UNIVERSITY
OF TRENTO

DIPARTIMENTO DI INGEGNERIA E SCIENZA DELL'INFORMAZIONE

38050 Povo – Trento (Italy), Via Sommarive 14
http://www.disi.unitn.it

TEN CHALLENGES FOR ONTOLOGY MATCHING

Pavel Shvaiko and Jerome Euzenat

August 2008

Technical Report # DISI-08-042

Also: in Proceedings of The 7th International Conference on Ontologies, DataBases, and Applications of Semantics (ODBASE), 2008.

Ten Challenges for Ontology Matching

Pavel Shvaiko[1] and Jérôme Euzenat[2]

[1] TasLab, Informatica Trentina S.p.A., Trento, Italy
pavel.shvaiko@infotn.it
[2] INRIA & LIG, Grenoble, France
Jerome.Euzenat@inrialpes.fr

Abstract. This paper aims at analyzing the key trends and challenges of the ontology matching field. The main motivation behind this work is the fact that despite many component matching solutions that have been developed so far, there is no integrated solution that is a clear success, which is robust enough to be the basis for future development, and which is usable by non expert users. In this paper we first provide the basics of ontology matching with the help of examples. Then, we present general trends of the field and discuss ten challenges for ontology matching, thereby aiming to direct research into the critical path and to facilitate progress of the field.

1 Introduction

The progress of information and communication technologies has made available a huge amount of disparate information. The number of different information resources is growing significantly, and therefore, the problem of managing heterogeneity among them is increasing. As a consequence, various solutions have been proposed to facilitate dealing with this situation, and specifically, of automating integration of distributed information sources. Among these, semantic technologies have attracted significant attention. For example, according to Gartner[1], semantic technologies is in the list of top ten disruptive technologies for 2008-2012. In this paper we focus on a particular part of semantic technologies, which is ontology matching.

An ontology typically provides a vocabulary that describes a domain of interest and a specification of the meaning of terms used in the vocabulary. Depending on the precision of this specification, the notion of ontology encompasses several data and conceptual models, for example, sets of terms, classifications, database schemas, or fully axiomatized theories. However, when several competing ontologies are in use in different applications, most often they cannot interoperate as is, though the fact of using ontologies rises heterogeneity problems to a higher level.

Ontology matching is a solution to the semantic heterogeneity problem. It finds correspondences between semantically related entities of ontologies. These correspondences can be used for various tasks, such as ontology merging, query answering, data translation, etc. Thus, matching ontologies enables the knowledge and data expressed in the matched ontologies to interoperate [25].

[1] http://www.gartner.com/it/page.jsp?id=681107

Many diverse solutions of matching have been proposed so far, see [49, 67] for some contributions of the last decades and [14, 46, 64, 68, 73] for recent surveys[2]. Finally, ontology matching has been given a book account in [25]. However, despite the many component matching solutions that have been developed so far, there is no integrated solution that is a clear success, which is robust enough to be the basis for future development, and which is usable by non expert users.

This is a prospective paper and its key contribution is a discussion of the main trends in the ontology matching field articulated along ten challenges accompanied for each of these with an overview of the recent advances in the field. This should direct research into the critical path and accelerate progress of the ontology matching field. The challenges discussed are: (i) large-scale evaluation, (ii) performance of ontology-matching techniques, (iii) discovering missing background knowledge, (iv) uncertainty in ontology matching, (v) matcher selection and self-configuration, (vi) user involvement, (vii) explanation of matching results, $(viii)$ social and collaborative ontology matching, (ix) alignment management: infrastructure and support, and (x) reasoning with alignments.

The remainder of the paper is organized as follows. Section 2 provides, with the help of an example, the basics of ontology matching. Section 3 outlines ontology matching applications and discusses the role of final users in defining application requirements. Section 4 presents a market watch for the ontology matching field. Sections 5-14 discuss ten challenges of the field and for each of these briefly overview the corresponding recent advances. Finally, Section 15 reports the major findings of the paper.

2 The ontology matching problem

In this section we first discuss a motivating example (§2.1), then we provide some basic definitions of ontology matching (§2.2), and finally we describe the alignment life cycle (§2.3).

2.1 Motivating example

Let us use two simple XML schemas (see Figure 1), which can be viewed as a particular type of ontology, in order to exemplify the ontology matching problem.

Let us suppose that an e-commerce company needs to acquire another one. Technically, this acquisition may require the integration of the databases of these companies. The documents of both companies are stored according to XML schemas $O1$ and $O2$, respectively. A first step in integrating the schemas is to identify candidates to be merged or to have taxonomic relationships under an integrated schema. This step refers to a process of matching. For example, the elements with labels Price in $O1$ and in $O2$ are candidates to be merged, while the element with label Digital_Cameras in $O2$ should be subsumed by the element with label Photo_and_Cameras in $O1$. Once the correspondences between two schemas have been determined, the next step has to generate, for

[2] See `http://www.ontologymatching.org` for a complete information on the topic, e.g., publications, tutorials, relevant events.

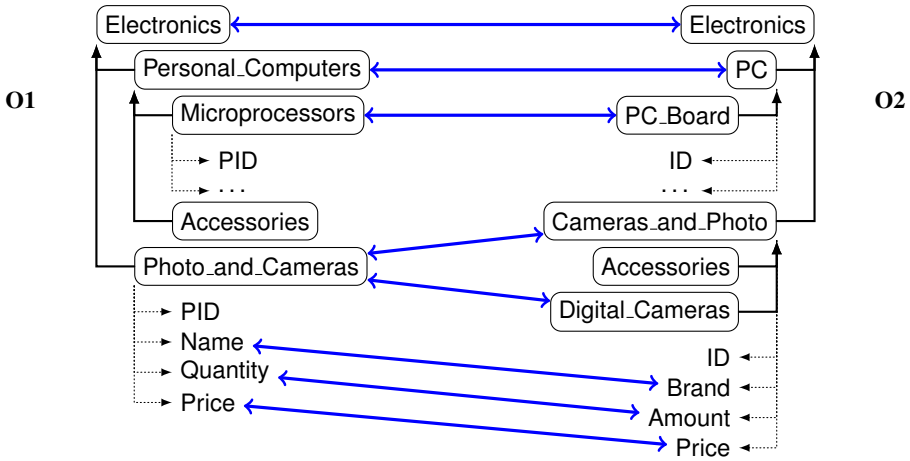

Fig. 1: Two simple XML schemas. XML elements are shown in rectangles with rounded corners, while attributes are shown without the latter. The correspondences are expressed by arrows.

example, query expressions that automatically translate data instances of these schemas under an integrated schema [73].

2.2 Problem statement

The *matching* operation determines the alignment A' for a pair of ontologies $O1$ and $O2$, each of which consisting of a set of discrete entities, such as classes, properties or individuals. There are some other parameters that can extend the definition of the matching process, namely: (i) the use of an input alignment A, which is to be completed by the process; (ii) the matching parameters, for instance, weights, thresholds; and (iii) external resources used by the matching process, for instance, common knowledge and domain specific thesauri.

Alignments express correspondences between entities belonging to different ontologies. Given two ontologies, a *correspondence* is a 5-uple: $\langle id, e_1, e_2, n, r\rangle$, where: id is a unique identifier of the given correspondence; e_1 and e_2 are entities (e.g., tables, XML elements, properties, classes) of the first and the second ontology, respectively; n is a confidence measure (typically in the $[0, 1]$ range) holding for the correspondence between e_1 and e_2; r is a relation (e.g., equivalence $(=)$, more general $(\sqsupseteq)$, disjointness $(\perp)$, overlapping $(\sqcap)$) holding between e_1 and e_2. The correspondence $\langle id, e_1, e_2, n, r\rangle$ asserts that the relation r holds between the ontology entities e_1 and e_2 with confidence n. The higher the confidence, the higher the likelihood that the relation holds.

For example, in Figure 1, according to some matching algorithm based on linguistic and structure analysis, the confidence measure (for the fact that the equivalence relation holds) between entities with labels Photo_and_Cameras in $O1$ and Cameras_and_Photo in $O2$ could be 0.67. Suppose that this matching algorithm uses a threshold of 0.55 for determining the resulting alignment, i.e., the algorithm considers all the pairs of entities with a confidence measure higher than 0.55 as correct correspondences. Thus, our hypothetical matching algorithm should return to the user the following correspondence: $\langle id_{3,3}, Photo\_and\_Cameras, Cameras\_and\_Photo, 0.67, =\rangle$. The relation between the same pair of entities, according to another matching algorithm which

is able to determine that both entities mean the same thing, could be exactly the equivalence relation (without computing the confidence measure). Thus, returning to the user $\langle id_{3,3}, Photo\_and\_Cameras, Cameras\_and\_Photo, n/a, = \rangle$.

2.3 Alignment life cycle

Like ontologies, alignments have their own life cycle [23] (see Figure 2). They are first created through a matching process, which may be manual. Then they can go through an iterative loop of evaluation and enhancement. Evaluation consists of assessing properties of the obtained alignment. It can be performed either manually or automatically. Enhancement can be obtained either through manual change of the alignment or application of refinement procedures, e.g., selecting some correspondences by applying thresholds. When an alignment is deemed worth publishing, then it can be stored and communicated to other parties interested in such an alignment. Finally, the alignment is transformed into another form or interpreted for performing actions, like mediation or merging.

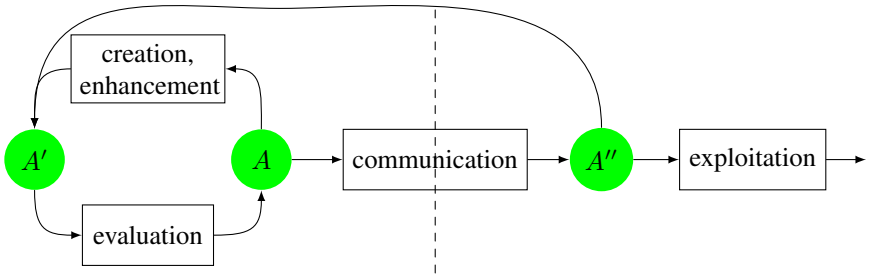

Fig. 2: The ontology alignment life cycle [23].

As Figure 2 indicates, creating an alignment is only the first step of the process. Very often these alignments have to be evaluated, improved and finally transformed into some executable procedure before being used by applications: transforming an ontology in order to integrate it with another one, generating a set of bridge axioms that will help the identification of corresponding concepts, translating messages sent from one agent to another, translating data circulating among heterogeneous web services, mediating queries and answers in peer-to-peer systems and federated databases.

3 Applications and use cases

Ontology matching is an important operation in traditional applications, such as ontology evolution, ontology integration, data integration, and data warehouses. Typically, these applications are characterized by heterogeneous structural models that are analyzed and matched either manually or semi-automatically at design time. In such applications, matching is a prerequisite to running the actual system.

There are some emerging applications that can be characterized by their dynamics, such as peer-to-peer information sharing, web service integration, multi-agent communication, query answering and semantic web browsing. Such applications, contrary to

traditional ones, require (ultimately) a run time matching operation and take advantage of more explicit conceptual models. A detailed description of these applications can be found in [25]. Let us now discuss the role of final users in defining application requirements.

User-oriented approach. Many research projects devoted to ontology matching correctly identify an application in which prototypes they develop can be eventually exploited. However, it is far rarely the case that *final users* are directly involved in the definition of requirements and use cases instantiating the applications under consideration within those projects. This is so because research projects are not usually concerned with bringing the original ideas developed within them down to the actual exploitation of these by the (expected) final users. Also enterprises that are often involved in larger research projects (e.g., of 4 years with about 1K man-month effort) are primarily interested in acquiring know-how to be later exploited in their internal projects. Hence, in order to foster an early practical exploitation of the research prototypes, it is necessary to directly involve final users in the research and development cycles. An example of such user-oriented open innovation methodologies includes Living Labs[3].

Below we exemplify a use case that has been elaborated together with final users, namely a public administration and more specifically, the Urban Planning and Environment Protection department of the Autonomous Province of Trento. Notice that involving final users into the research and development cycles requires addressing a *social challenge* of integrating relevant actors and facilitating the cross-fertilization among research centers, technology providers and user institutions, see [30] for a discussion of these in the context of the semantic heterogeneity problem. An example of undertaking this challenge includes Trentino as a Lab[4][31].

Emergency response. Within the OpenKnowledge[5] project there has been analyzed the organizational model of the distributed GIS agency infrastructure of Trentino that includes: civilian protection, urban planning, forestry, viability, etc. Each GIS agency is responsible for providing a subset of the geographic information for the local region. Let us focus on the most frequent use case, i.e., map request service, and in turn, on the most typical request, such as a digital map request. A service requestor - both in an emergency or normal situation - needs to visualize a map of a region with georeferenced information selected by a user. Therefore, the required map is a composition of different geographic layers offered by one of the service provider agents.

The OpenKnowledge project developed a peer-to-peer infrastructure which was used within the emergency response domain [56]. At the core of this approach is a specific view on semantics of both web service and agent coordination as proposed in [69]. Peers share explicit knowledge of the *interactions* in which they are engaged and these models of interaction are used operationally as the anchor for describing the semantics of the interaction. Instead of requiring a universal semantics across peers we require only that semantics is consistent (separately) for each instance of an interaction. These models of interactions are developed locally by peers and are shared on the network. Then, since there is no a priori semantic agreement (other than the interaction

[3] `http://www.cdt.ltu.se/projectweb/4421cddc626cb/Main.html`

[4] `http://www.taslab.eu`

[5] OpenKnowledge (FP6-027253): `http://www.openk.org`

model), matching is needed to automatically make semantic commitments between the interacting parts. In particular, it is used to identify peers, which are suitable to play a particular role in an interaction model.

In the context of formalization of a digital map request scenario mentioned above (see for details [56]), consider i-th interaction model IM_i, where a constraint on playing m-th role r_m in IM_i is as follows $C1$: getMap(MapFile, Version, Layers, Width, Height, Format, XMinBB, YMinBB, XMaxBB, YMaxBB), which can be viewed as a web service description. In turn, the getMap message will contain the URL of the requested map (MapFile), the version of the service (Version), the requested geographic layers (Layers), the dimensions of the map (Width, Heigth), its graphic format (Format), and finally its spatial coverage (XMinBB, YMinBB, XmaxBB, YMaxBB). Let us suppose that $C2$: getMap(Dimension(Width, Height), MapFile, Edition, Layers) is a description of the capabilities of k-th peer, p_k. Then, p_k wants to subscribe to r_m in IM_i, and thus, its capabilities should be matched to the constraints of r_m. If the matching between $C1$ and $C2$ is good enough, then, peer p_k can be allowed to play role r_m. Notice that matching between constraints of a role in an interaction model and peer capabilities should be performed at run time. A matching solution for this use case has been developed in [33].

4 Market watch

Let us make several observations concerning the development of the ontology matching field as such. With this respect, an important work has been conducted within the Knowledge Web project[6]. It concerned with the analysis of the Gartner hype curve[7] and placement of the various semantic web technologies along it, see Figure 3. In order to build this curve various distinct groups of researchers and practitioners have been involved, see [10] for details. On the one side, the topics addressed in Figure 3 are specific to the semantic web domain. These cannot be directly compared with any Gartner's counterpart, and, hence, the latter are not taken into account. On the other side, topics of Figure 3 include ontology matching, referred to as alignment.

The first observation is that for what concerns ontology matching, both researchers and practitioners agree on locating this topic just before the peak of inflated expectation, with the same long term duration (5 to 10 years) to mainstream adoption. Hence, there are still many challenges to be addressed before ontology matching technology can be seen among the mainstream components.

Let us now consider dynamics of papers devoted to ontology matching and published in the major conferences and journals[8], which is as follows (year:number of publications): $\leq$2000:18, 2001:15, 2002:13, 2003:17, 2004:29, 2005:54, 2006:60, and 2007:71. Another observation is that the dynamics of papers devoted to ontology matching reconfirm the overall trend indicated in Figure 3 that the ontology matching field keeps growing.

[6] KnowledgeWeb (IST-2004-507482): `http://knowledgeweb.semanticweb.org/`

[7] `http://www.gartner.com/pages/story.php.id.8795.s.8.jsp`

[8] `http://www.ontologymatching.org/publications`. Access date: 18.08.2008.

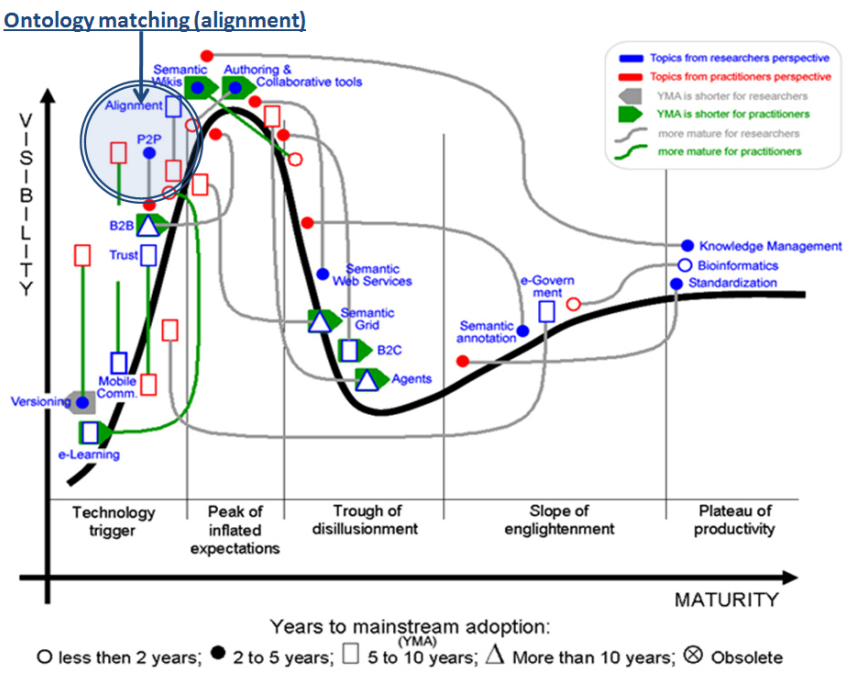

Fig. 3: Hype curve: comparison between researchers' and practitioners' points of view on semantic web technologies. Adapted from [10].

Based on the analysis above, we expect that, as the ontology matching technology is becoming more mature, practitioners will increase their expectations and will want to experiment with it more intensively.

In Sections 5-14 we discuss ten challenges for ontology matching together with a brief overview of the recent advances in the field for each of these challenges.

5 Large-scale evaluation

The rapid growth of various matching approaches makes the issues of their evaluation and comparison more severe. In order to address these issues, in 2005 the Ontology Alignment Evaluation Initiative - OAEI[9] was set up, which is a coordinated international initiative that organizes the evaluation of the increasing number of ontology matching systems. The main goal of OAEI is to support the comparison of the systems and algorithms on the same basis and to allow anyone to draw conclusions about the best matching strategies. Two first events were organized in 2004 [76]. Then, unique OAEI campaigns occurred in 2005 [2], 2006 [24], 2007 [22] and at the moment of writing of this paper OAEI-2008 is under way.

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

There are many issues to be addressed in ontology matching evaluation in order to empirically prove the matching technology to be mature and reliable.

- OAEI campaigns gave only some preliminary evidence of the scalability characteristics of the ontology matching technology. Therefore, larger tests involving 10.000, 100.000, and 1.000.000 entities per ontology (e.g., UMLS[10] has about 200.000 entities) are to be designed and conducted. In turn, this raises the issues of a wider automation for acquisition of reference alignments, e.g., by minimizing the human effort while increasing an evaluation dataset size.
- There is a need for more accurate evaluation quality measures (initial steps towards these have already been done in [21]). In particular, application specific measures are needed in order to assess whether the result of matching is good enough for an application.
- There is a need for evaluation methods grounded on a deep analysis of the matching problem space in order to offer semi-automatic test generation methods of desired test hardness by addressing a particular point of this space (initial steps towards this line have already been done in [39]).
- Despite efforts on meta-matching systems, composing matchers [18, 48, 50] and on Alignment API [19], ontology matching largely lacks interoperability benchmarks between tools.

6 Performance of ontology-matching techniques

Beside quality of matching results, there is an issue of performance, see, e.g., [6]. Performance is of prime importance in many dynamic applications, for example, where a user can not wait too long for the system to respond. Execution time indicator shows scalability properties of the matchers and their potential to become industrial-strength systems. Also, referring to [41], the fact that some systems run out of memory on some test cases, although being fast on the other test cases, suggests that their performance time is achieved by using a large amount of main memory. Therefore, usage of main memory should also be taken into account.

Optimizations are worth been done only once the underlying basic techniques are stable. For example, in the case of S-Match [35, 38, 41], when dealing with lightweight ontologies [32, 42], the matching problem was reduced to the validity problem for the propositional calculus. The basic version of S-Match was using a standard DPLL-based satisfiability procedure of SAT4J[11]. Once it has been realized that the approach is promising (based on the preliminary evaluation in [34]), the efficiency problems have been tackled. Specifically, for certain and quite frequent in practise cases, e.g., when matching formula is Horn, satisfiability became resolved in linear time, while standard SAT solver would require quadratic time, see [40] for details. Beside S-Match, several other groups, for example, Falcon [44] and COMA++ [13], have started addressing seriously the issues of performance. However, this fact cannot be still considered as a trend in the field, see, e.g., the results of the anatomy track of OAEI-2007 [22], where only

[10] http://www.nlm.nih.gov/research/umls/
[11] http://www.sat4j.org/

several systems, such as Falcon, took several minutes to complete this matching task, while other systems took much more time (hours and even days).

7 Discovering missing background knowledge

One of the sources of difficulty for the matching tasks is that ontologies are designed with certain background knowledge and in a certain context, which unfortunately do not become part of the ontology specification, and, thus, are not available to matchers. Hence, the lack of background knowledge increases the difficulty of the matching task, e.g., by generating too many ambiguities. Various strategies have been used to attack the problem of the lack of background knowledge. These include: (i) declaring the missing axioms manually as a pre-match effort [12, 54]; (ii) reusing previous match results [12]; (iii) querying the web [43]; (iv) using domain specific corpus [1, 52]; (v) using domain specific ontologies [1, 77]; and (vi) using ontologies available on the semantic web [71]. In addition, the work in [36] discussed an automatic approach to deal with the lack of background knowledge in matching tasks by using semantic matching [35, 37] iteratively. While the work in [9] proposed to automatically revise a mediated schema (which can be viewed as a background knowledge in data integration applications) in order to improve matchability.

The techniques mentioned above have helped improving the results of matchers in various cases. Moreover, these techniques can undergo different variations based on the way the background knowledge sources are selected, the way the ontology entities are matched against the background knowledge sources and the combination of the results obtained from the various external sources; though they still have to be systematically investigated, combined in a complementary fashion and improved.

Finally, it is worth noting that discovering missing background knowledge is particulary important in dynamic settings, where the matching input is often much more shallow (especially when dealing with fragmented descriptions), and therefore, incorporates fewer clues. To this end, it is vital to identify the minimal background knowledge necessary to resolve a particular problem with good enough results [74] and how to compute this minimal background knowledge.

8 Uncertainty in ontology matching

The issue of dealing with uncertainty in ontology matching has been addressed in [8, 16, 28, 29, 53, 63]. A way of modeling ontology matching as an uncertain process is by using similarity matrices as a measure of certainty. A matcher then is measured by the fit of its estimation of a certainty of a correspondence to the real world. In [29], such a formal framework was provided, attempting to answer the question of whether there are good and bad matchers. Uncertainty can also be reduced iteratively. In such a setting, initial assumptions are strengthened or discarded, thereby refining the initial measures of imperfection. In [28], uncertainty is refined by a comparison of K alignments, each with its own uncertainty measure (modeled as a fuzzy relation over the two ontologies) in order to improve precision of the matching results. Finally, the work in [16] introduced the notion of probabilistic schema mappings (correspondences), namely a set of

mappings with a probability attached to each mapping; and, used it to answer queries with uncertainty about semi-automatically created mappings. Imprecise mappings can be further improved over time as deemed necessary, for example, within the settings of approximate data integration, see, e.g., [72].

Beside the work done along this line, there is still a need to understand better the foundations of modeling uncertainty in ontology matching in order to improve detection of mappings causing inconsistencies, e.g., via probabilistic reasoning, or to identify where the user feedback is maximally useful. In the dynamic applications it often occurs that there is no precise correspondence or a correspondence identified is not specific enough, hence, there is a need to choose a good enough one (with respect to application needs). In turn, this requires formalizing a link between ontology matching tools and information integration systems that support uncertainty.

9 Matcher selection and self-configuration

There are many matchers that are available nowadays. Often these perform well in some cases and not so well in some other cases. This makes the issues of (i) matcher selection, (ii) matcher combination and (iii) matcher tuning of prime importance.

Matcher selection. The work on evaluation (§5) can be used in order to assess the strengths and the weaknesses of individual matchers by comparing their results with task requirements. Often, there are many different constraints and requirements brought by the matching tasks, e.g., correctness, completeness, execution time, main memory, thereby involving multi-decision criteria. This problem has been addressed so far through, e.g., analytic hierarchy process [62] and ad hoc rules [45].

Matcher combination. Beside matcher selection, another issue is the combination of individual matchers and libraries of matchers. This increases the complexity of the previous problem by allowing to put several matchers together and to combine them adequately. So far, only design time toolboxes allow to do this manually [13]. Another approach involves ontology meta-matching [50], i.e., a framework for combining a set of selected ontology matchers. The main issue here is the semi-automatic combination of matchers by looking for complementarities, balancing the weaknesses and reinforcing the strengths of the components.

Matcher tuning. In dynamic settings, such as the web, it is natural that applications are constantly changing their characteristics. Therefore, approaches that attempt to tune and adapt automatically matching solutions to the settings in which an application operates are of high importance. This may involve the run time reconfiguration of a matcher by finding its most appropriate parameters, such as thresholds, weights, and coefficients. The work in [50] proposed an approach to tune a library of schema matchers at design time; while the work in [15] discussed consensus building after many methods have been used. The challenge, however, is to be able to perform matcher self-tuning at run time, and therefore, efficiency of the matcher configuration search strategies becomes crucial.

The above mentioned problems share common characteristics: the search space is very large and the decision is made involving multiple criteria. Notice that resolving these simultaneously at run time makes the problem even harder.

10 User involvement

In traditional applications automatic ontology matching usually cannot deliver high quality results, especially on large datasets, see, e.g., [39]. Thus, for traditional applications, semi-automatic matching is a way to improve the effectiveness of the results. So far, there have only been few studies on how to involve users in ontology matching. Most of these efforts have been dedicated to design-time matcher interaction [13, 66].

Some recent work, however, has focussed on the ergonomic aspect of elaborating alignments, either for designing them manually or for checking and correcting them. The work in [27] proposed a graphical visualization of alignments based on cognitive studies. In turn, the work in [60, 61] has provided an environment for manually designing complex alignments through the use of connected perspective that allows to quickly deemphasize non relevant aspects of the ontologies being matched while keeping the connections between relevant entities. This line of work must be still consolidated and it should be possible to seamlessly plug the results obtained here into an alignment management system (see §13). With the development of interactive approaches the issues of their usability will become more severe. This includes scalability of visualization [70] and better user interfaces in general, which are expected to bring big productivity gains; as from [5] even bigger than from more accurate matching algorithms.

There remains an interesting path to follow concerning user involvement: relying on the application users in order to learn from them what is useful in the alignments under consideration. This can be exploited either at the matcher level by adjusting its parameters and providing new (partial) input alignments, or at the alignment level by experimenting with confidence weights to improve the results given to the users. Another promising direction in this respect is what we call "implicit matching", i.e., by serendipitously contributing to improve available alignments. For instance, in a semantic peer-to-peer system, if a user poses a query and there is no alignment in the system leading to an answer, this user may be willing to help the system by providing several correspondences that are necessary for answering the query. These correspondences can be collected by the system and, over time, the system will acquire enough knowledge about the useful correspondences. The example discussed can be also viewed as a part of typical interactions in a collaborative environment (see §12). The issue here is, both for design time and run time matching, to design interaction schemes which are burdenless to the user. At design time, interaction should be both natural and complete; at run time, it should be hidden in the user task.

Finally, let us note that dynamic applications have a specific feature that traditional applications have not: since there are multiple parties (agents) involved in the process, mismatches (mistakes) could be negotiated (corrected) in a fully automated way. This has already been considered in the field of multi-agent systems where raw alignments are refined by agent negotiation [17, 47]. Therefore, explanations of matching (see §11), being an argumentation schema, become crucial.

11 Explanation of matching results

In order for matching systems to gain a wider acceptance, it will be necessary that they can provide arguments for their results to users or to other programs that use them. In fact, alignments produced by matching systems may not be intuitively obvious to human users, and therefore, they need to be explained. Having understood the alignments returned by a matching system, users can deliberately edit them manually, thereby providing the feedback to the system, see [11, 47, 75] for the solutions proposed so far and [25] for their in-depth analysis.

A more recent work introduced the notion of a matchability score (computed via a synthetic workload), which quantifies how well on average a given schema matches future schemas [9]. Using the matchability score, different types of matching mistakes can be analyzed. Based on them a matchability report is generated, thereby guiding users in revising the correspondences by addressing the reported mistakes together with the suggested revisions to be made.

Generally, the key issue here is to represent explanations in a simple and clear way to the user in order to facilitate informed decision making. In a longer term, it would be useful to standardize explanations/proofs of matching results in order to facilitate the interaction of matching systems with other programs.

12 Social and collaborative ontology matching

Another way to tackle the matching task is to take advantage of the network effect: if it is too cumbersome for one person to come up with a correct alignment between several pairs of ontologies, this can be more easily solved by many people together. This comes from three aspects: (i) each person has to do a very small amount of work, (ii) each person can improve on what has been done by others, and (iii) errors remain in minority.

The work in [78] reported on early experiments with community-driven ontology matching in which a community of people can share alignments and argue about them by using annotations. Later the work in [65] proposed a collaborative system in the area of bio-informatics for sharing both ontologies and mappings (i.e., correspondences). It allows users to share, edit and rate these mappings. The strengths of this system are a user friendly interface with the possibility to annotate alignments and the direct connection with the ontologies which helps users to navigate. In turn, [57] proposed to enlist the multitude of users in a community to help match schemas in a Web 2.0 fashion by asking users simple questions and then learn from the answers to improve matching accuracy. Finally, the work on alignment server in [20] supported alignment storing, correspondence annotation and sharing, though it was more closely designed as a middleware component rather than a collaborative tool.

Collaborative and social approaches to ontology matching rely on infrastructures allowing for sharing alignments and annotating them in a rich way. These features can be used to facilitate alignment reuse. The current challenge in collaborative ontology matching is thus to find the right annotation support and the adequate description units to make it work at a large scale. In particular, contradictory and incomplete alignments

should be dealt with in a satisfactory way. Other issues include understanding how to deal with malicious users, and what would be the promising incentive schemas to facilitate user participation.

13 Alignment management: infrastructure and support

Alignments, like ontologies, must be supported during their life cycle phases by adequate tools and standards. These required functions can be implemented as services, the most notable of which are: (i) *match two ontologies* possibly by selecting an algorithm to be used and its parameters (including an initial alignment, see §2.2); (ii) *store an alignment* in a persistent storage; (iii) *retrieve an alignment* based on its identifier; (iv) *retrieve alignment metadata* such that its identifier can be used to choose between specific alignments; (v) *find (stored) alignments* between two specific ontologies; (vi) *edit an alignment* by adding or discarding correspondences (this is typically the result of a graphic editing session); (vii) *trim alignments* based on a threshold; $(viii)$ *generate code* implementing ontology transformations, data translations or bridge axioms based on a particular alignment; and (ix) *translate a message* with regard to an alignment.

This functional support must be complemented with rich metadata allowing users and systems to select the adequate alignments based on various criteria. It should also support permanent storage and identification of alignments in order to reliably use the existing alignments. In databases, several systems have been designed for offering a variety of matching methods and a library of mappings [4]. However, these were meant only as a component for design time integration and not as a service that can be used at run time. In turn, the alignment server [20] has been designed with this goal in mind. Notice that in the context of collaborative matching (§12) the above mentioned needs are vital.

We can distinguish two levels in alignment management: (i) the infrastructure middleware and (ii) the support environments that provide task related access to alignments. The support environments may be dedicated to alignment edition [60, 66], alignment processing, alignment sharing and discussing [65], or model management [59]. These two levels may be mixed in a single system [65] or kept clearly separated [20].

One of the challenges here is to provide an alignment support infrastructure at the web scale, such that tools and, more importantly, applications can rely on it in order to share, i.e., publish and reuse, alignments.

Moreover, the alignment life cycle (§2.3) is tightly related to the ontology life cycle: as soon as ontologies evolve, new alignments have to be produced following the ontology evolution. This can be achieved by recording the changes made to ontologies and transforming those changes into an alignment (from one ontology version to the next one). This can be used for computing new alignments that will update the previous ones. In this case, previously existing alignments can be replaced by their composition with the ontology update alignment (see Figure 4). As demonstrated by this evolution example, alignment management can rely on composition of alignments which, in turn, requires to reason about alignments (see §14).

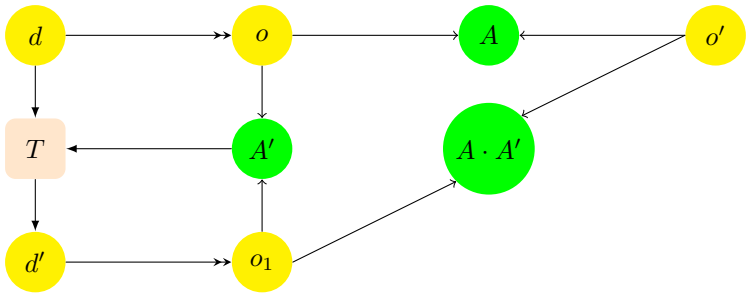

Fig. 4: Evolution of alignments [23]. When an ontology o evolves into a new version o_1, it is necessary to update the instances of this ontology (d) and the alignment(s) (A) it has with other ontologies (o'). To this extent, a new alignment (A') between the two versions can be established and used for generating the necessary instance transformation (T) and for linking ($A \cdot A'$) the ontologies o_1 and o'.

14 Reasoning with alignments

The ultimate goal of matching ontologies is to use alignments. For this purpose, they should be attributed a semantics. There have been developed various kinds of semantics [7, 51, 80] that allow to define the consequences of the aligned ontologies or distributed systems, i.e., several ontologies and several alignments.

At the level of alignments, an important question is what correspondences are the consequences of the aligned ontologies or distributed systems (α-consequences). This is important because it allows systems using alignments to take advantage of these correspondences, e.g., for transforming ontologies or translating messages. Computing α-consequences is used for finding missing alignments between two ontologies or strengthening the existing alignments. This is useful for: (i) deducing alignments; (ii) evolving alignments (see Figure 4); (iii) checking alignment consistency and repairing alignments [58]; and (iv) evaluating alignments [21].

A weaker level of reasoning that can be implemented is an alignment composition. It consists of deducing correspondences holding between two ontologies from alignments involving other ontologies. We can distinguish between two kinds of alignment composition: full alignment composition and ontology-free alignment composition. The latter composes alignments without any access to ontologies. Hence, it cannot, in general find all consequences of ontologies, but only the so-called quasi-consequences [79]. All these kinds of reasoning are correct but not semantically complete, i.e., they will not find all α-consequences of a set of alignments. This can however be useful because they may be faster to obtain.

In database schema matching, the notion of mapping composition is prominent and has been thoroughly investigated [3, 55]. The problem here is to design a composition operator that guarantees that the successive applications of two mappings yields the same results as the application of their composition [26]. Similar studies should be performed in the context of ontology alignments with various ontology and alignment languages.

15 Conclusions

We discussed ten challenges for ontology matching, accompanied for each of these with an overview of the recent advances in the field. We believe that challenges outlined are on the critical path, hence, addressing them should accelerate progress of ontology matching. Moreover, these challenges are not isolated from each others: collaborative matching requires an alignment infrastructure; alignment evolution and other operations of alignment management require reasoning with alignments; user involvement would benefit from and contribute to collaborative matching; etc. Hence, these challenges, even if clearly identified will certainly have to be considered in prospective relation with each other.

Beside the mentioned challenges, much more work is needed in order to bring the matching technology to the plateau of productivity. This includes dealing with multilinguism, spatial matching for GIS applications, etc.

Acknowledgements: The first author appreciates support from the Trentino as a Lab (TasLab) project of the European Network of the Living Labs. The second author has been partially supported by the European integrated project NeOn (IST-2005-027595) and the RNTL project WebContent. We are thankful to the TasLab group members: Isabella Bressan, Ivan Pilati, Valentina Ferrari, Luca Mion and Marco Combetto for many fruitful discussions on the living labs methodology for innovation. We are grateful to Fausto Giunchiglia, Maurizio Marchese, Mikalai Yatskevich, Roberta Cuel (University of Trento), Lorenzo Vaccari (Urban Planning and Environment Protection department of the Autonomous Province of Trento), Marta Sabou (Open University), Antoine Zimmermann (INRIA), Zharko Aleksovski (Vrije Universiteit Amsterdam), and Malgorzata Mochol (Free University of Berlin) for the insightful comments on various aspects of ontology matching covered in this paper.

References

1. Z. Aleksovski. *Using background knowledge in ontology matching*. PhD thesis, Vrije Universiteit Amsterdam, 2008.
2. B. Ashpole, M. Ehrig, J. Euzenat, and H. Stuckenschmidt, editors. *Proceedings of the workshop on Integrating Ontologies at K-CAP*, 2005.
3. P. Bernstein, T. Green, S. Melnik, and A. Nash. Implementing mapping composition. *The VLDB Journal*, 2008.
4. P. Bernstein, A. Halevy, and R. Pottinger. A vision of management of complex models. *ACM SIGMOD Record*, 2000.
5. P. Bernstein and S. Melnik. Model management 2.0: manipulating richer mappings. In *Proceedings of SIGMOD*, 2007.
6. P. Bernstein, S. Melnik, M. Petropoulos, and C. Quix. Industrial-strength schema matching. *ACM SIGMOD Record*, 2004.
7. A. Borgida and L. Serafini. Distributed description logics: Assimilating information from peer sources. *Journal on Data Semantics*, 2003.
8. S. Castano, A. Ferrara, D. Lorusso, T. Näth, and R. Möller. Mapping validation by probabilistic reasoning. In *Proceedings of ESWC*, 2008.
9. X. Chai, M. Sayyadian, A. Doan, A. Rosenthal, and L. Seligman. Analyzing and revising mediated schemas to improve their matchability. In *Proceedings of VLDB*, 2008.

10. R. Cuel, A. Delteil, V. Louis, and C. Rizzi. *Knowledge Web white paper: The Technology Roadmap of the Semantic Web*. `http://knowledgeweb.semanticweb.org/o2i/menu/KWTR-whitepaper-43-final.pdf`, 2007.
11. R. Dhamankar, Y. Lee, A. Doan, A. Halevy, and P. Domingos. iMAP: Discovering complex semantic matches between database schemas. In *Proceedings of SIGMOD*, 2004.
12. H. Do and E. Rahm. COMA – a system for flexible combination of schema matching approaches. In *Proceedings of VLDB*, 2002.
13. H. Do and E. Rahm. Matching large schemas: Approaches and evaluation. *Information Systems*, 2007.
14. A. Doan and A. Halevy. Semantic integration research in the database community: A brief survey. *AI Magazine*, 2005. Special issue on Semantic integration.
15. C. Domshlak, A. Gal, and H. Roitman. Rank aggregation for automatic schema matching. *IEEE Transactions on Knowledge and Data Engineering*, 2007.
16. X. Dong, A. Halevy, and C. Yu. Data integration with uncertainty. In *Proceedings of VLDB*, 2007.
17. C. dos Santos, M. Moraes, P. Quaresma, and R. Vieira. A cooperative approach for composite ontology mapping. *Journal on Data Semantics*, 2008.
18. M. Ehrig, S. Staab, and Y. Sure. Bootstrapping ontology alignment methods with APFEL. In *Proceedings of ISWC*, 2005.
19. J. Euzenat. An API for ontology alignment. In *Proceedings of ISWC*, 2004.
20. J. Euzenat. Alignment infrastructure for ontology mediation and other applications. In *Proceedings of the workshop on Mediation in Semantic Web Services*, 2005.
21. J. Euzenat. Semantic precision and recall for ontology alignment evaluation. In *Proceedings of IJCAI*, 2007.
22. J. Euzenat, A. Isaac, C. Meilicke, P. Shvaiko, H. Stuckenschmidt, O. Šváb, V. Svátek, W. van Hage, and M. Yatskevich. Results of the ontology alignment evaluation initiative 2007. In *Proceedings of the workshop on Ontology Matching at ISWC/ASWC*, 2007.
23. J. Euzenat, A. Mocan, and F. Scharffe. Ontology alignments: an ontology management perspective. In *Ontology management: semantic web, semantic web services, and business applications*. Springer, 2008.
24. J. Euzenat, M. Mochol, P. Shvaiko, H. Stuckenschmidt, O. Svab, V. Svatek, W. van Hage, and M. Yatskevich. Results of the ontology alignment evaluation initiative 2006. In *Proceedings of the workshop on Ontology Matching at ISWC*, 2006.
25. J. Euzenat and P. Shvaiko. *Ontology matching*. Springer, 2007.
26. R. Fagin, P. Kolaitis, L. Popa, and W. Tan. Composing schema mappings: Second-order dependencies to the rescue. *ACM Transactions on Database Systems*, 2005.
27. S. Falconer and M. Storey. A cognitive support framework for ontology mapping. In *Proceedings of ISWC/ASWC*, 2007.
28. A. Gal. Managing uncertainty in schema matching with top-k schema mappings. *Journal on Data Semantics*, 2006.
29. A. Gal, A. Anaby-Tavor, A. Trombetta, and D. Montesi. A framework for modeling and evaluating automatic semantic reconciliation. *The VLDB Journal*, 2005.
30. F. Giunchiglia. Managing diversity in knowledge. *Keynote talk at ECAI*, 2006.
31. F. Giunchiglia. Il ruolo degli enti di ricerca per lo sviluppo dell'ICT del Trentino (English translation: The role of the research centers in the development of Trentino). In *Le Tecnologie Digitali nell'economia del Trentino*, 2008.
32. F. Giunchiglia, M. Marchese, and I. Zaihrayeu. Encoding classifications into lightweight ontologies. *Journal of Data Semantics*, 2007.
33. F. Giunchiglia, F. McNeill, M. Yatskevich, J. Pane, P. Besana, and P. Shvaiko. Approximate structure preserving semantic matching. In *Proceedings of ODBASE*, 2008.

34. F. Giunchiglia, P. Shvaiko, and M. Yatskevich. S-Match: an algorithm and an implementation of semantic matching. In *Proceedings of ESWS*, 2004.
35. F. Giunchiglia, P. Shvaiko, and M. Yatskevich. Semantic schema matching. In *Proceedings of CoopIS*, 2005.
36. F. Giunchiglia, P. Shvaiko, and M. Yatskevich. Discovering missing background knowledge in ontology matching. In *Proceedings of ECAI*, 2006.
37. F. Giunchiglia, P. Shvaiko, and M. Yatskevich. Semantic matching. *Encyclopedia of Database Systems*, 2009, to appear.
38. F. Giunchiglia and M. Yatskevich. Element level semantic matching. In *Proceedings of the workshop on Meaning Coordination and Negotiation at ISWC*, 2004.
39. F. Giunchiglia, M. Yatskevich, P. Avesani, and P. Shvaiko. A large scale dataset for the evaluation of ontology matching systems. *The Knowledge Engineering Review*, 2008, to appear.
40. F. Giunchiglia, M. Yatskevich, and E. Giunchiglia. Efficient semantic matching. In *Proceedings of ESWC*, 2005.
41. F. Giunchiglia, M. Yatskevich, and P. Shvaiko. Semantic matching: Algorithms and implementation. *Journal on Data Semantics*, 2007.
42. F. Giunchiglia and I. Zaihrayeu. Lightweight ontologies. *Encyclopedia of Database Systems*, 2009, to appear.
43. R. Gligorov, Z. Aleksovski, W. ten Kate, and F. van Harmelen. Using google distance to weight approximate ontology matches. In *Proceedings of WWW*, 2007.
44. W. Hu, Y. Qu, and G. Cheng. Matching large ontologies: A divide-and-conquer approach. *Data and Knowledge Engineering*, 2008, to appear.
45. M. Huza, M. Harzallah, and F. Trichet. OntoMas: a tutoring system dedicated to ontology matching. In *Proceedings of the workshop on Ontology Matching*, 2006.
46. Y. Kalfoglou and M. Schorlemmer. Ontology mapping: the state of the art. *The Knowledge Engineering Review*, 2003.
47. L. Laera, I. Blacoe, V. Tamma, T. Payne, J. Euzenat, and T. Bench-Capon. Argumentation over ontology correspondences in MAS. In *Proceedings of AAMAS*, 2007.
48. P. Lambrix and H. Tan. A tool for evaluating ontology alignment strategies. *Journal on Data Semantics*, 2007.
49. J. Larson, S. Navathe, and R. Elmasri. A theory of attributed equivalence in databases with application to schema integration. *IEEE Transactions on Software Engineering*, 1989.
50. Y. Lee, M. Sayyadian, A. Doan, and A. Rosenthal. eTuner: tuning schema matching software using synthetic scenarios. *The VLDB Journal*, 2007.
51. M. Lenzerini. Data integration: A theoretical perspective. In *Proceedings of PODS*, 2002.
52. J. Madhavan, P. Bernstein, A. Doan, and A. Halevy. Corpus-based schema matching. In *Proceedings of ICDE*, 2005.
53. J. Madhavan, P. Bernstein, P. Domingos, and A. Halevy. Representing and reasoning about mappings between domain models. In *Proceedings of AAAI*, 2002.
54. J. Madhavan, P. Bernstein, and E. Rahm. Generic schema matching with Cupid. In *Proceedings of VLDB*, 2001.
55. J. Madhavan and A. Halevy. Composing mappings among data sources. In *Proceedings of VLDB*, 2003.
56. M. Marchese, L. Vaccari, P. Shvaiko, and J. Pane. An application of approximate ontology matching in eResponse. In *Proceedings of ISCRAM*, 2008.
57. R. McCann, W. Shen, and A. Doan. Matching schemas in online communities: A web 2.0 approach. In *Proceedings of ICDE*, 2008.
58. C. Meilicke, H. Stuckenschmidt, and A. Tamilin. Repairing ontology mappings. In *Proceedings of AAAI*, 2007.

59. S. Melnik, E. Rahm, and P. Bernstein. Developing metadata-intensive applications with Rondo. *Journal of Web Semantics*, 2003.
60. A. Mocan. *Ontology-based data mediation for semantic environments*. PhD thesis, National University Ireland Galway, 2008.
61. A. Mocan, E. Cimpian, and M. Kerrigan. Formal model for ontology mapping creation. In *Proceedings of ISWC*, 2006.
62. M. Mochol, A. Jentzsch, and J. Euzenat. Applying an analytic method for matching approach selection. In *Proceedings of the workshop on Ontology Matching*, 2006.
63. H. Nottelmann and U. Straccia. Information retrieval and machine learning for probabilistic schema matching. *Information Processing and Management*, 2007.
64. N. Noy. Semantic integration: A survey of ontology-based approaches. *ACM SIGMOD Record*, 2004.
65. N. Noy, N. Griffith, and M. Musen. Collecting community-based mappings in an ontology repository. In *Proceedings of ISWC*, 2008.
66. N. Noy and M. Musen. The PROMPT suite: interactive tools for ontology merging and mapping. *International Journal of Human-Computer Studies*, 2003.
67. C. Parent and S. Spaccapietra. Issues and approaches of database integration. *Communications of the ACM*, 1998.
68. E. Rahm and P. Bernstein. A survey of approaches to automatic schema matching. *The VLDB Journal*, 2001.
69. D. Robertson. A lightweight coordination calculus for agent systems. In *Declarative Agent Languages and Technologies*, 2004.
70. G. Robertson, M. Czerwinski, and J. Churchill. Visualization of mappings between schemas. In *Proceedings of CHI*, 2005.
71. M. Sabou, M. d'Aquin, and E. Motta. Exploring the semantic web as background knowledge for ontology matching. *Journal on Data Semantics*, 2008, to appear.
72. A. Sarma, X. Dong, and A. Halevy. Bootstrapping pay-as-you-go data integration systems. In *Proceedings of SIGMOD*, 2008.
73. P. Shvaiko and J. Euzenat. A survey of schema-based matching approaches. *Journal on Data Semantics*, 2005.
74. P. Shvaiko, F. Giunchiglia, A. Bundy, P. Besana, C. Sierra, F. van Harmelen, and I. Zaihrayeu. *OpenKnowledge Deliverable 4.2:Benchmarking methodology for good enough answers*. `http://www.cisa.informatics.ed.ac.uk/OK/Deliverables/D4.2.pdf`, 2008.
75. P. Shvaiko, F. Giunchiglia, P. Pinheiro da Silva, and D. McGuinness. Web explanations for semantic heterogeneity discovery. In *Proceedings of ESWC*, 2005.
76. Y. Sure, O. Corcho, J. Euzenat, and T. Hughes, editors. *Proceedings of the workshop on Evaluation of Ontology-based tools*, 2004.
77. S. Zhang and O. Bodenreider. Experience in aligning anatomical ontologies. *International Journal on Semantic Web and Information Systems*, 2007.
78. A. Zhdanova and P. Shvaiko. Community-driven ontology matching. In *Proceedings of ESWC*, 2006.
79. A. Zimmermann. *Sémantique des connaissances distribuées*. PhD thesis, Université Joseph-Fourier, Grenoble (FR), 2008.
80. A. Zimmermann and J. Euzenat. Three semantics for distributed systems and their relations with alignment composition. In *Proceedings of ISWC*, 2006.