Towards Ontology-Based Question Answering in Vague Domains

Panos Alexopoulos*, Andrew Walker†, Jose Manuel Gomez Perez* and Manolis Wallace‡
*iSOCO S.A., Av. del Partenon 10, 28042, Madrid, Spain
Email: {palexopoulos,jmgomez}@isoco.com
†Department of Computing Science, Aberdeen University, UK
Email: andrew.walker.05@abdn.ac.uk
‡Department of Computer Science and Technology, University of Peloponnese, Tripolis, Greece
Email: wallace@uop.gr

***Abstract*—The rapid growth of the Semantic Web in the last years has led to the emergence of Ontology-Based Question Answering systems, namely systems that exploit the ontological structure of data in order to interpret and answer natural language questions. In this paper, we are interested in question answering scenarios where both semantic data and user questions can be characterized by vagueness and we propose a novel framework that is able to handle this vagueness by means of fuzzy ontologies.**

I. Introduction

Question Answering systems (QA) have been investigated for a long time by the communities of Information Retrieval, Artificial Intelligence and Databases, as a way to enable users to ask questions in Natural Language (NL), using their own terminology, and receive concise answers [1]. As sources for these answers have been mainly considered document collections [2], databases [3] and the Web [4].

Nevertheless, with the recent rapid growth of the Semantic Web and the availability of large-scale ontologies and linked data [5], there has been a growing interest in developing QA systems that can answer questions over such data. One reason for that is that user-friendly interfaces, which can support end users in querying and exploring this new type of information space are crucial for the realization of the Semantic Web vision [6]. Another reason is that the exploitation of the expressive power and reasoning capabilities of semantic data can lead to much more effective interpretation (and thus answering) of user questions.

In an ontology-based QA scenario, data is typically stored as a graph in an ontology with each node connected to others by various relations. This is a highly structured format where the semantic details are explicitly defined within the context of the ontology and typically also with references to external ontologies. Natural language queries however are structured only insofar as one word comes before another, and the semantic content is often unclear. Thus, the challenge is the accurate and efficient interpretation of these queries as they pertain to the knowledge contained in the relevant ontologies.

Towards tackling this challenge, several question answering systems for semantic data have been proposed in the past, including Aqualog [7], Power- Aqua [8], NLP-Reduce [6], Pythia [9], and FREyA [10]. Nevertheless, none of these systems is currently designed to answer questions that contain **vague terms**.

More specifically, vagueness, typically manifested by terms and concepts like *Expensive, Strong, Near, Modern* etc., is a quite common phenomenon in human knowledge and communication and it is related to our inability to precisely determine the extensions of such concepts in certain domains and contexts. That is because vague concepts have typically fuzzy boundaries that do not allow for a sharp distinction between the entities that fall within the extension of these concepts and those which do not [11] [12]. This is not usually a problem in individual human reasoning, but it may become one, when multiple agents (human or machines) need to agree on the exact meaning of such terms and reason with them.

In the context of an ontology-based question answering scenario, vagueness may appear both in user queries and the ontology. For example, a user may ask *"Which European countries have* ***long*** *rivers?"* with the ontology containing the exact length of each river. Conversely, a user may ask *"Which restaurants cost not more than 20 euros per person?"* with the restaurant price attribute in the ontology taking as values one of the vague terms *"cheap", "moderate", "expensive"*. And, of course, the user may ask *"Which restaurants are relatively cheap?"* with the price attribute in the ontology taking again only as values one of the above terms.

The challenge in all these cases for a QA system is to express the question's content in a compatible to the ontology way so that it can answer it. This means that the term "long" in the above example should be somehow expressed with the help of actual length values if it is to be matched with the ontology's values. Similarly, the "20 euros" price constraint in the restaurant request, or even that of "relatively cheap" needs to be expressed by means of the ontology's available three price characterization vague terms.

In this context, we propose in this paper a novel ontology-based question answering architecture that is able to tackle scenarios like the above and enable effective question answering in vague domains. The way it does that is by using **Fuzzy Ontologies** to formally represent and interpret the domain's vagueness and by implementing a modular pipeline that takes as input (vague) queries expressed in natural language and returns answers drawn from the fuzzy ontology. A key feature of the system, as we will explain subsequent sections, is that the answers it provides are accompanied by the degree to which they satisfy the user's vague query.

The structure of the rest of the paper is as follows. In the next section we define the problem we wish to tackle more rigorously and we provide an overview of our approach. In section III we describe the way domain knowledge is needs to be organized/enriched in order to facilitate vague question answering while in IV we describe the components of a software pipeline that performs such answering. Finally, in section V we discuss summarize our approach and discuss future directions.

II. Problem Definition and Approach

In traditional ontology-based QA systems, the main task to be tackled is the transformation of the natural language query into a set of required RDF[1] triples, typically expressed in the SPARQL[2] query language and in accordance to the system's ontology. Achieving such a transformation, the answering of the query is simply performed by executing it against the ontology and showing the results to the user. Thus, for example, the question *"Give me the birthdays of all actors of the television show Charmed"* against the DBPedia[3] ontology would be transformed into the SPARQL query:

```
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX res:<http://dbpedia.org/resource/>
SELECT DISTINCT ?date WHERE
{
    res:Charmed dbo:starring ?actor .
    ?actor dbo:birthDate ?date .
}
```

Nevertheless, this process is insufficient in the following cases:

1) When the query refers in a vague way to an ontology element that is not vaguely represented in the ontology. For example, assume that the user asks for *"movies where young actors star"* but the knowledge base merely contains the exact ages of the actors. In such a case the transformed SPARQL query will not be able to retrieve any results, as it won't be able to match the vague query term with the non-vague ontology content.
2) When the query refers in a non-vague way to an ontology element that is vaguely represented. For example, assume that the user asks for *"restaurants with an average price of 20 euros"* but the knowledge base merely contains 3 possible values for the price attribute, namely *"cheap"*, *"moderate"* and *"expensive"*. Again, the SPARQL query won't be able to match the non-vague query value with the ontology's vague one.
3) The query refers in a vague way to a vague ontology element. In this case, the SPARQL will match exact correspondences of vague values (e.g. *"cheap"* with *"cheap"*) but won't do the same for non-exact but close correspondences (e.g. *"cheap"* with *"relatively cheap"*).

Thus, the problem we wish to tackle can be defined as follows: *"Given an ontology-based question answering scenario where both ontology elements and user queries may be expressed with vague terms, how can we effectively interpret the users' queries and provide them with accurate answers?"*

Our approach towards tackling this question has two dimensions. The first is the adoption of the fuzzy ontology paradigm as a way to formally represent the meaning of the domain's vague terms and concepts within the system's ontology. As we discussed in D4.3, fuzzy Ontologies [13] are extensions of classical ontologies that, based on principles of Fuzzy Set Theory [14], allow the assignment of truth degrees to vague ontological elements, in an effort to quantify their vagueness. Thus, whereas in a traditional ontology one would claim that '*'A restaurant price of 100 Euros is expensive"*, in a fuzzy ontology one would claim that *"A restaurant price of 100 Euros is expensive to a degree of 0.7"*. In fact, as we will explain in the next section, a fuzzy ontology can define a more complete mapping between the space of potential restaurant price values and the degrees to which these are considered "expensive", "cheap", etc., as shown in figure 1.

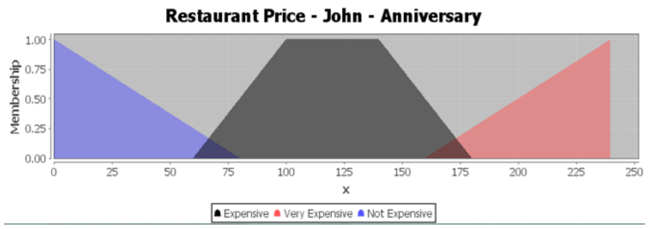

Figure 1. Fuzzy Datatype Example

The availability of such mappings (and other fuzzy definitions) for the domain's vague terms practically enables a system to deal with the three vagueness-related question answering cases described above. How this may be exactly achieved is the second dimension of our approach, namely a **Vague Question Answering Pipeline** that defines a set of components for interpreting and answering vagueness-

[1]http://www.w3.org/RDF/
[2]http://www.w3.org/TR/sparql11-query/
[3]http://dbpedia.org

related natural language queries, based on domain fuzzy ontologies. In what follows we provide an overview of each component of the pipeline and explain how it exactly works.

III. Knowledge Base Organization

When considering the problem of vague question answering, we assume the existence of an ontological knowledge base describing one or more domains. This ontology consists of concepts, relations, attributes, datatypes and instances, a subset of which may be vague. Along with the above knowledge base, we also assume a **fuzzy ontology** that describes the vague aspects of the ontology. From a conceptual point of view, such an ontology consists of:

- **Fuzzy Concepts**, namely vague concepts whose instances may belong to them to certain degrees (e.g. *Goal X is an instance of StrategicGoal to a degree of 0.8*).
- **Fuzzy Relations/Attributes**, namely vague relations and attributes that may link concept instances to other instances or literal values to certain degrees (e.g. *John is expert at Knowledge Management to a degree of 0.5*).
- **Fuzzy Concept Membership Axioms**, namely axioms that relate an instance to the concept(s) it belongs to a certain degree.
- **Fuzzy Property Value Axioms**, namely axioms that relate two instances to a certain degree.
- **Fuzzy Datatypes**, namely sets of vague terms which may be used within the ontology as attribute values (e.g. attribute *experience* mentioned above). In a fuzzy datatype each term is mapped to a fuzzy set that assigns to each of the datatype's potential exact values a fuzzy degree indicating the extent to which the exact value and the vague term express the same thing (e.g. *A consultant with 5 years of experiences is considered experienced to a degree of 0.6*)

The construction of the fuzzy ontology and the definition of its degrees has already been discussed in D4.3. Additional tasks we perform here for enabling vague QA are the following:

- We define fuzzy datatypes even for attributes that actually take non-vague values in the ontology as long as these may be expressed by means of vague terms. The reason for that is that these terms may still be used within user questions, so the system needs to be aware of their correspondence to actual values.
- For attributes that take non-vague values but are associated to a fuzzy datatype, we express their relevant value axioms by using the datatype's terms and fuzzy degrees. This means, for example, that the axiom *"Restaurant A's average price is 20 Euros"* becomes *"Restaurant A's average price is cheap to a degree d_1"* and *"Restaurant A's average price is moderate to a degree d_2"*, d_1, d_2 being derived from the fuzzy membership function of the fuzzy datatype of figure 1.
- For attributes that take vague values and are associated to a fuzzy datatype, we expand their relevant value axioms so as to include the other terms of the datatype. This means, for example, that the axiom *"Restaurant A is cheap"* is expanded with the axioms *"Restaurant A is moderate to a degree of d_1"* and *"Restaurant A is expensive to a degree of d_2"*. The degrees d_1 and d_2" are derived from computing the fuzzy subsumption [15] between the respective fuzzy sets of the terms.

It should be noted that all the above generated fuzzy axioms do not replace the original ones in our main knowledge base; they are merely used by the fuzzy query engine to enable the system to answer queries that refers in a vague way to ontology attribute values that are not vaguely expressed in the original ontology.

IV. Vague Question Answering Pipeline

The QA system operates as a pipeline, incrementally analyzing and augmenting the interpretation of the natural language input, based on an ontological knowledge base. There are two main stages, as shown in Figure 2; the first involves a linguistic and semantic analysis of the question to determine what information is being requested and with what restrictions while the second converts the interpretation into a formal query that can retrieve the answer from the knowledge base.

A. Question Linguistic and Semantic Analysis

This stage involves the transformation of an natural language question into a set of ontological statements that express the required information. This process typically starts with the tokenization and parsing of the question's text. Tokenisation is the segmentation of text into parts that represent distinct units to be further interpreted while grammatical parsing allows for more in-depth analysis of the structure of the query by identifying the phrasal structure of the sentence (i.e. verb-phrases, nouns, conjunctions etc) and the semantic dependencies between lexical entities. Syntactic parses are helpful for identifying the tokens that are likely to have mappings to the ontology while dependency parses provide a direct understanding of the subject/object relations of a sentence, giving a preliminary glance at the high-level structure of the query itself.

The process continues with entity mapping, namely the tagging of the question's tokens with the ontological entities they refer to. Typically, this is done by lexically matching the labels of ontological concepts to the query tokens, followed by the application of some entity disambiguation algorithm [] to resolve ambiguous entities (e.g., in the question *"How many goals has Pedro scored for Barcelona"*, the entity "Pedro" is ambiguous as there are many soccer players with that name).

Once entities are identified and disambiguated it is necessary to determine how they are related in the target

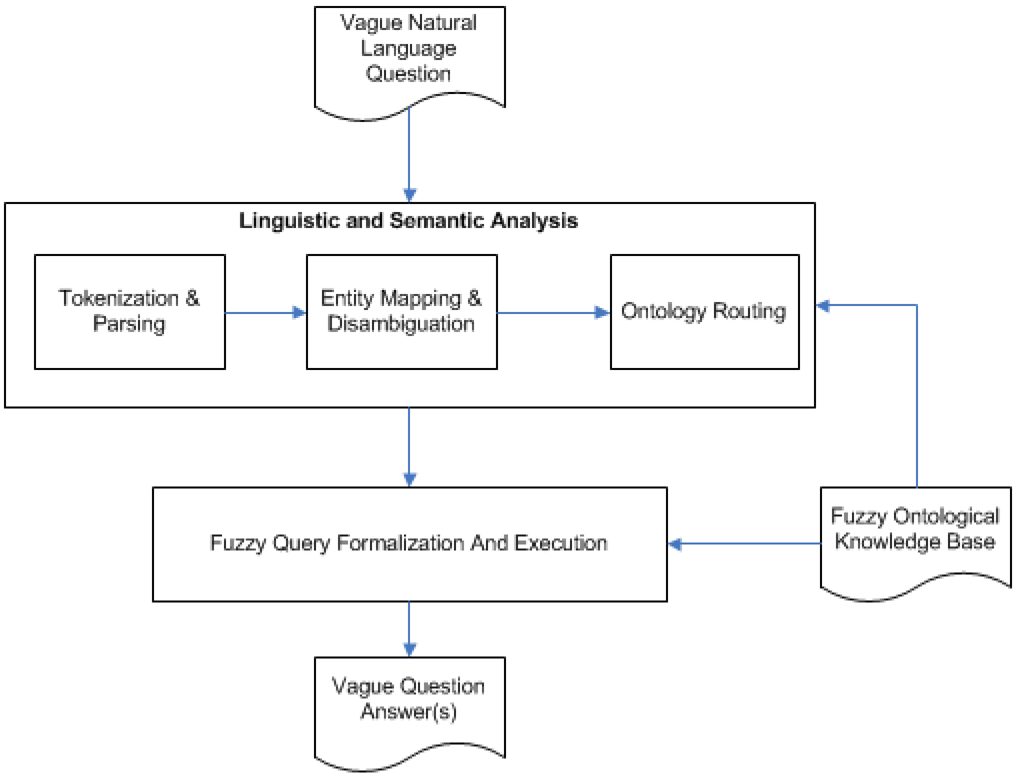
Vague Natural Language Question
Linguistic and Semantic Analysis
Tokenization & Parsing
Entity Mapping & Disambiguation
Ontology Routing
Fuzzy Query Formalization And Execution
Fuzzy Ontological Knowledge Base
Vague Question Answer(s)

Figure 2. Vague Question Answering Pipeline

knowledge-base. Oftentimes there will not be a direct relationship between two concepts identified in a natural language query, but instead have intermediary concepts (e.g., bands and songs may be related via albums). Thus, ontology routing determines the possible relation paths that link the identified entities of the question within the knowledge base. In small ontologies such cases may be limited but in larger ones there may be several possible "routes" between two entities, and which one to use must be determined. Normally, the "shortest route" is assumed to be optimal, but care must be made when other tokens in the query reflect entities and/or relations found only along longer routes.

B. Query Fuzzification and Execution

The fuzzification stage is the conversion of the derived triples to a formal fuzzy query language. For the purposes of this study, we consider the f-SPARQL query language [16] which allows to query ontologies that are formalized in the f-DL-Lite language [17]. The latter is a fuzzy extension of DL-Lite [ref] that allows the definition of fuzzy assertions of the forms $B(a) \geq n$, $R(a,b) \geq n$, where B is a class, R is a property (relation or attribute), a and b are individuals and n is a real number in the range [0, 1]. These assertions can also be expressed in RDF/XML syntax, using the serialization proposed in [26]. For example, stating that "Elvis is a FamousPerson to degree at-least 0.9" is specified as follows.

```
<FamousPerson rdf:about="Elvis"
owlx:ineqType=">=" owlx:degree="0.7"/>
```

f-SPARQL supports two types of queries:

1) Threshold queries: Ask for entities that satisfy fuzzy axioms to certain degrees.
2) General Fuzzy Queries: Ask for the degrees to which given axioms are true.

Given that, the conversion of the query triples to the above query language is done as follows. If the user's query involves a fuzzy concept or fuzzy relation (e.g. *"Give me all our competitors"*), then we form the query as a General Fuzzy Query:

```
#GFCQ:SEM=FUZZYTHRESHOLD# SELECT ?x WHERE
{ ?x rdf:type Competitor. #DG# 1.0 }
```

The execution of this query against the fuzzy ontology will return all the competitors with their actual degrees. These degrees make the answer more complete as they reflect the extent to which each result entity is actually considered a competitor in the knowledge base.

On the other hand, if the user's query involves an attribute associated to a fuzzy datatype (e.g., the price of a restaurant) then there are three possibilities:

1) The attribute's value in the knowledge base is crisp but vague in the user's query.
2) The attribute's value is vague in both the knowledge

base and the user's query.

3) The attribute's value in the knowledge base is vague but crisp in the user's query.

In the first case, we form again a General Fuzzy Query like the following:

```
#GFCQ:SEM=FUZZYTHRESHOLD#
SELECT ?x
WHERE {?x price  Expensive . #DG# 1.0}
```

This query is executed on the a priori fuzzified facts and retrieves the relevant entities along with their degrees. In the second case, we form the same f-SPARQL query as in the first case, but this time this will be executed on the expanded fuzzified vague terms. This ensures that different vague terms will match (even if only to a certain degree) as long as they have some overlap in their meanings, thus increasing the recall of the system.

Finally, in the third case, we use the fuzzy datatype to express the query's crisp value by means of its corresponding vague terms. Thus, for example, the query *"Give me all our restaurants with a price of 20 euros"* will be transformed into a threshold query as follows:

```
#GFCQ:SEM=FUZZYTHRESHOLD# SELECT ?x WHERE
  ?x price  cheap . #DG# 0.6
  ?x price  moderate . #DG# 0.4
}
```

Executing this query will retrieve all restaurants that are average to a degree of 0.4 or cheap to a degree of 0.6. Thus here not only the matching between the crisp value and the vague term is facilitated, but also the returned results are accompanied by a degree that tells the user how much they fit his/her question.

V. Conclusions & Future Work

In this paper, we proposed a novel framework for performing ontology-based question answering in scenarios where both domain knowledge and user questions are characterized by vagueness. The framework manages to tackle the three vagueness-related problematic cases that we identified in section II by utilizing fuzzy ontologies as complementary domain knowledge, along with state-of-the-art fuzzy ontology querying and reasoning technologies. It should be noticed that the quality of the answers provided by the system is directly proportionate to the accuracy of the degrees in the fuzzy knowledge base; nevertheless this is a matter of effective fuzzy ontology engineering and falls outside the scope of this paper.

An aspect of our approach to be addressed in the future is the assessment of the QA pipeline's efficiency, i.e., the execution of the system against fuzzy knowledge bases of increasing size and complexity and the measurement of the average time it takes to interpret and answer a question. Moreover, we intend to integrate our QA architecture with a fuzzy knowledge acquisition framework that we have developed in the past [18] and evaluate the overall system's effectiveness.

Acknowledgments

This work was supported by the European Commission under contract FP7-287615 PARLANCE as well as by the People Programme (Marie Curie Actions) of the European Unions 7th Framework Programme P7/2007-2013 under REA grant agreement no 286348.

References

[1] L. Hirschman and R. Gaizauskas, "Natural Language Question Answering: The View from Here," *Natural Language Engineering*, vol. 7, no. 4, pp. 275–300, Dec. 2001.

[2] E. Breck, J. Burger, D. House, M. Light, and I. Mani, "Question Answering from Large Document Collections," in *Proceedings of the AAAI Fall Symposium on Question Answering Systems*, 1999.

[3] A.-M. Popescu, O. Etzioni, and H. Kautz, "Towards a Theory of Natural Language Interfaces to Databases," in *Proceedings of the 8th International Conference on Intelligent User Interfaces*, ser. IUI '03. New York, NY, USA: ACM, 2003, pp. 149–157.

[4] Kwok, Cody and Etzioni, Oren and Weld, Daniel S., "Scaling question answering to the web," *ACM Trans. Inf. Syst.*, vol. 19, no. 3, pp. 242–262, Jul. 2001.

[5] C. Bizer, T. Heath, and T. Berners-Lee, "Linked Data - The Story so Far," *Int. J. Semantic Web Inf. Syst.*, vol. 5, no. 3, p. 122, 2009.

[6] E. Kaufmann and A. Bernstein, "How Useful are Natural Language Interfaces to the Semantic Web for Casual End-users?" in *Proceedings of the 6th International The Semantic Web and 2Nd Asian Conference on Asian Semantic Web Conference*, ser. ISWC'07/ASWC'07, 2007, pp. 281–294.

[7] V. Lopez, V. Uren, E. Motta, and M. Pasin, "AquaLog: An Ontology-Driven Question Answering System for Organizational Semantic Intranets," *Web Semant.*, vol. 5, no. 2, pp. 72–105, Jun. 2007.

[8] V. Lopez, M. Fernandez, E. Motta, and N. Stieler, "PowerAqua: Supporting Users in Querying and Exploring the Semantic Web," *Semantic Web - Interoperability, Usability, Applicability, pre-press*, 2011.

[9] C. Unger and P. Cimiano, "Pythia: Compositional Meaning Construction for Ontology-Based Question Answering on the Semantic Web," in *Proceedings of the 16th international conference on Natural language processing and information systems*, ser. NLDB'11, 2011, pp. 153–160.

[10] D. Damljanovic, M. Agatonovic, and H. Cunningham, "FREyA: An Interactive Way of Querying Linked Data Using Natural Language," in *Proceedings of the 8th international conference on The Semantic Web*, ser. ESWC'11. Berlin, Heidelberg: Springer-Verlag, 2012, pp. 125–138.

[11] D. Hyde, *Vagueness, Logic and Ontology*. Ashgate New Critical Thinking in Philosophy, 2008.

[12] S. Shapiro, *Vagueness in Context*. Oxford University Press, 2006.

[13] P. Alexopoulos, M. Wallace, K. Kafentzis, and D. Askounis, “IKARUS-Onto: A Methodology to Develop Fuzzy Ontologies from Crisp Ones,” *Knowledge Information Systems*, vol. 32, no. 3, pp. 667–695, Sep. 2012.

[14] G. Klir and B. Yuan, *Fuzzy Sets and Fuzzy Logic, Theory and Applications*. Prentice Hall, 1995.

[15] G. Stoilos, U. Straccia, G. B. Stamou, and J. Z. Pan, “General Concept Inclusions in Fuzzy Description Logics,” in *ECAI*, 2006, pp. 457–461.

[16] J. Z. Pan, G. Stamou, G. Stoilos, S. Taylor, and E. Thomas, “Scalable Querying Services over Fuzzy Ontologies,” in *the Proc. of the 17th International World Wide Web Conference (WWW2008)*, 2008.

[17] U. Straccia, “Answering Vague Queries in Fuzzy DL-LITE,” in *Proceedings of the 11th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, (IPMU-06)*, 2006, pp. 2238–2245.

[18] P. Alexopoulos and J. M. Gomez-Perez, “Interactive Acquisition of Fuzzy Ontological Knowledge in Dialogue Systems.” in *K-CAP*, V. R. Benjamins, M. d’Aquin, and A. Gordon, Eds. ACM, 2013, pp. 133–134.