PowerMap: Mapping the Real Semantic Web on the Fly

Vanessa Lopez[1], Marta Sabou[1], Enrico Motta[1]

[1] Knowledge Media Institute (KMi), The Open University.
Walton Hall, Milton Keynes, MK7 6AA, United Kingdom.
{v.lopez, r.m.sabou, e.motta}@open.ac.uk

Abstract. Ontology mapping plays an important role in bridging the semantic gap between distributed and heterogeneous data sources. As the Semantic Web slowly becomes real and the amount of online semantic data increases, a new generation of tools is developed that automatically find and integrate this data. Unlike in the case of earlier tools where mapping has been performed at the design time of the tool, these new tools require mapping techniques that can be performed at run time. The contribution of this paper is twofold. First, we investigate the general requirements for run time mapping techniques. Second, we describe our PowerMap mapping algorithm that was designed to be used at run-time by an ontology based question answering tool.

Keywords: Semantic Web, question answering, heterogeneity, and ontology mapping.

1 Introduction

The Semantic Web (SW) is evolving towards an open, distributed and heterogeneous environment. Core to the information integration tasks that would be supported by SW technology are algorithms that allow matching between the elements of several, distributed ontologies. The importance of mapping for the SW has been widely recognized [1] and a range of techniques and tools have already been developed. However, the predominant view of mapping is that it will be performed at ***"design time"***, e.g. when deciding on mapping rules between a set of ontologies [2]. This was a plausible assumption because, until recently, only a limited amount of semantic data was available; therefore, there was little need for run time integration. Indeed, one of the main characteristics of SW based applications built so far is that they tackle the data heterogeneity problem in the context of a given domain or application by integrating a few, a-priori determined sources [3, 4]. As such, they act more as smart, database centered applications rather than tools that truly explore the dynamic and heterogeneous nature of the SW [5].

Recently, things have started to change. There is now a reasonable amount of online semantic data, to such an extent that the need has arisen for a semantic search engine, Swoogle [6], which can crawl and index all these data. Hence, we are now slowly reaching a key point in the history of this very young discipline, where we can start moving away from the early applications characterized by limited heterogeneity and start developing the kind of applications, which will define the SW of the future.

These tools will dynamically find and integrate data from online available sources depending on their current information need. However, mapping still remains an important step. Rather than being performed during the development of the application it now needs to be performed at ***"run time"***. Obviously, this new scenario brings novel challenges for ontology mapping techniques.

In this paper we present a mapping algorithm, PowerMap, which is a core component of the PowerAqua ontology based question answering system. PowerAqua belongs to the new generation of SW tools as it tries to answer questions asked in natural language by leveraging on the semantic data available online. As a result, PowerMap needs to be able to create mappings between heterogeneous data on-the-fly and with no pre-determined assumption about the source and the ontological structure of these data.

The paper is organized as follows. Section 2 provides a perspective about the novel scenarios that the evolving SW tools will impose on mapping. Section 3 describes the context in which our own mapping algorithm was developed, the PowerAqua question answering system, and illustrates through an example some of the challenges that such run time mapping operations face. Section 4 details the major design components that underlie PowerMap. In Section 5 we present the details of the algorithm. Finally, we provide an example (Section 6). We summarize in Section 7.

2 Mapping in the Context of Semantic Web Tools

The problem of ontology schema mapping has been investigated by many research groups which have proposed a large variety of approaches [1, 7]. While all this research has produced increasingly complex algorithms, the setting in which the mapping problem was tackled was almost always the same: given two ontologies, find all the possible mappings between their entities attaching a confidence level to the mappings that are returned. One of the challenges in the field of ontology mapping now is not so much perfecting these algorithms, but rather trying to adapt them to novel scenarios, which require SW applications to automatically select and integrate semantic data available online. Obviously, mapping techniques are crucial in achieving this goal. However, the setting in which the mapping would take place is quite different from the "traditional" ontology mapping scenario. Indeed, the focus is not on mapping complete ontologies but rather small snippets that are relevant for a given task. These new scenarios impose a number of requirements:

1) **More ontologies** – when integrating data from online ontologies it is often necessary to map between **several** online ontologies. This is very unlike the traditional scenario where only two ontologies were mapped at a time.
2) **Increased heterogeneity** – traditional mapping techniques often assume that the ontologies to be matched will be similar in structure, describe more or less the same topic domain. For example, S-Match [8] is targeted towards matching classification hierarchies. Or, due to its structure based techniques, Anchor-PROMPT [9] works best if the matched ontologies have structures of similar complexity. Such similarity assumptions fail on the SW: we cannot predict whether relevant information will be provided by a simple FOAF file or by WordNet, or top level ontologies, or combined

from these different sources. Mapping techniques should function without any pre-formulated assumptions about the ontological structure.

3) **Time Performance is important** - As already pointed out in [10], the majority of mapping approaches focus on the effectiveness (i.e., quality) of the mapping rather than on its efficiency (i.e., speed). This is a major challenge that needs to be solved in the context of run-time mappings where the speed of the response is a crucial factor. The above mentioned paper also shows that some minor modifications of the mapping strategy can highly improve response time and have only a marginal negative effect on the quality of the mappings. Unfortunately the work presented in [10] is rather unique in the context of mapping research -- although we think that such research is crucial for making mapping techniques usable during run-time.

4) **Consider relation and instance mappings** – much of the work in ontology mapping has focused on matching the concepts in two schemas, while other ontology entities, such as relations and instances, have largely been ignored so far (although relations and instances are taken into account as evidence to support the matching process in some approaches). However, SW tools are often used to find out information about specific entities (traditionally modeled as ontology instances), as well as the relations between entities. Therefore, we think that mapping techniques should be developed to efficiently map also between these kinds of entities, for example, on instance mapping, by reusing earlier work on tuple matching from the database community.

5) **Cross-ontology mapping filtering** - several approaches adopt the model of first generating all possible mappings and then filtering the relevant ones. However, in these approaches mappings are typically created between two ontologies describing the same domain. When performing mappings on the SW, we are also likely to discover several mappings but this time the mapping candidates might be drawn from different ontologies. Therefore we need to be able to reason about ontologies which may only have very few concepts in common. As discussed later in this paper, this requires mechanisms to assess whether or not such 'sparse concepts' are related.

6) **Produce Semantic output** – with the exception of S-Match, most mapping algorithms simply determine a similarity coefficient between the concepts that are mapped. Such coefficients are not very useful if the mappings have to be automatically used by a tool. In the scenario of SW tools, to support automatic processing of the mapping results, it would be more useful to return the semantic relations between the mapped entities (equivalent, more generic/specific) rather than just a number.

3 Motivating Scenario: Question Answering on the Semantic Web

Question answering has been investigated for many years by several different communities [11] (e.g., information retrieval). These approaches have largely been focused on retrieving the answer from raw text[1]. An obvious hypothesis is that QA would become easier if the answers could be retrieved from semantic data.

[1] Sponsored by the American National Institute (NIST) and the Defence Advanced Research Projects Agency (DARPA), TREC introduced an open-domain QA track in 1999 (TREC-8).

Based on this hypothesis, we have developed the **AquaLog** [12] ontology-based question answering system. The novelty of the system with respect to traditional QA systems is that it relies on the knowledge encoded in the underlying ontology and its explicit semantics to disambiguate the meaning of the questions and to provide answers. AquaLog has been developed during a period when little semantic data was available online. As a result it only uses one ontology at a time, even though AquaLog is portable from one domain to the other, being agnostic to the domain of the ontology that it exploits. In other words, while AquaLog is ontology independent, the user needs to tell the system which ontology is going to be used to interpret the queries. To briefly illustrate the question answering process, imagine that the system is asked the following question: *"Who are the researchers in KMi that have publications at ISWC?"*. The major task of the system is to bridge between the terminology used by the user and the concepts used by the underlying ontology. In a first step, by using linguistic techniques, the system breaks up the question into the following binary linguistic triples *(person, researcher, Knowledge Media Institute) (?, have publication, ISWC)*. Then, these terms are linked and mapped to ontology elements, generating the following ontology compliant triples *(researcher, works-for, knowledge-media-institute-at-the-open-university) (researcher, has-publications, international-semantic-web-conference)* from where the answer is derived. Obviously, if one of the terms of the question cannot be mapped to the ontology then no answer will be retrieved.

One way to overcome this limited scope is to take advantage of online available semantic data. The new version of AquaLog, **PowerAqua** [13], adopts an "open question answering strategy" by consulting and aggregating information derived from multiple heterogeneous ontologies on the Web. PowerAqua will function in the same way as AquaLog does, with the essential difference that the terms of the question will need to be **dynamically** mapped to **several** online ontologies. This run time mapping brings up several challenges in comparison with Aqualog, which need to be solved by the PowerMap mapping algorithm of PowerAqua:

a) **Finding the right ontologies**. PowerMap matching operations first need to determine the ontology(ies) from where the answer will be derived. Syntactic matching techniques can be used in a first step to identify all those ontologies with potential mappings to the terms in the triples. For example, "researcher" is a concept appearing in almost all ontologies about the academic domain, while "KMi" appears only in one of those ontologies[2]. For the second triple, we find many concepts related to "publication" and "iswc".

b) **Semantic relevance analysis.** When multiple mapping candidates are discovered, only semantically relevant ones should be selected.

c) **Filtering the right mappings.** From the identified ontologies the ones that potentially provide the most information need to be selected. PowerMap relies on two criteria. First, at least a complete mapping coverage for each triple is crucial (i.e., one triple should not be spread over many ontologies. However, triples can be mapped over several ontologies that provide equivalent information, or whose information can be partially combined and integrated through similar semantically interoperable

[2] This populated ontology can be browsed through at: http://semanticweb.kmi.open.ac.uk.

classes from different ontologies to provide a complete mapping. In the example above, it is easy to choose the first ontology which completely covers the "researcher" and "KMi" terms. Second, in case of ambiguity (more than one interpretation of the same query term) the correct interpretation for the given term in the context of the user query (triples) and the ontology relatedness should be returned. In case this is not enough to perform disambiguation, the final decision should be left to the target tools or to the user. For the second triple, there are considerably more ontologies that completely cover the triple. However, all of them are semantically equivalent solutions. Nevertheless, only one ontology contains a path between "researcher", "publication" and "iswc".

d) **Composing heterogeneous information. PowerAqua** will use PowerMap mapping results to find the relations that link those entities and the triples, so the resulting ontology triples will be *(researcher, works-in, knowledge-media-institute)* referring to the KMi ontology and, e.g., the triples *(researchers, wrote, publications) (publications, published, iswc)* in a second ontology, although other equivalent triples in other ontologies will also be valid. Finally, PowerAqua needs to combine partial answers from these different ontologies, e.g., to obtain the researchers on KMi and the researchers that have publications at *ISWC*. Among other things, to give an answer this requires the ability to recognize whether two instances from different sources may refer to the same individual. Some co-relation and disambiguation methods to determine if two resources refer to the same individual have been used in Flink [4]

4 PowerMap at a Glance

The requirements imposed by SW applications, like PowerAqua, that open up to harvest the rich ontological knowledge on the Web are the foundations for the design of PowerMap. In PowerMap the **mapping process is driven by the task** that has to be performed, more concretely by the query that is asked by the user. Indeed, this is novel in comparison with traditional approaches where mappings are done prior to the ontology being used for a specific task. An input query is represented by a triple or set of triples that indicated how the words are related together (in fact, better results are expected considering the triples than by only considering isolated words). We envision a scenario where a user may need to interact with thousands of knowledge bases structured according to hundreds of ontologies. However, we believe that good performance could be obtained also at such scale because PowerMap avoids a global interpretation of the mapped ontologies, in which the level of effort is at least linear in the number of matches to be performed [8] (e.g., the Match operator). In this sense only relevant concepts to the user's query are analyzed.

PowerMap is a hybrid matching algorithm comprising *terminological and structural schema matching* techniques with the assistance of large scale ontological or lexical resources. Figure 1 depicts the three main phases of PowerMap.

Phase I: Syntactic Mapping. The role of this phase is to identify candidate mappings for all query terms in different online ontologies (therefore identify potentially relevant ontologies for that particular query). This is the simplest phase as it only considers concept labels (i.e., ignores the structure of ontologies). It relies on simple,

string-based comparison methods (e.g., *edit distance metrics*) and WordNet to look-up lexically related words (synonyms, hypernyms and hyponyms).

Phase II: Semantic Mapping. This phase operates on the reduced set of ontologies identified in the previous phase. The goal is to verify the syntactic mappings identified previously and exclude those that do not make sense from a semantic perspective (e.g., the intended meaning of the query term differs from the intended meaning of the concept that was proposed as a candidate match). For example, if the term "capital" is matched to concepts with identical labels in a geographical ontology and a financial ontology, these two meanings are not semantically equivalent. Unlike the previous phase, this phase relies on more complex methods. First, it exploits the hierarchical structure of the candidate ontologies to elicit the sense of the candidate concepts. Second, it uses WordNet based methods to compute the *semantic similarity* between the query terms and the ontology classes.

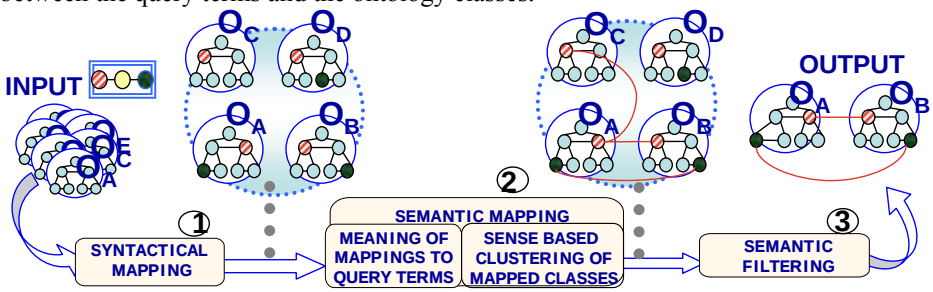

Fig. 1. Mapping process example to obtain potential ontology mappings for a triple.

Phase III: Semantic Filtering. The mappings filtered out by the previous phase are spread over several ontologies. The goal of this final phase is to filter out the meaningful mappings that better represent the query domain by (a) determining those ontologies that cover entire triples and not just individual terms of the triples and by (b) studying the ontology relatedness to determine the valid semantic interpretation (e.g. to decide which ontology interpretation of "capital" is valid for the sense of the query term). In this phase we employ relation mapping techniques to match between the predicates of the triples and relations in the identified ontologies. This step will return a small set of ontologies that jointly cover all terms and hopefully contain enough information to deduce the answer to the question.

Note that, in order to optimize performance, the complexity of these phases increases both because of the type of ontology entities that they consider and because of the techniques they use. Hence the most time-consuming techniques are executed last, when the search has been narrowed down to a smaller set of ontologies.

5 Details of the PowerMap Algorithm

We explore each major step of this algorithm that is currently being implemented.

5.1 Phase I: Syntactic Mapping

The syntactic mapping phase identifies candidate entities from different ontologies to be mapped to each input term in the triple(s) by means of syntax driven techniques (SDT) using the labels and local names of the ontology elements. We test our prototype on a collection of ontologies saved into online repositories but in the mean time we are working on adapting it to directly fetch relevant ontologies from Swoogle.

This phase is responsible to bridge the gap between user terminology and the multiple heterogeneous ontologies. This is done through two mechanisms. First, the set of query terms is broadened with semantically equivalent terms using WordNet. We take into account synonyms, hypernyms and hyponyms. Currently we experiment with using the SUMO upper level ontology and extending it with the mappings to the WordNet lexicon [14]. The mappings of SUMO to WordNet avoid the excessive fine-grainedness of WN sense distinctions, which is the most frequently cited problem of WordNet [15]. The second mechanism to ensure a high recall, is to perform so called "fuzzy" syntactic matches between terms and ontology entities (e.g. "PhDStudent" is a fuzzy match to "Student"). We are also considering the use of wikipedia [16] to find similar names, abbreviations, and acronyms in the case of instances.

SDT (fuzzy searches and lexically related words) are good mechanisms to broaden the search space as they can return a lot of hits that contain the term. However, they have two main weaknesses. First, SDT become increasingly computationally expensive as the number of ontologies increases. Second, many of the discovered ontology elements syntactically related with the query terms may be similarly spelled words (labels) that do not have precisely the same meaning.

The first weakness is addressed by using efficient and large-scale ontology repositories [17] in combination with Lucene[3]. Lucene indexes the semantic entities in the online and distributed back-end repositories into one or more indexes, and is used as our fast search engine[4], which supports fuzzy searches based on the Lavenshtein Distance, or Edit Distance algorithm. Moreover, it includes a *Spell Checker* to suggest a list of words close to a misspelled word using the n-gram technique. Also, query terms and in some cases relations are mapped to instances or classes therefore the system searches for classes, instances, properties and literals. Studying relations is computationally expensive and it is done only after the arguments are well know (although if one of the argument is unmapped, they can also be used to broaden the search space of candidate classes, i.e., through the ontology relationships that are valid for the mapped term, we can identify a set of possible candidate classes that can complete the triple). Relations are considered on the third phase to help filtering out the most relevant mapping candidates (Section 5.4).

The second weakness is addressed in the next semantic phases, where we will focus on the issue of checking the semantic validity of the mappings and disambiguating among the possible interpretations of a query.

---

[3] http://lucene.apache.org

[4] A first implementation of the search engine can be found on the KMi semantic web portal: http://semanticweb.kmi.open.ac.uk:8080/ksw/pages/semantic_searching.jsp

5.2 Phase II: Semantic Mapping

Semantic mapping checks the semantic validity of the previously identified syntactic ontology mappings for each query term in the triple. We perform two main steps. In the first step (Section 5.3.1) we discard mappings established between terms and concepts with different meaning. Then, in the next step (Section 5.3.2), we cluster the resulting mappings according to the senses that they cover. These steps rely on two more generic algorithms to determine the similarity between two senses in WN (Section 5.3.3) and to obtain the meaning of a concept and compare the senses of concepts in different ontologies (Section 5.3.4).

5.2.1 Step1: Verifying the meaning of mappings respect to query terms

In this step we verify whether a mapping is also valid at a semantic level, i.e., the intended meaning of the term is the same as that of the concept. Mappings between elements with completely different meaning will be discarded (e.g. the "research-area" = "researcher" mapping).

We rely on sense information provided by WordNet to check the semantic similarity between the mapped terms. We perform the following steps:

1) For a term T, we extract all its WordNet senses, S_T
2) For the proposed mapping of T, a concept C, we also extract its senses S_C
3) We compute semantic similarity, using the algorithm in Section 5.3.3, between T and C to obtain the shared senses $S_{T,C}$
4) Based on the value of $S_{T,C,}$ we determine the semantic relation between T and C as follows
    a. If $S_{T,C,}$ is empty, the terms share no sense, and therefore the mapping is discarded
    b. If $S_{T,C,}$ is not empty there is a semantic relation between the two terms which needs to be further investigated ($S_{T,C,} \leq S_C$)
    c. If $S_{T,C,} = S_T$, then the terms share all senses and they are potentially semantically identical (see "capital" example in Section 6).

Note that in this step we took into account all possible senses for C. However, the true senses of C are determined by its place in the hierarchy of the ontology. Because this sense is more costly to compute, to improve performance we use it only in the next step after the obviously wrong mappings have been discarded in this step.

5.2.2 Step 2: Sense-based clustering of retrieved mappings

The previous step might result in several mappings for the same term to concepts in different ontologies and these ontologies might have different subject domains (thus enforcing different meaning on their concepts). In this step we compare the concepts to which the term is mapped to determine whether they have the same sense (in case of instances we study the class they belong to). For this we rely on their place in the hierarchy of the ontology.

Apart from the senses being delimited by the query term $\{S_{T,C}\}$, the senses of the candidate mapped ontology class C are also delimited by its meaning in the ontology.

For each concept C, we determine its sense as restricted by its place in the hierarchy S^H_C by using the algorithm presented in Section 5.3.4.. We then intersect this sense with the senses that C and T share according to our previous step, $S_{T,C,}$. Obviously, if this intersection ($S^H_C \cap S_{T,C}$) is empty it means that the sense of the concept in the hierarchy is different from the sense that we though it might have in the previous step, and therefore that mapping should be discarded. Otherwise, the intersection represents the sense which is captured by the mapping. For example, if the term "queen" was previously mapped to two concepts Bee/Queen and Royalty/Queen having the same label, after interpreting the meaning of the two concepts according to their parent concept, we deduce that they have two different meanings as their intersection with the senses of the query term contains different senses (in case the two mapped concepts don't share the same label, the intersection or shared senses are computed using the semantic similarity notion on Section 5.3.3). Mappings with different meanings or interpretations are not semantically equivalent and therefore the correct interpretation should be disambiguate and filtered in the next step (5.3.4).

We group the mappings that refer to the same sense together.

5.2.3 Computing Semantic Similarity

In this section we detail the semantic similarity algorithm used to find shared senses of two words by relying on WordNet. In *Hierarchy distance based matchers* [18] the relatedness between words is measured by the distance between the two concept/senses in a given input hierarchy. In particular, similarity between words is measured by looking at the shortest path between two given concepts/sense in the WN *IS-A* taxonomy of concepts. Note that similarity ("bank-trust") is a more specialized notion than association or relatedness (i.e. any kind of functional relationship or frequent association, which cannot always be determined purely from a priori lexical resources such as WN, like "penguin-Antarctica") [19].

We say that **two words are similar** if any of the following hold:

1. They have a synset(s) in common (e.g. "human" and "person")
2. Any of the senses of a word is a hypernym/hyponym in the taxonomy of any of the senses of the other word.
3. If there exists an allowable "is-a" path (in the WN taxonomy) connecting a synset associated with each word. To evaluate this, we make use of two WN indexes: the *depth* and the *common parent index* (C.P.I). The rationale of this point is based on the two criteria of similarity between concepts established by Resnik in [20]. The first one is that the shorter the path between two terms the more similar they are, this is measured using the *depth* index. However, a widely acknowledged problem is that the approach typically "relies on the notion that links in the taxonomy represent uniform distances", but typically this is not true and there is a wide variability in the "distance" covered by a single taxonomic link [19]. As a consequence the second criterion of similarity is the extent to which the concepts share information in common, which in an *IS-A* taxonomy can be determined by inspecting the relative position of the most-specific concept that subsumes them both, which is the *C.P.I* index. With the use of the *C.P.I* we can immediately identify the lowest super-ordinate concept (*lso*) between the two terms, also called

the most specific common subsumer. Apart from point 1 of the algorithm, in which the words have a synset in common, the most immediate case occurs in point 2 (C.P.I = 1, Depth = 1), e.g. while comparing "poultry" and "chicken" we notice that "poultry#2" is the common subsumer (hypernym) of "chicken#1".

4. Additionally, if any of the previous cases is true and the definition (gloss) of one of the synsets of the word (or its direct hypernyms/hyponyms) includes the other word as one of the synonyms, we say they are strongly similar.

For example, for the input triple (investigators, work, akt project) using string algorithms over WordNet synonyms, PowerMap discovers the following candidate mappings for "investigators": "researcher", "research-area". Going back to the step 1 (Section 5.3.1), using the WordNet "IS-A" taxonomy we must find at least one synset in common with the mapped ontology class and the query term or a short/relevant path in the IS-A WordNet taxonomy that relates them together. Otherwise it is discarded as a solution. Here, "researcher" and "investigator" have a synset in common, namely "research-worker, researcher, investigator – a scientist who devotes himself to doing research". However "research-area" will be discarded because not only do they not share any sense in common but also there is not a relevant "IS-A" path that connects "researcher" with "research-area" -- "researcher" is connected to the root through the path "scientist/man of science" and "person", while "research-area" is connected through "investigation" which is connected to "work".

5.2.4 Ontology Structure based sense disambiguation

The meaning of an ontology term should be made explicit by an interpretation of its label through a WordNet sense and its position in the ontology taxonomy.

According to the algorithm presented by Magnini et al. in [21] to make explicit the semantics hidden in schema models, in a nutshell, given a concept *c* and either one of its ancestors or descendants *r* all WordNet synsets for both labels are retrieved. Then, if any of the senses for *c* is related to any of the senses for *r* either by being a synonym, hypernym, holonym, hyponym or a meronym, then that sense of c is considered the right one.

Our algorithm, originally based on Magnini et al., is adapted to use ontologies rather than catalogues or classifications, therefore we can exploit the use of the notion of similarity *IS-A* given by the ontology taxonomy explicit semantics instead of the notion of relatedness (e.g. "hospital" is not a good match for "nurse" even if they are highly related). The WN senses of an ontology class are obtained by looking at the similarity (as previously defined) between the class and its ascendant/descendant in the ontology. The senses of the class that are similar to at least one of the senses of its ascendant/descendant are retained and the rest of the senses discarded.

5.3 Phase III: Semantic Filtering

Having worked at the level of individual term mappings so far, in this step we select those ontologies that cover *entire* triples (ontologies with better domain coverage).

Moreover, we take advantage of the relatedness expressed in the ontology semantics and input triples to filter out the semantically interoperable candidate mappings for the query terms. Also, if different ontologies cover different triples then we must make sure that the concepts that link between the triples have the same sense in those ontologies (semantic interoperable concepts, as studies in Section 5.3.2).

Previous steps have only determined mappings of concepts and instances from the query. The reason for this is based on our experience with AquaLog where mapping relations is more difficult than mapping concepts. In the case of PowerAqua due to the increasing number of heterogeneous ontologies the challenge is to semantically map the terms. Once the terms are mapped the meaning of a relation is given by the type of its domain and its range rather than by its name (typically vaguely defined as e.g. "related to"), so the precondition of a mapping between two relations is that their domain and range classes match to some extent. With the exception of the cases in which some relations are presented in some ontologies as a concept (e.g. *has Author* can be modeled as a concept *Author* in a given ontology), in PowerMap relations are treated as "second class citizens" to help disambiguating the candidate classes, and ontologies, that better cover the query domain.

The following is a disambiguating example considering the coverage criterion. The query "Which wine is appropriate with chicken?" translated into the triple (wine, appropriate, chicken) has syntactic mappings with the class "wine" in an ontology of colors, and in an ontology of food and wines. Similarly, the term "chicken" maps to an ontology of farming and to the same food and wine ontology. Since the food and wine ontology presents a complete potential translation for the triple we retain it, and we discard the partial translations from both the farming and color ontologies. A disambiguating example using ontology relations is described in Section 6.

6 Experimental Example

In this section we present an example run on our prototype. Consider the query "what is the capital of Spain?" translated in a triple without information about the *focus* of the query: (?, capital, Spain). After the execution of phase I we get the following mappings for the terms and their lexical variations:

- Geographical ontology. Contains the class "capital-city" and "Spain" as an instance of "country", "capital-city" and "country" are connected by a direct relation.
- Financial ontology. Contains the class "capital" and "Spain" as an instance of "country", "capital" and "country" are related through the concept "company".
- Country statistics ontology. Contains the term "Spain".

The coverage criterion can be already applied to this stage of the algorithm, however the three interpretations will remain because both ontology 1 and 2 cover the terms "capital" and "Spain", and ontology 3 only covers the term "Spain" but "capital" is considered as a relation and as such it may be mapped into an ontology relation.

There is only one possible sense for "Spain", therefore we only study the semantic similarity for the term "capital". In principle, both interpretations remain (step 1, Section 5.3.1), as the lemma for both terms is the same as the query term, potentially

they have all the synset in common. Semantic equivalence between both classes is then determined by studying their ontology meaning (step 2, Section 5.3.2.). When running the similarity algorithm between "capital" and its ancestor "city" in the geographical ontology we obtained the results presented in Table 3.

Table 1. Similarity between "capital" and its ontology ancestor "city"

	City#1: large and densely populated urban area.., metropolis	**City#2:** an incorporated administrative district ..	**City#3:** people living in large municipality
Capital#a (assests ..)	*Not an allowable path or depth is too long to be considered relevant*		
Capital#b (wealth ..)			
Capital#c (seat of government)	**Depth** = 8, **lso** = region **Num_so**(common_subsumers) = 3 (region, location, entity)	**Depth** = 7, **lso** = region **Num_so** = 3 (entity, location, region)	
Capital#d (capital letter)			
Capital#e (book by Karl Marx)			
Capital#f (upper part column)	**Depth** = 8**, lso** = location **Num_so** = 2 (entity, location)	**Depth** = 7, **lso** = location **Num_so**=2 (entity,location)	

Analyzing the results of Table 1 we can quickly filter *capital#c*, *capital#f, city#1, city#2* and discard the others. A deeper study will show that *capital#c* is more likely than *capital#f* because there are only 2 common subsumers in the latter (entity and location), both of them representing abstract top elements of the WordNet taxonomy, while in the former we have 3 common subsumers. We can not study the descendants of "capital" in the ontology because none exist. The study of the next direct ascendant of "city" ("geographical-unit") does not offer additional information. Moreover, the hypernym of *capital#c* is "*seat#5*", defined as "seat –centre of authority (*city* from which authority is exercised)". The word "city" is used as part of its definition, therefore *capital#c* is strongly related to "*city*".

After the semantic similarity analysis the sense of "capital" is made explicit as senses #1 and #2 in the financial ontology, while the geographical ontology is referred to sense *#3*. Therefore both terms in different ontologies are not semantically equivalent and the system must select one of them using ontology semantics or query relatedness. Using SUMO's mapping files to WordNet synsets we can identify senses that are not very distinctive (they are mapped to the same SUMO concept), e.g. for city {#1 an incorporated administrative district, #2: metropolis, and #3: people living in large municipality}, all its senses map to the same SUMO class.

A deeper analysis of the ontology relationships to narrow down between the two valid non-equivalent mappings "capital" shows a direct relation that connects any country, e.g. Spain, with its capital for the geographical ontology. However, in the financial ontology there is not a direct relation between countries and capital. There is a mediating concept that represents a company, that has a series of capital goods and it is based in a country. This is a strong indication that the geographical ontology is more related to our query and should be selected. For the country statistics ontology, where capital is considered a relation, a relationship analysis simply using of string distance metrics [22] will uncover the relation "is-capital-of" between "country" and

"city". Therefore both mappings in the geographical and statistics ontologies will be valid semantically equivalent representations of the query.

7 Summary

The main message of this paper is that the new context introduced by the evolving SW tools will require mapping techniques that can be used at run-time rather than at the design time of such tools and applications. Our main contribution is to recognize and analyze this need which could present a turning point in the field of ontology mapping. We presented some of the requirements that have to be addressed by such novel mapping techniques. In particular, such techniques need to balance the heterogeneity and large scale of online available semantic data and the requirement of being fast so that they can be used at run-time.

The core of the paper exemplifies the requirements for run time ontology mapping in the context of a concrete application, PowerAqua, an ontology based QA system and then describes the PowerMap algorithm which performs such run-time mappings. Unlike traditional mapping algorithms, PowerMap is focused towards dealing with several, heterogeneous ontologies which are not given a priory but rather discovered depending on the content of the user's query (thus we fulfill requirements 1 and 2). To maintain a good performance, as requested by our third requirement, PowerMap employs three steps that are increasingly complex: we start with syntactic mappings that take into account only concept labels to find potentially useful ontologies, then we rely on WordNet information and on the meaning of the mapped concepts in their hierarchy to verify that the proposed mappings are also semantically sound. Finally, we rely on the structure of the triples and techniques to map between relations in order to filter out a set of relevant ontologies from which PowerAqua will extract the answers (requirement 5).

PowerMap is currently under implementation and our obvious future work is in finalizing the prototype and evaluating it. In particular we are working on extending the technique to work directly with Swoogle and to provide mappings between instances as well (see requirement 4). However, we think that our ideas about run-time ontology mapping and the proposed algorithm could benefit the ontology mapping community in particular, and the SW research in general.

Acknowledgments. This work was partially supported by the AKT project sponsored by UK EPSRC and by the EU OpenKnowledge project (FP6-027253). Thanks to Yuangui Lei and Victoria Uren for all the technical help and relevant input.

References

1. Shvaiko, P., Euzenat, J.: A Survey of Schema-Based Matching Approaches. *J. of Data Semantics IV* , 2005. 146-171.
2. Bouquet P., Serafini L. and Zanobini S. Semantic coordination: a new approach and an application. *In Proc of ISWC,* 2003, 130-145.

3. Hyvonen, E., Makela, E., Salminen, M., Valo, A., Viljanen, K., Saarela, S., Junnila, M., Kettula, S.. MuseumFinland – Finnish Museums on the SemanticWeb. *Journal of Web Semantics*, 3(2), 2005
4. Mika, P. Flink: SemanticWeb Technology for the Extraction and Analysis of Social Networks. *Journal of Web Semantics*, 3(2), 2005
5. Motta, E., Sabou, M., Language Technologies and the Evolution of the Semantic Web. *In Proceedings of LREC*, 2006
6. Ding, L., Pan, R., Finin, T., Joshi, A., Peng, Y., Kolari, P. Finding and Ranking Knowledge on the Semantic Web. *In Proceedings of ISWC*, 2005, p. 156 – 170.
7. Rahm E. and Bernstein P. A. A survey of approaches to automatic schema matching. *The International Journal on Very Large Data Bases* 10(4): 334-350, 2001.
8. Giunchiglia F., Shvaiko P and Yatskevich M. S-Match: an algorithm and an implementation of semantic matching. *In Proc. of the 1st European Semantic Web Symposium*, 2004.
9. N. Noy and M. Musen. Anchor-PROMPT: using non-local context for semantic matching. *In Proceedings of the workshop on Ontologies and Information Sharing at the International Joint Conference on Artificial Intelligence (IJCAI)*, 2001, pages 63–70.
10. M. Ehrig and S. Staab. QOM: Quick ontology mapping. In *Proceedings of ISWC*, 2004, pages 683–697.
11. Hirschman, L., Gaizauskas, R.: Natural Language question answering: the view from here. ***Natural Language*** *Engineering, Special Issue on* ***QA***, 7(4) 275-300, 2001
12. Lopez V., Pasin M. and Motta E. AquaLog: An Ontology-portable Question Answering System for the Semantic Web. *In Proc. of ESWC,* 2005.
13. Lopez V., Motta E. and Uren, V. PowerAqua: Fishing the Semantic Web. *In Proc. of ESWC,* 2006.
14. Pease, A., Niles, I., and Li, J. The Suggested Upper Merged Ontology: A Large Ontology for the Semantic Web and its Applications. *In Working Notes of the AAAI Workshop on Ontologies and the Semantic Web*, 2002.
15. Ide N. and Veronis J. Word Sense Disambiguation: The State of the Art. *Computational Linguistics*, 24(1):1-40, 1998.
16. Bunescu, R., Pasca, M. Using Encyclopedic Knowledge for Named Entity Disambiguation. *In Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL-06)*, 2006.
17. Guo, Y., Pan, Z., Heflin, J. An Evaluation of Knowledge Base Systems for Large OWL Datasets. *In Proc of ISWC*, 2004, pages 274-288.
18. Giunchiglia F. and Yatskevich M. Element Level Semantic Matching. *Meaning Coordination and Negotiation Workshop, ISWC*, 2004.
19. Budanitsky, A. and Hirst, G. Evaluating WordNet-based measures of semantic distance. *Computational Linguistics*, 2006.
20. Resnik P. Disambiguating noun grouping with respect to WordNet senses. *In Proc. of the 3rd Workshop on very Large Corpora*. MIT, 1995.
21. Magnini B., Serafin L., and Speranza M. Making Explicit the Semantics Hidden in Schema Models. *In Proc. of the Workshop on Human Language Technology for the Semantic Web and Web Services*, held at *ISWC-2003*, Sanibel Island, Florida, 2003.
22. Cohen, W., W., Ravikumar, P., Fienberg, S., E.: A Comparison of String Distance Metrics for Name-Matching Tasks. *In Proc. of the 2nd Web Workshop at IJCAI ,2003.*