for the expert in order to provide the results. However, we know two main things: firstly, in Ontology Matching there are a limited amount of categories for grouping algorithms with similar behaviors, and secondly, we understand the notion of composite matchers, that it is to say, the idea behind to combine similarity values predicted by multiple algorithms to determine correspondences between ontology elements in order to benefit from the high degree of precision of some algorithms and at the same time the broader coverage of other algorithms [6]. For the rest of this work, we are working under the assumption that the agent, expert or tool which generate an initial alignment always try to maximize the precision and coverage of their solutions.

2.1 Use Cases

There are many applications where reverse ontology matching will be very useful. We are going to show here four of them: a) Capturing behavior from experts, b) Matching by example, c) Reverse engineer existing tools, and d) Compressing large volumes of ontology alignments.

2.1.1 *Capturing behavior from experts.*

If we ask an expert for creating mappings between two ontologies that belong to its area of expertise but that have been developed separately, we are going to obtain a few correspondences but nothing else. These correspondences are only useful for the current case and we are not going to be able to get profit from them in the future. But, if we reverse engineer these correspondences, we will be able to obtain the heuristic used by the expert and apply it in a lot of additional scenarios. Moreover, we can become experts because we are not going to see only results but the way to reach these results. Finally, but not least, we can compare heuristics from a wide variety of experts and obtain easily a core heuristic, thus, a common way to solve problems.

2.1.2 *Matching by example.*

In many ontology matching scenarios is popular the use of a technique called matching by example. This technique consists of given two ontologies, try to find several samples correspondences in order to the system may learn how to find the rest of existing correspondences between the two ontologies automatically. In this way, the user only has to do a little part of the work manually. The existing techniques use methods from the machine learning field (e.g. genetic algorithms, neural networks, and so on). In this way, better the set of mappings provided by the user larger the quality of the automatic matching to be performed. One of the advantages of reverse ontology matching functions is that can be computed in real time, so it is possible to compute the equivalent reverse matching function for a little set of mappings in order to apply this function to the rest of the given ontologies. If the user is able to provide all possible cases initially, the automatic part of the matching process will be very good.

2.1.3 *Reverse engineer existing tools.*

Author of the initial set of mappings is not relevant for our reverse ontology matching approach. This means that is possible to detect, and therefore to simulate, an equivalent working mode for the most of deterministic ontology matching tools. Deterministic here means that for a given input, the same output is always provided. The reason is that our approach evaluates some sample inputs and outputs for these tools, and then, configures a deterministic black box which uses well-known techniques to generate the same results for the initially given sample inputs. This technique can be useful to analyze and categorize existing tools. Larger our knowledge of these tools larger the possibility to find errors or improve them.

2.1.4 *Compressing large ontology alignments.*

There are many repositories of ontology alignments available on the web. The problem when storing an ontology alignment is that it is necessary to store a lot of information which a) needs much disk space and b) is very difficult to reuse. The reason for the first fact is that it is necessary to store the mappings, information regarding to the initial ontologies, related overhead, and so on. Secondly, knowledge contained in the alignment only can be reused when comparing the same correspondences. Storing only the function that was used to generate the alignment can save much disk space (only a function is stored), contains the same knowledge that the alignment (the alignment can be generated again using the function), and is very reusable (the function can be used in other scenarios).

3. RELATED WORKS

The importance of ontology matching is evidenced by the large number of related works that have been made. Unable to cite all these works, we reference the most important surveys in this field, [3, 5, 7, 11, 14, 16] where ontology matching methods and tools are described. There are several improvements like the possibility to match very large ontologies [10] or the capability to make predictions [15].

Many authors tend to categorize simple ontology matching algorithms in the groups defined by Ehrig [5], thus, they try to categorize basic matching algorithms in four categories corresponding to the ontology features to exploit, i.e. Linguistic Features, Structural Features, Constraint-based Features and Integration-Knowledge-based features.

On the other hand, we have not found works addressing the problem of the reverse ontology matching. However, the problem has been treated in adjacent fields such as data exchange. For example, Fagin et al [8] developed a framework for reverse data exchange that supports source instances that may contain nulls. This development required a careful reformulation of all the important notions, including the identity schema mapping, inverse, and maximum recovery. Like in our approach, operators originally introduced by Arenas et al.[1, 2], thus, the composition operator and the inverse operator have been recognized as two fundamental ones.

4. REVERSE ONTOLOGY MATCHING

It is possible to compute an equivalent reverse matching function for the alignments that have been created using several of techniques surveyed previously, either they have been combined in a parallel or in a sequential way. Algorithm combination means that algorithms are considered independently of each other, instead of algorithm composition which consists of using several algorithms in order to create a new one (hybrid algorithm). The way to obtain this equivalent reverse matching function requires four main steps that are going to be described now. It should be taken into account that, although engineering details are outside the aim of this work, these steps are susceptible to automation.

1. Choose the set of algorithms which are going to be used to obtain the equivalent matching function
2. Apply rules for computing the matching function and generating an intermediate equation
3. Obtain the equivalent matching function from the intermediate equation
4. Simplify (if possible) the equivalent reverse ontology matching function

On the other hand, if the mappings are given in a probabilistic form, firstly we have to decide a threshold in order to identify the valid ones. Then, we can proceed with point number one. A future improvement could consist of managing the uncertainty inherent in these mappings.

4.1 Choosing the set of algorithms

Choosing an appropriate set of algorithms to compose the equivalent matching function is very important in order to get success in the reverse matching process. Ideally, we need to use all existing matching algorithms, but this choice is not viable in practice, so we propose to choose, at least, a representative from each of the categories, although it is possible to choose hybrid algorithms too.

4.2 The equivalent matching function

In this step, it is necessary to apply several rules to know the algorithms that has been used originally to perform the alignment. There rules are:

Rule 1. *All algorithms which satisfy a mapping will be included in the intermediate equation.*

Rule 2. *All algorithms which satisfy a same input will be combined in a sequential way, thus, using the operator AND.*

Rule 3. *All algorithms, or set of algorithms, which satisfy two different inputs will be combined parallely, thus, using the operator OR.*

After applying these rules, there might be mappings which cannot be found using the algorithms included in the previous step. For this reason it is necessary to define the concept of magic mapping.

Definition 5 (Magic mapping). *We define a magic mapping as the tuple (id, e, e', n, R) belonging to an alignment A so we do not know what algorithm was used to find it.*

The notion of magic mapping tell us that either we have not used an appropriate set of matching algorithms or the need to design a new matching algorithm that addresses this issue.

4.3 Extraction of the generalization pattern

This step consists of erasing duplicates expressions from the intermediate equation and making free linked variables. In this way, we obtain a clean of redundancies function.

4.4 Reduction Properties

We have borrowed several rules from the boolean algebra in order to reduce the length of the equivalent reverse ontology matching function. In fact, we have classified these rules in two different groups. For expressions with overlapped (set of) algorithms:

$$(a \wedge b) \wedge b \rightarrow (a \wedge b) \tag{5}$$

$$(a \wedge b) \vee b \rightarrow b \tag{6}$$

$$(a \vee b) \wedge b \rightarrow b \tag{7}$$

$$(a \vee b) \vee b \rightarrow (a \vee b) \tag{8}$$

For expressions without overlapped (set of) algorithms:

$$(a \wedge b) \wedge c \rightarrow (a \wedge b \wedge c) \tag{9}$$

$$(a \vee b) \wedge c \rightarrow (a \wedge c) \vee (b \wedge c) \tag{10}$$

4.5 Reverse Ontology Matching in practice

We show here two examples of how reverse ontology matching can be performed in the practice: (a) We extract the equivalent reverse matching function from a set of mappings (as example of capturing expert behavior), (b) we apply the obtained function that we have obtained to find mappings between two lemmaries (as example of matching-by-example).

Example 1. *Given the following set of mappings (Note the mispellings)* {*(Paris, Charles-de-Gaulle), (London, Heathrow), (Berlin, Schonenfeld), (Rome, Romans), (Madrid, Barajas), (Lisboa, Lisbon)*} *compute the equivalent reverse matching function that has been used to generate them.*

1. We are going to choose this set of non-overlapped[2] ontology matching algorithms:

   {*Synonym, 3Grams, Stoilos, Wikip., Google* }

2. If we follow the rules proposed, we are going to obtain the following intermediate equation:

   (Wikipedia (Paris, Charles-de-Gaulle) AND Google (Paris, Charles-de-Gaulle)) OR (Wikipedia (London, Heathrow) AND Google (London, Heathrow)) OR Google (Berlin, Schonenfeld) OR (3-Grams (Rome, Romans) AND Stoilos (Rome, Romans)) OR (Wikipedia (Madrid, Barajas) AND Google (Madrid, Barajas)) OR (3-Grams (Lisboa, Lisbon) AND Stoilos (Lisboa, Lisbon))

3. Now, we obtain the generalization pattern:

   (Wikipedia (c1, c2) AND Google (c1, c2)) OR Google (c1, c2) OR (3-Grams (c1, c2) AND Stoilos (c1, c2))

4. Finally, we apply the appropriate reduction properties ((6) in this case) in order to obtain the final equivalent matching function:

   Google (c1, c2) OR (3-Grams (c1, c2) AND Stoilos (c1, c2))

   What means that all input mappings that meet these conditions will be included in the final alignment. If more complex alignments are going to be analyzed, the two ontologies have to be accessible so that matching algorithms can detect structural similarities.

As it can be seen, we have captured the equivalent reverse matching function that was initially applied by the expert in order to match the concepts. If the function is applied to the input set of concepts, the mappings will be obtained again.

Example 2. *Use the equivalent reverse matching function obtained in the Example 1 to match these two simple lemmaries* {*Canada, Asia, Boston, Mexico, New-York*} *and* {*Celtics, Canadian, MexicoDF, Lakers, Manhattan*}

[2]Two algorithms are overlapped if they aim to exploit the same ontology characteristics when looking for a correspondence

1. The equivalent matching function that we obtained in the Example 1 was:

   Google (c1, c2) OR (3-Grams (c1, c2) AND Stoilos (c1, c2))

2. We generate the set of all possible correspondences between the two given lemmaries:

   {*(Canada, Celtics), (Canada, Canadian), (Canada, MexicoDF), (Canada, Lakers), (Canada, Manhattan), (Asia, Celtics), (Asia, Canadian), (Asia, MexicoDF), (Asia, Lakers), (Asia, Manhattan), (Boston, Celtics), (Boston, Canadian), (Boston, MexicoDF), (Boston, Lakers), (Boston, Manhattan), (Mexico, Celtics), (Mexico, Canadian), (Mexico, MexicoDF), (Mexico, Lakers), (Mexico, Manhattan), (New-York, Celtics), (New-York, Canadian), (New-York, MexicoDF), (New-York, Lakers), (New-York, Manhattan)* }

3. We apply the equivalent matching function to the set of all correspondences and we have,

   - *Google: (Boston, Celtics), (Boston, Lakers), (Mexico, MexicoDF), (New-York, Manhattan)*
   - *3-Grams AND Stoilos: (Canada, Canadian), (Mexico, MexicoDF*)

4. The final set of mappings is

   {*(Canada, Canadian), (Boston, Celtics), (Boston, Lakers), (Mexico, Mexico DF), (New-York, Manhattan)*}

5. We have that Boston belongs to two different mappings. This is because there are a lot of pages referring to NBA indexed by Google, so this algorithm generates a false positive. It is possible to implement the system in two ways: a) Allowing only 1:1 correspondences, in this case, only the mapping with a higher degree of confidence according to the algorithms will be added to the final alignment b) Allowing n:m correspondences, in this case, all mappings that meet the conditions will be included in the final results.

5. EVALUATION

We perform here an evaluation of our proposal. Firstly, we define that way to measure the quality of an equivalent matching function. Then, we describe and discuss the cases that we can find when evaluating this kind of functions, and lastly, we apply our technique in several real world scenarios in order to show that reverse ontology matching is viable in the practice.

Definition 6 (Equivalent reverse matching function evaluation). *An equivalent reverse matching function evaluation ermfe is a function* $ermfe : S \times S' \mapsto precision \in \Re \in [0,1] \times recall \in \Re \in [0,1]$ *that associates an alignment S and a reference alignment S' to two real numbers stating the precision and recall of S in relation to S'.*

Precision states the fraction of retrieved mappings that were included in the original alignment S. Recall is the fraction of the correct mappings that are obtained successfully in comparison with the mappings belonging to S. In this way, precision is a measure of exactness and recall a measure of completeness. The problem here is that techniques can be optimized to obtain a high precision at the cost of the recall or, on the contrary, it is easy to optimize the recall at the cost of the precision. By this reason a F-measure is defined as a weighting factor between precision and recall. In this work, we use the most common configuration which consists of weighting precision and recall equally.

Let S the alignment provided initially, and let emf be the equivalent matching function obtained using reverse engineering and S' its output alignment. Then, we can face to these three cases:

- $S' = S$. We have a perfect equivalent matching function. The reason is that the equivalent matching function has been able to replicate exactly the results of the expert, technique or tool that created the original alignment.

- $S' \subset S$ ($S' = S - \mu$, where μ is the set of magic mappings). In this case, it has not been possible to find some of the algorithms used by the expert, technique or tool to generate some specific mappings. The final set of mappings provided by the equivalent matching function is a subset of the original set. The rest of mappings are magic mappings. A large number of magic mappings means that either we have not used an appropriate set of matching algorithms or that the alignment was generated using a hitherto unknown technique.

Ontology	#Map.	Pr.	Rc.	F-M.
Russia12	85	0.97	0.04	0.08
RussiaAB	117	1.00	0.07	0.13
TourismAB	226	0.96	0.12	0.21
Sports	150	0.99	0.02	0.04
AnimalsAB	24	0.92	0.14	0.24

Table 1: Quality for the Equivalent Reverse Ontology Matching functions using Web Knowledge algorithms

Ontology	#Map.	Pr.	Rc.	F-M.
Russia12	85	0.86	0.53	0.65
RussiaAB	117	1.00	0.51	0.68
TourismAB	226	0.93	0.47	0.62
Sports	150	0.94	0.39	0.55
AnimalsAB	24	0.75	0.80	0.77

Table 2: Quality for the Equivalent Reverse Ontology Matching functions not using Web Knowledge algorithms

- $S' \supset S$ ($S' = S + \lambda$, where λ is a set of new discovered mappings). In this case, the expert, technique or tool used an ambiguous strategy to create the final alignment. If the result provided by the equivalent matching function is a superset from the original one, then, we know that a strategy was used only for an arbitrary set of entities. For example, (Mexico, Mexican) was included in the final alignment but (Canada, Canadian) was not. Our technique is able to capture the strategy, but it cannot be applied in the same arbitrary way.

5.1 Empirical Results

In our experiment (see Table 1 and 2), we have noticed that algorithms which use Web Knowledge (Google and Wikipedia distance in this case) have a big impact in our results. The reason is that such kind of algorithms, which detect the co-occurrence of terms in the same websites of the Web, are able to find a lot of correspondences, and therefore the precision is increased when using them. But these algorithms generate a lot of false positives too, so the recall is decreased.

We have that values for the precision are good. This means that algorithms that we have used are able to capture the most of the mappings from the alignment. On the other hand, the value for the recall is lower, what means that these algorithms find more mappings than the alignment had originally. Therefore, F-measure, thus, the overall quality measure decreases.

6. CONCLUSIONS

We have presented a novel approach for reverse ontology matching. To the best of our knowledge, this approach is the first attempt to extract the functions that were originally used to create an alignment between ontologies. As we have shown, it is very difficult to obtain the original function, but it is possible to compute an equivalent reverse ontology matching function for all ontology matching functions that have been created using one or several of the techniques studied, either they have been combined in a parallel or in a sequential way.

Results show us that we have reached a reasonable quality when capturing the equivalent reverse matching functions in the most of cases. Moreover, we have introduced the notion of magic mapping as a way to deal with those mappings which we do not know how they were found. The notion of magic mapping tells us that either we have not used an appropriate set of matching algorithms or we need to design a new matching algorithm that addresses this issue. However, in practice, web knowledge algorithms limit the presence of magic mappings and have a great impact on the results. The reason is that such algorithms are able to find a lot of correspondences (even those that are not very frequent), and therefore precision is increased. But these algorithms generate a lot of false positives too, so the recall is decreased. For this reason, we propose to use web knowledge algorithms only in domains where a great precision is required.

As future work, we have to face some important challenges. Firstly, it is necessary to improve the recall of the equivalent reverse matching functions. We think that this can be achieved either by the incorporation of more efficient matching algorithms either the design of a new composition model more effective than the current model. Secondly, it is necessary to research faster ways to reverse engineer an alignment. Checking one by one the mappings is a time consuming strategy, so it is necessary to research ways to accelerate the process without loss of quality. One possible way to do this could be working only on a sample of mappings, for example.

Acknowledgements

We wish to thank to the anonymous reviewers for the suggestions which have helped to improve this work. We thank to Anne Doherty for proofreading this document. This work has been funded by the Spanish Ministry of Innovation & Science through the project TIN2008-04844 and by the Dept. of Innovation & Science from the Regional Government of Andalucia through the project P07-TIC-02978.

7. REFERENCES

[1] Arenas, M, Perez, J., Riveros, C. (2008) The recovery of a schema mapping: bringing exchanged data back. *Proc. of PODS* 13–22.

[2] Arenas, M, Perez, J., Reutter, C. (2009) Inverting Schema Mappings: Bridging the Gap between Theory and Practice. *PVLDB 2(1)* 1018–1029.

[3] Choi, C., Song, I.,Han, H. (2006) A Survey on Ontology Mapping. *ACM Sigmod Record* **35(3)** 34–41.

[4] Cilibrasi, R., Vitanyi, P. (2007) The Google Similarity Distance. *IEEE Trans. Knowl. Data Eng.* **19(3)** 370–383.

[5] Ehrig, M. (2006) Ontology Alignment: Bridging the Semantic Gap. Springer-Verlag.

[6] Eckert, K., Meilicke, C., Stuckenschmidt, H. (2009) Improving Ontology Matching Using Meta-level Learning. *Proc. of European Semantic Web Conference ESWC 2009* 158–172.

[7] Euzenat, J., Shvaiko, P. (2007) Ontology Matching. Springer-Verlag.

[8] Fagin, R., Kolaitis, PG., Popa, L., Tan, WC. (2009) Reverse data exchange: coping with nulls. *Proc. of PODS* 23–32.

[9] Gruber, T. (1993) A translation approach to portable ontology specifications. *Knowledge Adquisition* **5(2)** 199–220.

[10] Hu, W., Qu, Y., Cheng, G. (2008) Matching large ontologies: A divide-and-conquer approach. *Data Knowl. Eng.* **67(1)** 140–160.

[11] Kalfoglou, Y., Schorlemmer, M. (2003) Ontology mapping: the state of the art. *Knowledge Eng. Review* **18(1)** 1–31.

[12] Li, J., Tang, J., Li, Y., Luo, Q. (2009) RiMOM: A Dynamic Multistrategy Ontology Alignment Framework. *IEEE Trans. Knowl. Data Eng.* **21(8)** 1218–1232.

[13] Martinez-Gil, J., Aldana-Montes, J. (2011) Evaluation of two heuristic approaches to solve the ontology meta-matching problem *Knowl. Inf. Syst.* **26(2)** 225–247.

[14] Noy, N. (2004) Semantic Integration: A Survey Of Ontology-Based Approaches. *ACM Sigmod Record* **33(4)** 65–70.

[15] Pirró, G., Talia, D. (2010) UFOme: An ontology mapping system with strategy prediction capabilities. *Data Knowl. Eng.* **69(5)** 444–471.

[16] Rahm, E., Bernstein, P.A. (2001) A survey of approaches to automatic schema matching. *VLDB J.* **10(4)** 334–350.