Cross Ontology Query Answering on the Semantic Web: An Initial Evaluation

Vanessa Lopez v.lopez@open.ac.uk
Victoria Uren v.s.uren@open.ac.uk
Marta Sabou r.m.sabou@open.ac.uk
Enrico Motta e.motta@open.ac.uk
The Knowledge Media Institute. The Open University. MK76AA. Milton Keynes. United Kingdom.

ABSTRACT
PowerAqua[1] is a Question Answering system, which takes as input a natural language query and is able to return answers drawn from relevant semantic resources found anywhere on the Semantic Web. In this paper we provide two novel contributions: First, we detail a new component of the system, the *Triple Similarity Service,* which is able to match queries effectively to triples found in different ontologies on the Semantic Web. Second, we provide a first evaluation of the system, which in addition to providing data about PowerAqua's competence, also gives us important insights into the issues related to using the Semantic Web as the target answer set in Question Answering. In particular, we show that, despite the problems related to the noisy and incomplete conceptualizations, which can be found on the Semantic Web, good results can already be obtained.

Categories and Subject Descriptors
I.2.7 Natural Language Processing – *semantic web, multi-ontology question answering, knowledge acquisition*.

General Terms: Design

INTRODUCTION
Recent years have witnessed a resurgence of interest in Natural Language (NL) Interfaces to knowledge bases and in particular the rise of a new paradigm of research, which can be termed as *Ontology-Based Query Answering* [3][1][9][13][11]. These systems use an underlying ontology to drive and/or to give meaning to the queries expressed by a user. In general, these systems are ontology-modular, i.e., they can be used for different domains, even though, in practice they differ considerably in the degree of domain customization they require. At one end of the spectrum, AquaLog [9] uses interactivity to learn user terminology over time and does not necessitate any customization effort; at the other end of the spectrum, a system such as Orakel [3] requires significant domain-specific lexicon customization.

Unfortunately, regardless of the various fine-grained differences between them, all the aforementioned systems arguably suffer from an important limitation: they are in practice only suitable for semantic intranets, where a pre-defined domain ontology (or a set of them) is used to provide a homogeneous encoding of organizational data. In such a scenario ontology-driven interfaces have been shown to effectively support the user in formulating complex queries, without resorting to formal query languages. However, any information which is either outside the semantic intranet, or simply not integrated with the corporate ontology remains out of bounds.

In the meantime we are also seeing a dramatic increase in the amount of semantic markup available on the web, with ontology search engines, such as Sindice (http://sindice.com/), claiming to index "over 10 billion pieces… across 100 million web pages". As discussed in [4], the availability of this large amount of heterogeneous semantic markup is unprecedented in the history of Artificial Intelligence and may provide the semantic basis for a new generation of intelligent systems. At the same time, the emergence of a large scale Semantic Web (SW) introduces a new challenge: how can we support users in querying and exploring this novel, massively heterogeneous, structured information space? In particular, the 'static' ontology-based query answering systems mentioned earlier cannot cope with the sheer scale and heterogeneity of the SW. The keyword interfaces provided by ontology search engines such as Swoogle (swoogle.umbc.edu/), Watson (watson.kmi.open.ac.uk/), or Sindice work reasonably effectively when asked to find specific items, such as "Enrico Motta" or "Researcher", however, they cannot answer more complex queries, such as "Which Russian rivers end in the Black Sea?".

Hence, in this paper we tackle the problem of supporting users in locating information on the SW. Our approach, in contrast with the keyword-based ontology search interfaces, is based on providing a NL Interface, PowerAqua, which is able to accept user queries expressed in NL and retrieve answers from any semantic source on the SW[2]. In contrast with the ontology-based query answering tools mentioned earlier, PowerAqua is not restricted to a specific set of ontologies, but can in principle retrieve answers from many semantic sources.

The ideas behind PowerAqua were originally presented in [8], where we introduced the vision underlying the system and the key challenges facing the proposed research. In [10] we provided a detailed description of the core engine of the

[1] An online demo of the system can be found at: http://kmi.open.ac.uk/technologies/poweraqua.

K-CAP'09, September 1–4, 2009, Redondo Beach, California, USA.

[2] Given that PowerAqua accesses the Semantic Web through the Watson Semantic Web Gateway [4], in practice PowerAqua will only retrieve information if this has been crawled and indexed by Watson.

system, the PowerMap component, which dynamically finds the potentially relevant ontologies and element mappings for a given NL query. In this paper we report on the first complete implementation of the system, we describe the novel Triple Similarity Service component, and give a comprehensive account of the way the system returns answers to queries. In addition we also present an initial evaluation of the system, which in addition to providing data about PowerAqua's competence, also gives us important insights about the current strengths and limitations of the SW when used for question answering. The rest of the paper is organized as follows: we describe the system overall architecture in Section 2, introduce the novel Triple Similarity Service component in Section 3, and give a brief overview of our current work in integrating and ranking answers from different sources in Section 4. We then present the results obtained from an initial evaluation of the system in Section 5, while in Section 6 and 7 we discuss related work, draw the key conclusions from this work, and outline future research directions.

SYSTEM OVERVIEW

PowerAqua takes as input a question expressed in NL and returns all the answers it can derive from online semantic sources. In this section we give an overview of the system, by means of an illustrative example.

The overall architecture of PowerAqua is shown in Figure 1. Its Linguistic Component is invoked first; it analyzes a NL query, and translates it into a set of linguistic triples, called *Query-Triples (QTs)*, by identifying associations that relate terms together. For instance, the query "who plays in the rock group nirvana?" is translated into the QT <person / organization, plays, rock group nirvana>. This component is based on GATE [16], and it is essentially the same as the one included in AquaLog [9].

The QTs produced by the Linguistic Component are passed on to *PowerMap*, which is responsible for identifying the semantic sources that may answer the given query, and for producing initial element-level mappings between QT terms and entities in these sources. PowerMap has already been presented in [10] and here we only summarize the key aspects of its behavior, in the context of an example.

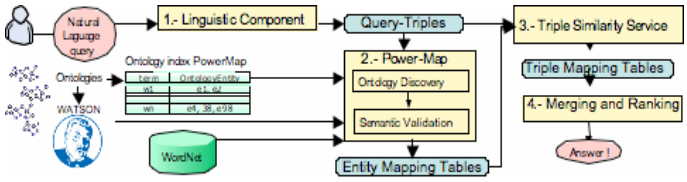

Figure 1: PowerAqua's components

Initially, PowerMap's Ontology Discovery sub-module identifies all semantic sources that are likely to describe QT terms. In this phase PowerMap maximizes recall in order to broaden the search space and bridge the gap between the user terminology and that of the various ontologies. This is achieved by searching for *approximate mappings* (lexical overlap) as well as *exact mappings* (lexical equality). These are jointly referred to as *equivalent mappings*. PowerMap uses both WordNet and the SW itself as sources of background knowledge to perform query expansion and to find lexically dissimilar (but semantically similar) matches – including synonyms, hypernyms and hyponyms. The output is a set of *Entity Mapping Tables (EMTs)*, where each table associates each QT term with a set of entities found on the SW (Table 1). For instance, the fifth row in Table 1 shows a mapping between Person (a term in the QT) and Musician (a concept in the Music Ontology) discovered using the hyponymy relation between Person and Musician, suggested by the TAP Ontology. For the example query, PowerMap is able to find a large number of candidate mappings in several ontologies. The first row in Table 1 indicates that no mappings were found for any of the compound terms in the query, although we were able to for the individual components (e.g., rock).

PowerMap uses the Watson Semantic Gateway as the entry point to the SW. Watson crawls and indexes online SW documents and provides an access point through its API. In addition PowerMap also provides a plug-in mechanism, which supports a common API to query ontologies residing on different repositories. Currently plug-ins are available for Sesame (www.openrdf.org) and Watson, however, it would be relatively easy to create plug-ins for other platforms as well.

Table 1. Partial view of the EMTs for QT <person/org, plays, rock group nirvana>

Rock group Nirvana, rock group, group nirvana	ø
Nirvana	***Music***: Nirvana (type: group); ***TAP***: MusicianNirvana (type: person); ***SWETO***: Nirvana Meratnia (type: researcher); ***KIM***: Eden (synonym); ***Spiritual***: Nirvana; ***Magnatune***: Passion of Nirvana (type: "track"), …
Rock	***Music***: rock (as a type of genre); ***SWETO***: Michael_Rock, Sibyl Rock, etc; ***ATO***: rock (as a type of substance), Ayers_Rock (as a place); …
Group	***Music ontology***: group, …
Person	***Music ontology***: musicians (as a hyponym of person according to TAP), ***TAP***: person, ***KIM***: person, ***Magnatune***: musicArtist (hyponym),…
Play	***KIM ontology***: sport (as synonym of "play")…

PowerMap's Semantic Validation component filters out the least promising mappings within an ontology by using a number of heuristics (equivalent mappings are preferred over hyper/hyponyms, redundant mappings within the same taxonomy are removed, etc). In addition, this component also attempts to generate WordNet synsets for all classes and individuals in the EMTs. In our example the system fails to find a valid synset for Nirvana, as the intended meaning is not in WordNet. It does, however, produce valid synsets for "rock" (e.g., synsets would be generated for all the entities listed in row 3 of Table 1 associated with the term "rock") interpreted as a "music genre" in both the music and TAP ontologies, and as a "stone (material consisting of aggregate minerals)" in ATO. While obviously only one of these interpretations is correct, at this stage the system is unable to disambiguate between the two. Nevertheless, other interpretations can already be ruled out at this stage. For instance, the association between the query term "rock" and class "stone", interpreted as a measure of weight, can be discarded because there is no intersection in WordNet between the intended synset and its synset in the ontology (therefore such association does not appear in Table 1).

PowerAqua's third step, the *Triple Similarity Service (TSS)*, takes as input the EMTs generated by PowerMap and the QTs, and returns a set of *Triple Mapping Tables (TMTs)*, which specify complete mappings between a set of Query Triples and the appropriate *Ontology Triples (OT)*, as shown in Table 2.

Table 2. The TMT for OTs in ontologies that match the QTs

<person / organization, play, Nirvana>	
SWETO	<Nirvana Meratnia, IS_A, person>
Magnatune	<MusicArtist (hyponym), maker (ad-hoc), Passion of Nirvana>
Music	<Musician (hyponym), has_members (ad-hoc), Nirvana>
TAP	<Person, hasMember (ad-hoc), MusicianNirvana>
<rock, ?, nirvana>; <group, ?, nirvana>	
Music	<Nirvana, has_genre, rock>; <nirvana, is-a, group>

Finally, the Merging and Ranking component generates the final answers from the returned Ontology Triples. In our example, one set of answers is produced by merging the OTs obtained from the Music and TAP ontologies. In particular, the instances on which both ontologies agree ("Dave Grohl", "Kurt Cobain", "Chad Channing" and "Chris Novoselic") are ranked higher. Then, the answer is augmented with additional instances from the music ontology ("Dan Peters", "Dave Foster", "Jason Everman", "Pat Smear", "Dale Crover" and "Aaron Burckhard", all former members of the band). The Music ontology also produces additional mappings for the compound term "rock group nirvana": <nirvana, has-genre, rock> <nirvana, is-a, group>. The answers from the SWETO and Magnatune ontologies are ranked last.

THE TRIPLE SIMILARITY SERVICE (TSS)

In the TSS the element level matches recorded in EMT's (QT terms to ontology elements) are assembled to produce triple level matches (entire QTs to OTs). The algorithm has been optimized towards finding the most precise ontological translations. Its design has been influenced by the following observations:

1) An ontology with a higher coverage of a QT is likely to lead to a better result (i.e., ontologies that cover entire triples and not just individual terms);
2) If no OTs can be found for a QT containing a compound term, potentially, relevant OTs may still be found for the individual elements of the compound. Therefore, the TSS is re-invoked with new QTs formed by splitting the compound term.
3) We observed that often the subject (QT_{i1}) of a QT refers to a less specific ontological entity than its object (QT_{i3}), which is frequently mapped to an individual (e.g: <Russian rivers, flow, Black Sea>). Therefore, splitting QT_{i1} (e.g. into "Russian" and "rivers") has less negative influence on the quality of the final ontology triples than splitting QT_{i3} (splitting QT_{i3} is more likely to introduce noise, e.g., "Black" and "Sea")
4) For queries translated into more than one QT, or in case of linguistic ambiguity, the TSS requires domain knowledge to solve modifier attachment and disambiguate how the triples link among themselves.

The TSS algorithm contains four steps that parallel these four observations and lead to decreasingly precise translations (but increased recall). The TSS executes the highest quality steps first and only uses inferior quality steps if no answer is found.

For each QT (QT_i), in **step S1**, the TSS inspects all ontologies that contain mappings for at least two of the terms in the QT. This coverage-centric criterion ensures that the algorithm focuses first on the ontologies most likely to address the domain of the query. The *Relation Similarity Service (RSS)*, which is explained in more detail later, is called for each ontology in order to find the concrete OTs, which match the input QT. If any of these OTs leads to an answer, then this is recorded in the TMT. After all potentially relevant ontologies have been inspected, if at least one answer has been derived the algorithm stops (lines 2-6). Otherwise, in **step S2**, the TSS increases recall by splitting the subject (QT_{i1}) and the TSS re-iterates for the resulting QTs (obtained after splitting the subject). At the end of this phase, if an answer has been obtained the algorithm stops, otherwise it continues (lines 14-19). In **step S3** we resort to splitting the object QT_{i3} and re-invoking the TSS for the resulting QTs. Finally, if none of the above strategies leads to an answer, recall is further improved in **step S4** by inspecting ontologies that cover the original (non split) QT_{i3}. In this case, the RSS is only called for this term (and for QT_{i2} if it has mappings) and it returns all OTs that partially cover the QT, if any (lines 19-26).

```
1: for all Query-Triple_i do
2:   for all O_j with cover(O_j ;QT_i) ≧ 2 do
3:     if RSS(QT_i1; QT_i2; QT_i3):hasAnswer() then
4:       recordInTMT()
5:     end if
6:   end for
7:   if TMT ≠ Empty then
8:     STOP
9:   else
10:    if QT_i1:isCompund() then
11:      Split(QT_i1)
12:      TSS(resultingTriples)
13:      STOP
14:    else
15:      if QT_i3:isCompund() then
16:        Split(QT_i3)
17:        TSS(resultingTriples)
18:        STOP
19:      else
20:        for all O_j with cover(O_j ; QT_i3) = 1 do
21:          if RSS(QT_i2; QT_i3):hasAnswer() then
22:            recordInTMT()
23:          else
24:            STOP - NoAnswers
25:          end if
26:        end for
27:      end if
28:    end if
29:  end if
30: end for
```

Relation Similarity Service

The Relation Similarity Service (RSS**)** is core to the TSS. The RSS inspects an ontology and identifies the OTs that are appropriate translations for the given QT, from which an answer can potentially be inferred. It preferentially uses exact mappings to obtain the OTs; otherwise equivalent mappings and synonyms are selected, leaving the use of hypernyms and hyponyms as the last choice.

The RSS can map a QT to either one OT (direct mapping) or to two OTs (indirect mappings). Depending on the type of their predicate, direct mappings can be "IS-A" (a subsumption relation) or "ad-hoc" (considering relations inherited from the superclasses). For the QT <*capitals, ?, Europe*> ("find me all capitals in Europe"), <*city,*

capitalCity, EuropeanNation> is an ad-hoc mapping while <*capital_city, has_capital_city, country*> <*country, has_member, Europe*> is an indirect mapping. "Ad-hoc" direct relationships between the arguments of the triple are analyzed before "IS-A"[3] unless the original question contains an IS-A relation (an indication that such a relation is expected). For example, for the query "which animals are reptiles" the answers are encoded as the subclasses of the class "reptile", which is a subclass (IS-A) of the class "animal". If such relations are not found then indirect relations are inspected (with only one mediating concept). In case the linguistic relation is mapped to an ontological property the matching and joining of triples is controlled by the domain and range information of the relation, if possible.

Examples

Table 3 contains a few examples of TSS outputs, which we will use below to illustrate the ways that answers are derived at each step of the algorithm.

Table 3: Examples of ontological translation for queries (TMTs)

Q1: Who works in the climaprediction project?
KMi ontology: <**person**, has-project-member, **climaprediction-net** (type: project)>, <**person**, has-contact-person, **climaprediction-net** (type: **project**)>
Q2: Which Russian rivers flow into the Black Sea?
RussiaB: <**river**, **flow-to**, **Black_Sea**> ***RussiaB***: <**river**, has_major_river, country> <country, has_political_fact, **russian**> ***KIM***: <**river**, partOf, entity> <entity, hasAlias, **Russian Soviet Federated Socialistic Republic**>
Q3: What pathologies produce hair loss?
Biomedical: <hyperthyroidism, hasSymptom, **Alopecia**>, <stress, hasSymptom, **Alopecia**>, <iron_deficiency, hasSymptom, **Alopecia**>

Let's consider query Q1 first, where the related QT is <*person/organization, work, climaprediction project*>. Because there exists a covering ontology with mappings for both "person" and "climaprediction" (as an instance of "project"), which contains valid ontological triples, an answer can be inferred at step S1.

Query Q2, whose corresponding QT is <*russian rivers, flow, Black Sea*>, is answered in step S2. The reason is that while there is a mapping for "Russian Rivers" in an ontology about restaurants, no valid OTs were produced in step S1 as there are no covering ontologies for both arguments. Therefore, the algorithm tries again by splitting the compound QT term, and consequently modifying the QTs into: <*Russian/rivers, flow, black sea*> <*Russian, ?, rivers*>. For these resultant QTs, in this second TSS iteration, a covering ontology containing the valid OTs that produce an answer is found by the RSS. The RSS analyzes the ontology relations to disambiguate which part of the compound "Russian" or "rivers" (or both) links to "flow into the black sea" in order to create valid OTs. If both compound parts produce valid OTs the different interpretations will be merged or ranked at a later stage.

Third, in the case of query Q3, with QT <*pathologies, produce, hair loss*>, there are several ontologies with mappings for both "pathologies" and "hair loss", which do not contain links between the entities. Therefore, steps S1, S2 and S3 fail to derive an answer using the covering ontologies, even when splitting "hair loss". In step S4, the only course of action left is to inspect all ontologies that contain at least one mapping for "hair loss" (QT_3). One such ontology is the biomedical ontology that contains the term "alopecia" (a medical term for "hair loss") and produces a set of potential answers (hyperthyroidism, stress, iron deficiency, etc.).

Efficiency of the TSS

The TSS and RSS are designed to avoid expensive computations in those scenarios where simple methods are able to yield solutions, in particular:

(1) The algorithm can re-iterate through the two different phases of collecting candidate ontological entities and then identifying relevant relationships between these entities in order to look only for the mappings needed in the first instance. This is useful in the case of compound terms, where the algorithm would look for mappings for terms composing the compound only if required (step 2 and 3).

(2) The time consuming process of analyzing indirect relationships in the RSS (i.e., relationships which require two triples to be joined) is only carried out in those cases where no direct relationship between candidate entities within the same ontology is found.

(3) In some cases the TSS algorithm can use semantic information to disambiguate how the triples link to each other (modifier attachment) and minimize the number of triple combinations to be analyzed in order to translate a query. E.g., in "which cities are located in the region of Sacramento?", whose corresponding QT are: <*cities, located, region*> <*region, ?, Sacramento*>, the TSS in step S1 finds an ontology stating that "SacramentoArea IS-A region". Using such semantic information, it transforms the two QTs into just one: <*cities, located, Sacramento Area*>.

Summing up, to avoid analyzing an unfeasibly large space of solutions, filtering heuristics are used, and the TSS tries to find an answer by augmenting the search space in each re-iteration until either an answer is found, or all the compounds (if any) are split, and all the ontologies with less coverage have been analyzed.

Known Issues

The TSS has a couple of "blindspots" where we know it returns noisy or incomplete answers in situations where the correct answers exist.

Noisy answers are typically produced when the knowledge encoded on the SW only covers part of the user query and the algorithm has to resort to ontologies with less coverage to generate an answer (step S4). In this case irrelevant results may be produced, as PowerAqua cannot fill in the missing information in order to fully understand (map) the query. E.g., in "who are the professors in Southampton?" for which there are no mappings for "professor" in the selected ontologies, the algorithm returns the "persons" who were

[3] The reason for this is that many ontologies misuse IS-A relations to model other types or relations (e.g. partonomy)

born and died in "Southampton" from the dbpedia ontology about people, and the "persons" who are part of the "University of Southampton", from the ISWC ontology. The latter set contains "Nigel Shadbolt", one of the answers we were looking for, but also false answers.

Incomplete answers can sometimes be produced because, to avoid performance problems, PowerAqua does not search for triples among relevant entities connected by long paths. E.g. for "find me cities in Europe," there is an exact mapping "Europe" in the KIM ontology, and, consequently, the non exact mappings in the same ontology like "eastern_europe" are discarded. As a result valid triples with more than 1 indirect relation are missed, such as: *<warsaw (city), locatedIn, Republic of Poland> <Republic of Poland, partOf, central_europe> <central_europe, locatedIn, Europe>*.

MERGING AND RANKING COMPONENT

The Merging and Ranking component is still work in progress, therefore we only provide a brief overview here. A side effect of the fact that PowerMap explores multiple knowledge sources is that, after the TSS, the query is frequently associated to more OTs from different ontologies, each OT generating an answer. Depending on the complexity of the query (i.e., the number of QTs it has been translated to by the linguistic component), as well as the way each QT was matched to OTs, these individual answers can a) be redundant or b) be part of a composite answer to the entire query or c) be alternative answers derived from different ontological interpretations of the QTs. Hence, different merging scenarios may arise depending on how the terms are linked across OTs. For instance, in "which Russian rivers flow into the Black Sea?", because each resultant OT only leads to partial answers, they need to be merged to generate a complete answer. This is achieved by intersecting the answers returned by "rivers in Russia" with those returned by "rivers that flow in the Black sea". Among other things, merging requires to identify similar entities across ontologies, e.g., "Russian Soviet Federated Socialistic Republic" and "Russia".

Additionally, whenever a set of answers is returned, it is important to be able to rank them. This component applies a range of ranking measures which take into account (in this order) a) criteria derived during merging (do the answers from different OTs denote similar interpretations- in order to obtain an unique set of answers?), b) criteria referring to the quality of the mapping between QTs and their OTs (how confident is PowerAqua about the mappings used to derive the answer?), and c) criteria referring to the popularity (frequency), of the answer (which answer is returned by the most ontologies?).

EVALUATION AND DISCUSSION

In our evaluation we focus on assessing PowerAqua's ability to derive answers to NL queries by relying on the information provided by multiple ontologies identified on the fly, rather than its linguistic coverage or its merging and ranking capabilities.

Experimental Setup

In this section we describe the evaluation criteria as well as the data sets used for our experiments. Our goal is to build a system that provides correct answers to a query, in a reasonable amount of time, by making use of at least one ontology, which provides the required information. In order for an answer to be correct, PowerAqua has to align the vocabularies of both the query and the answering ontologies. PowerAqua fails to give an answer if the knowledge is in the ontology(ies) but it can not find it. A conceptual failure (the knowledge is not in the ontology) is not considered as a PowerAqua failure because the ontology does not cover the information needed to map the user query or is not populated with the answers. Recall cannot be measured in this open scenario, as we don't know in advance how many ontologies can potentially answer the user's query. Therefore, the major evaluation criteria are *success*, in terms of getting (or not) at least one correct answer, and *speed*.

We tested our prototype on a collection of ontologies saved into online repositories and indexed by PowerMap. The collection includes high level ontologies, like ATO, TAP, SUMO and DOLCE, and very large ontologies, like SWETO, with around 800,000 entities and 1,600,000 relations. In total, we collected around 3GBs of data stored in 130 Sesame repositories (each repository containing one or more semantic sources, in total more than 700 documents). We preferred to use this big static data set rather than directly fetching ontologies from Watson [4] because: 1) the experiments are reproducible, 2) the question designers can easily browse the semantic sources in the collection to generate queries, and 3) the size and quality of ontologies is higher than those found in Watson, which includes a large number of small, lightweight ontologies (often not populated) and foaf files.

The questions used during the evaluation were selected as follows. We asked seven members of KMi, familiar with the SW, to generate factual questions[4] for the system that were covered by at least one ontology in our collection. We pointed out to our colleagues that the system is limited in handling questions that required temporal reasoning (e.g. today, last month, before 2004) and compositional semantic constructions (quantification, comparison, negation and number restrictions). As no 'quality control' was carried out on the questions, it was admissible for them to contain some minor grammatical errors. Also, we pointed out that PowerAqua is not a conversational system, it can't prompt users for extra information. We did not provide any additional detail to assess to what degree PowerAqua satisfies the expectation of the users. We collected a total of

[4] Factual queries formed with wh-terms (which, what, who, when, where) or commands (give, list, show, tell,..) vary in length and complexity:from simple queries, with adjunct structures or modifiers, to complex queries with relative sentences and conjuctions/disjunctions

69 questions. All the questions, answers and ontologies are accessible through the PowerAqua website[5].

Analysis of Results

PowerAqua successfully answered 48 (69.5%) out of 69 questions. This is a good result given i) the open nature of the question answering set up (hardly any constraints were imposed on the choice of the questions), and ii) the size and heterogeneity of the dataset. We analyzed the failures and divided them into the following categories depending on the component that led to the error (see Table 4):

*Linguistic analysis***.** A failure can be due to the query being out of the scope of the linguistic coverage (4 failures), or an incorrect annotation on the underlying GATE platform and grammars (e.g., annotating a verb as a noun) that leads to a misunderstanding of the query (1 failure). In total 5 queries, 7.2%, failed because of incorrect linguistic analysis. Extending the coverage of the linguistic grammars, currently only focus on factual queries, to queries that require a meaningful dependency structure of the sentence elements might solve such errors (e.g., Q17: "*Who are the people working in the same place of Paolo Bouquet?"*).

PowerMap. This component tries to maximize recall to broaden the search space. Accuracy is not crucial at this stage, as incorrect mappings will probably be discarded at a later stage by using the semantics of the ontologies. However, too many irrelevant mappings collected in this phase inevitably affect the overall performance of the system, therefore filtering heuristics are applied to achieve a compromise between performance and recall. In our evaluation 13 queries, 18.8% of the total failed either because of relevant mappings that could not be syntactically found (10 of them), or because they were found but later discarded by the filtering heuristics, as they were considered less likely to lead to the correct solution than others (3 of them). PowerMap needs semantic sources with enough human understandable labels to obtain high performance. In this evaluation there were no failures due to the WordNet-based semantic component to assess semantically sound mappings ([6] shows an earlier evaluation of this module).

Triple Similarity Service. A TSS related failure occurs when PowerMap correctly finds an ontology (and all mappings) containing the answer to the query, but the TSS fails to complete the matching process by locating the correct triples answering the query. This can be due to several reasons, such as incorrectly linking the terms into triples, or because of low quality or incomplete ontologies. In our evaluation, 3 of the queries, 4.3%, failed because of this component.

Merging component. This component worked as expected for the two queries that required merging across ontologies. Nevertheless, its integration with PowerAqua is still work in progress, and will be further evaluated later.

[5]http://technologies.kmi.open.ac.uk/poweraqua/ok-evaluation.html

Table 4. Overview of the evaluation results

Successful queries			**Total**
Q1, 3, 4, 6, 7, 8, 10, 11, 12, 13, 14, 16, 19, 20, 21, 22, 23, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 43, 44, 46, 48, 49, 52, 54, 55, 57, 58, 59, 61, 62, 64 and 65			48 (69.5%)
Failures			**Total**
Linguistic	Out of coverage	Wrong classification	5 (7.2%)
	Q2, 9, 17, and 51	Q24	
PowerMap	Fail to find the mappings	Filtering valid mappings	13 (18.8%)
	Q5, 18, 42, 47, 50, 53, 56, 63, 66, and 69	Q25, 60, and 67	
TSS	Q15, 45 and 68		3 (4.3%)

The average answering time for the successful queries was 15.39 seconds, the best time was 0.5 secs, while the worst was 79.2 secs. This worst case was for Q35: *which sea do the Russian rivers flow to?* (QT: <sea, flow, Russian rivers>) where the keyword "sea" and its lexically related words ("ocean") produces more than 300 mapping hits and "Russian rivers" produces two mappings: "ID_russianriverPub3493" (as an instance of a restaurant in Forestville) in an ontology about restaurants and as the literal "Russian river tule perch" (a species name from the *Embiotocidae* family) in the *FAO* ontology. The *FAO* ontology is selected by the TSS as the only covering ontology because it also has 40 approximate mappings for "sea" (*sea-bass, sea-cucumber, sea-pollution, sea-level, sea-sickness,* among others) and 1 for "flow" as a subclass of "situation". As the TSS fails to find OTs for <*sea, ?, flow*> and <*flow, ?, Russian rivers*> the linguistic relation "flow" is ignored and the algorithm searches for OTs by looking for "ad-hoc", "is-a" and indirect relationships (1 mediating concept) between any of the *FAO* hits. As no results are produced the TSS splits the compound and re-iterates to find other relevant ontologies for the new QTs: <*sea, flow, rivers/russian*> and <*Russian, ?, rivers*>. There are 18 covering ontologies for the former triple (considering either "rivers" or "Russia") and 8 for the later, from which only one, *russiaB*, contains the answer encoded in the OTs: <*sea, flow_to, river*> and <*river, has_river, country*>, <*country, has_political_fact, Russian*>. For this query 2305 SeRQL queries (the major bottleneck in the system) are needed.

In sum, we found that the tool was able to answer correctly well more than half of the proposed queries, with most of the failures being due to lexical level issues rather than the multi-ontology matching mechanism. The biggest group of PowerAqua failures are because relevant mappings could not be found. For instance, PowerMap fails to find the ontological entites "Spain" in Q56: "*List some Spanish islands"*, "CaliforniaRegion" in Q5: "*Give me all Californian dry wines"*, and "brain tumor SY NCI" in Q53: "*What kind of brain tumours are there?"*. Even when the relevant mappings were found, they were discarded by the PowerMap filtering heuristics. In fact, many of these errors are the consequence of poorly modeled or incomplete ontologies (i.e., containing redundant, disconnected terms). For example, for Q60, "*Which terrorist organization performed attacks in London?",* PowerMap correctly maps the linguistic term "London" to the instance representing the city, but this instance is not related to the entity "terrorist

organizations" leading to a TSS failure. PowerMap had indeed identified the mapping which would have led to the answer (the approximate literal "London, United Kingdom"), however this mapping was discarded because PowerMap considered it less likely to be correct than the exact instance mapping "London". If the literal "London, United Kingdom" would have had an ontological relation to the instance "London", or this one would have been ontologically connected with the instance "United Kingdom", PowerAqua would have found the answer, as it correctly does for the similar query Q61, "*Which are the main attacks that took place in the United Kingdom?*".

On the other hand, once the relevant mappings are found, the TSS only failed to translate three of the queries to ontology triples for different unexpected reasons. For example, for Q68, "W*hat drugs can be used for reducing fever?",* the answers are found in the subclasses of the single class "FeverReducingDrug", which is not connected to the mapped class "Fever".

RELATED WORK

Question answering has, until now, two main branches, depending on sources to generate an answer: closed domain QA over structured data (databases, knowledge bases etc.) and open domain QA over free text. With PowerAqua, we develop a new branch: open QA over structured data. Open domain QA over free text, stimulated since 1999 by the TREC QA track, has developed very sophisticated, syntactic, semantic and contextual processing. However, as stated in [7] the pitfalls of QA over free text, with respect to modern closed domain QA arise when a correct answer is unlikely to be available in one document, but must be assembled by aggregating answers from multiple ones, and when the questions are not trivial to interpret.

Concerning query interpretation, recent work on closed domain semantic search has addressed successfully key issues such as portability. In [7] users can pose complex NL queries to a large medical repository. A logical representation is constructed using a NL interface, where, instead of typing in text, all editing operations are defined directly on an underlying logical representation governed by a predefined ontology ensuring that no problem of interpretation arises. In [11] NL queries are translated into formal queries but the system is reliant on the use of gazetteers initialized for the domain ontology. Other systems with interesting approaches to query interpretation include SPARK, SemSearch and XXPloreKnow! [15][14][12] where keyword queries are translated into formal queries and K-Search [2], where a hybrid search using a mix of keywords-based and metadata-based strategies is used to recover documents.

With respect to open scenarios, there is growing interest in NL search engines that use semantic information to understand and disambiguate the intended meaning of the words in a query and how they are connected. The current trend in semantic search is to search for web pages based on the meaning of the words in the query, rather than just matching keywords and ranking pages by popularity (like Google or Yahoo). This class of systems include Powerset, Hakia and TrueKnowledge. For instance, Powerset tries to match the meaning of a query with the meaning of a sentence in Wikipedia. Powerset not only works on the query side of the search (converting the NL queries into database understandable queries, and then highlighting, the relevant passage of the document) but it also reads every word of every (Wikipedia) page to extract the semantic meaning (it also imports Freebase data). However, these systems, although they use semantics, can not be considered as QA systems that reuse the whole open SW because, like open QA over free text, they just return information from web pages, rather than constructing answers directly from structured data. Although PowerAqua is primarily built for QA on the SW, we have also investigated its use in enhancing keyword search technologies on the web by providing the semantic context to meaningfully extend an IR query. An initial evaluation study of such an approach is presented in [5].

CONCLUSIONS AND FUTURE WORK

Exploiting the large heterogeneous SW is essentially about discovering interesting connections between items in a meaningful way. PowerAqua provides a NL front end, which makes it possible to perform QA on the SW, hence supporting such a discovery process across multiple heterogeneous sources.

Our evaluation shows promising results, proving that it is feasible to answer questions with not just one but many ontologies selected on the fly in a reasonable time frame. Indeed, we obtained a success rate in answering questions of about 70% over a data set of 69 queries. The average answering time was 15.39 seconds, with some queries being answered within 0.5 seconds.

Our evaluation has highlighted an illustrative sample of problems for any generic algorithm that wishes to explore SW sources without making any a priori assumptions about them. Firstly, such algorithms are not only challenged because of the *scale of the SW* but more importantly because of its considerable *heterogeneity*, as entities are modeled at different levels of granularity and with different degrees of richness. Under this perspective, the TSS algorithm has performed well. Secondly, while the distinctive feature of our system is its openness to unlimited domains, its potential is overshadowed by the *sparseness of the knowledge* on the SW. In the evaluation of semantic search systems presented in [5], we found that the content collected by semantic search engines such as Swoogle and Watson only covers 20% of the query topics put forward in the TREC 9 and 2001 (http://trec.nist.gov) IR collections. To counter this sparseness, the PowerAqua algorithms maximize recall, which leads to a decrease in accuracy and an increase in execution time. Thirdly, in addition to the sparseness, most of the identified ontologies were *barely populated with instance data*. This caused PowerAqua's failure to retrieve a concrete answer in some cases even when a correct mapping

of the query was found in an ontology. A fourth aspect that hampered our system was the existence of many *low quality ontologies* which contained redundant, unrelated terms (causing the selection of incorrect mappings), presented unclear labels (thus hampering the system's ability to align query terms and ontology labels), or lacked relevant domain and range information (thus requiring more complex and time-consuming SeRQL queries). Finally, we note the as yet *suboptimal performance of ontology repositories and semantic search platforms* for querying large datasets. This limits the amount of information PowerAqua can acquire in a reasonable amount of time and hence hampers the system from considering all available information (especially that which needs to be identified by combining multiple knowledge elements).

Based on this analysis, we believe that the performance and the accuracy of the system are likely to improve as the SW grows and semantic search engines mature. As more information becomes available, it will become possible for PowerAqua to focus primarily on precision rather than recall, thus leading in principle to better accuracy and speed. Meanwhile, Watson is already addressing significant scalability issues to provide PowerAqua with efficient access to large amounts of new semantic data (e.g., the billion triple challenge dataset: http://challenge.semanticweb.org/). This will allow PowerAqua to find good mappings more easily, and therefore increase the precision in efficiently filtering noisy results thus improving accuracy and speed.

During our work we noted the lack of standard evaluation benchmarks for semantic search tools. This evaluation gave us an insight into PowerAqua's mapping capabilities and the range of questions it should be able to answer. In future work we plan to extend this evaluation with the independent Mooney[6] data sets used to evaluate single ontology based interfaces such as [1][13] and compare existing ontology based QA systems over a single source. Moreover, we would also like to evaluate the PowerAqua merging and ranking capabilities across ontologies with queries that can only be answered by a combination of semantic sources.

We are also experimenting with the integration of PowerAqua with standard search engines (such as Yahoo), by using the results from PowerAqua queries to automatically trigger contextualized searches in these search engines. The preliminary results have been promising (in particular, the answers retrieved from PowerAqua appear to improve the quality of the results returned by the search engines).

REFERENCES

[1] Bernstein, A., Kaufmann, E. (2006). GINO- A Guided Input Natural Language Ontology Editor. In the 5th International Semantic Web Conference. Athens, USA.

[2] Bhagdev, R., Chapman, S., Ciravegna, F., Lanfranchi, V. and Petrelli, D (2008). Hybrid Search: Effectively combining keywords and semantic searches. In the 5th European Semantic Web Conference . Tenerife.

[3] Cimiano, P., Haase, P., Heizmann, J. (2007). Porting Natural Language Interfaces between Domains -- An Experimental User Study with the ORAKEL System. In the International Conference on Intelligent User Interfaces.

[4] D'Aquin, M., Motta, E., Sabou, M., Angeletou, S., Gridinoc, L., Lopez, V., Guidi, D. (2008) Towards a new Generation of Semantic Web Applications. IEEE Intelligent Systems, 23 (3).

[5] Fernandez, M., Lopez, V., Motta, E., Sabou, M., Uren, V., Vallet, D., Castells, P. (2008) Semantic Search meets the Web. In the International Conference on Semantic Computing. Santa Clara, California.

[6] Gracia, J., Lopez, V., d'Aquin, M., Sabou, M., Motta, E., Mena, E. (2007). Solving Semantic Ambiguity to Improve Semantic Web based Ontology Matching. In the Ontology Matching Workshop at ISWC/ASWC. Busan, Korea.

[7] Hallett, C., Scott, D. and Power, R. (2007). Composing Questions through Conceptual Authoring. Computational Linguistics 33 (1)

[8] Lopez, V., Motta, E. and Uren, V. (2006). PowerAqua: Fishing the Semantic Web. In the 3th European Semantic Web Conference. Budva, Montenegro.

[9] Lopez, V., Motta, E., Uren, V. and Pasin, M. (2007). AquaLog: An ontology-driven Question Answering System for Semantic intranets, *Journal of Web Semantics*, 5 (2)

[10] Lopez, V., Sabou, M. and Motta, E. (2006). PowerMap: Mapping the Semantic Web on the Fly. In the 5th International Semantic Web Conference. Athens, USA.

[11] Tablan, V, Damljanovic, D., and Bontcheva, K. (2008). A Natural Language Query Interface to Structured Information. In the 5th European Semantic Web Conference.

[12] Tran, T., Cimiano, P., Rudolph, S., and Studer. R. (2007). Ontology-based interpretation of keywords for semantic search. In the 6th International Semantic Web Conference.

[13] Wang, C, Xiong, M., Zhou, Q., Yu, Y. (2007). PANTO: A portable Natural Language Interface to Ontologies. In the 4th European Semantic Web Conference. Innsbruck, Austria.

[14] Y. Lei, V. Uren, and E.Motta (2006). SemSearch: A Search Engine for the Semantic Web. In the 15th International Conference of Knowledge Engineering and Knowledge Management, EKAW. Podebrady, Czech Republic.

[15] Zhou, Q., Wang, C., Xiong, M., Wang, H., and Yu, Y. (2007). Spark: Adapting keyword query to semantic search. In the 6th International Semantic Web Conference. Korea.

[16] Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V. (2002). GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In Proc of the 40th Anniversary Meeting of the Association for Computational Linguistics (2002).

[6] http://www.cs.utexas.edu/users/ml/nldata.html