A Knowledge-Based Approach to Ontologies Data Integration

Tech Report kmi-04-17

Maria Vargas-Vera and Enrico Motta

The Open University

A Knowledge-Based Approach to Ontologies Data Integration

Maria Vargas-Vera and Enrico Motta
Knowledge Media Institute (KMi),
The Open University, Walton Hall,
Milton Keynes, MK7 6AA,
United Kingdom
{m.vargas-vera@open.ac.uk}

Abstract

This paper describes a proposal of multiple ontology data integration system for a question answering framework called AQUA. We propose an approach for mediating between a given query and a set of resources. This method is based on a Meta-ontology (which contains contents of each individual sources) and our similarity algorithm based on analysis of neighborhood of classes. We argue that AQUA can perform mappings between queries and an ontological space by using a mediator agent based on a Meta-ontology and our similarity algorithm.

Keywords: Ontologies, Data Integration, Meta-ontology, Similarity

1. Introduction

This paper focuses on the problem of incorporating an integration system to AQUA (Vargas-Vera and Motta 2004; Vargas-Vera et al 2003a,b), with a mediator agent. AQUA was developed at the Open University in England, United Kingdom. AQUA joins two different paradigms of closed-domain and open domain question answering into a single framework. One of the main characteristics of AQUA is that it uses knowledge (encoded in ontologies). This knowledge is used in several steps of the question answering process like query reformulation. AQUA translates user query written in English to first order logic (FOL). Currently, AQUA when is used as closed-domain question answering uses a single populated ontology. Therefore, we aimed to extend the architecture in order to handle multiple-ontologies. To achieve this goal we needed a mediator agent which could perform mappings between terms in queries and ontological relations.

Our solution to the ontology data integration problem was the use a Meta-ontology coupled with our similarity algorithm (described in section 3.1). This Meta-ontology contains information about relations of each resource. This is not a limitation of the system because new meta-information (a new resource) can be added incrementally. The process of answering a query is divided in four steps

1. A query planner which takes a given a query written as first order logic (FOL) and divides into sub-queries.
2. A selection procedure provides with a subset of ontologies relevant to the query.
3. The query-satisfaction algorithm answers the using standard techniques in question answering already implemented in AQUA Vargas-Vera and Motta 2004; Vargas-Vera et al 2003a,b).

The main contributions of our paper can be summarized as 1) identifying the information which need to be kept in the meta-ontology and identification of fragments of information which are relevant to a given query. 2) algorithms for generating relevant ontologies and similarity algorithm based on neighborhood of classes.

The paper is organized as follows:
Section 2 presents briefly the current architecture of AQUA which uses a single ontology. Section 3 introduces the problem of data integration and outlines our suggested algorithm for generating relevant ontologies and it also introduces our similarity algorithm embedded in AQUA. Section 4 describes related work. Finally, Section 5 gives conclusions and directions for future work.

2. AQUA process model

The AQUA process model generalizes other approaches by providing a uniform framework which integrates NLP, Logic, Ontologies and information retrieval. Within this work we have focused on creating a process model for the AQUA system. Figure 1 shows the architecture of our AQUA system.

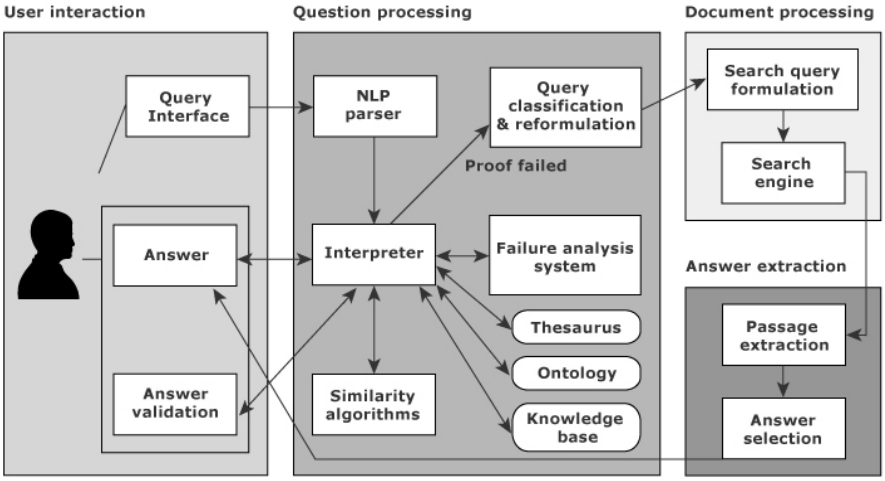

Figure 1. The AQUA process model

In the process model there are four phases: **user interaction, question processing, document processing and answer extraction.**

1. **User interaction**. The user inputs the question and validates the answer (indicates whether it is correct or not). This phase uses the following components:
   - *Query interface*. The user inputs a question (in English) using the user interface -a simple dialogue box. The user can reformulate the query if the answer is not satisfactory. *The answers consist of* a ranked set of answers is presented to the user.
   - *Answer validation*. The user gives feedback to AQUA by indicating agreement or disagreement with the answer.

2. **Question processing**. Question processing is performed in order to understand the question asked by the user. This 'understanding' of the question requires several steps

such as parsing the question, representation of the question and classification. The question processing phase uses the following components:

- *NLP parser*. This segments the sentence into subject, verb, prepositional phrases, adjectives and objects. The output of this module is the logic representation of the query.
- *Interpreter.* This finds a logical proof of the query over the knowledge base using unification and resolution algorithms.
- *WordNet/Thesaurus.* AQUA's lexical resource.
- *Ontology*. Currently AQUA works with a single ontology – the AKT reference ontology which contains people, organizations, research areas, projects, publications, technologies and events.
- *Failure-analysis system*. This analyzes the failure of given question and gives an explanation of why the query failed. Then the user can provide new information for the pending proof and the proof can be re-started. This process can be repeated as needed.
- *Question classification & reformulation*. This classifies questions as belonging to any of the types supported in AQUA, (what, who, when, which, why and where). This classification is only performed if the proof failed. AQUA then tries to use an information retrieval approach. This means that AQUA has to perform document processing and answer extraction phases.

3. **Document Processing**. A set of documents are selected and a set of paragraphs are extracted. This relies on the identification of the focus[1] of the question. Document processing consists of two components:
   - *Search query formulation*. This transforms the original question, **Q** into a new question **Q'**, using transformation rules. Synonymous words can be used, punctuation symbols are removed, and words are stemmed.
   - *Search engine*. This searches the web for a set of documents using a set of keywords.

4. **Answer processing**. In this phase answers are extracted from passages and given a score, using the two components:
   - *Passage selection*. This extracts passages from the set of documents likely to have the answer.
   - *Answer selection*. This clusters answers, scores answers (using a voting model), and lastly obtains a final ballot.

We want to remind the reader that phases 1, and 2 deals with the AQUA part as closed-domain question answering. Whilst 1,2, 3 and 4 deals with AQUA as open-domain question answering..

3. Data integration

Integration of different sources has been one of the fundamental problems faced in the Database community in the last decades (Batini et al., 1986) . The goal of a data integration system is to provide a uniform interface between various data sources (Levy 2000). As a result of a data integration users obtain some benefits such as

[1] Focus is a word or a sequence of words which defines the question and disambiguates it in the sense that it indicates what the question is looking for.

- users do not need to find the relevant sources to a given query,
- to interact with each source in isolation,
- to select and cleaning and
- to combine data from multiple sources in order to answer a given query.

Description Logics have been used in data integration projects to represent the conceptual level (Catarci and Lenzerini, 1993; Arens et al 1993; Goasdoue et al 2000). In our work we have extended Halevy et al. (Halevy et al 1996) work on data integration for Databases. The design of a data integration system is a very complex task which comprises several aspects. In this paper we only concentrate in one part of the mediator agent (the selection of sources). This selection procedure has as a target goal to find relevant fragments to a given query using different resources. It can be seen as a way to trim the ontological search space.

We propose a **generate-relevant-ontologies** procedure which works for ontologies. The **generate-relevant-ontologies** procedure relies on the fact that there is a meta description of the different sources (Meta-Ontology). Once that the relevant sources are identified the answer to the question can be performed using the selected sources. The Meta-ontology contains a formal description of the concepts, relationships between concepts A good feature is that such descriptions are independent of any system consideration. Our meta-ontology **Meta-O** consists of the union of each individual description of each ontology i.e.

Meta-O = $\boldsymbol{O_1\ U\ \ldots\ U\ O_n}$

where Meta-O is the global schema and $\mathbf{O_i}$ is one the description of ontology **i** and

$\boldsymbol{O_j}$ = { $\mathbf{R_{1j}, \ldots, R_{nj}}$ }

where $\mathbf{R_{ij}}$ means the relation **i** in the ontology **j.**

Each ontology description consists of a set of relations written in FOL with types and arities. These descriptions can be defined in a formal language which can be FOL or DL (description Logic).

The procedure generate-relevant-ontologies can be defined as follows:

Procedure **generate-relevant-ontologies**(**O,Q**)
/* **O** is the set of ontologies
Q is a query in conjuctive form */

1. Query planner decompose query in subqueries

$$Q\ (X_1, \ldots, X_n) = Q_1(Z_1)\ \ U\ \ Q_2(Z_2)\ \ U, \ldots, U\ Q_n(Z_n)$$

Where each Q_i can be defined as a predicate $Q_i(W_1, \ldots W_n) = \exists\ X_1, \ldots, X_n\ \omega\ (X_1, \ldots, X_n)$

2. $S = \Theta$; i=1;

3 For each Q_j do

4. For each $\eta_i (Y_1,...,Y_n) \in$ Meta-O do i
Begin
Compute *Sim($\omega(X_1, ..., X_n), \eta(Y_1,...,Y_n)$) using our ontology-based similarity algorithm given in section 3.1*

If $\exists$ β a mapping between $\omega(X_1, ..., X_n$ *and* , $\eta(Y_1,...,Y_n)$ *then*

$\Pi = \Pi$ U {($\eta(Y_1,...,Y_n)$, O_i)} were relation $\eta(Y_1,...,Y_n)$ *is on ontology* O_i
Else

End do i

5. $S = S \cup \Pi$
End do j
6. Return S

7. If $\boldsymbol{S}$ is $\boldsymbol{\Theta}$ print "there is not set of ontologies relevant to the query''
else evaluate query using the subset of ontologies in $\boldsymbol{S}$

8. End **generate-relevant ontologies**

3.1 Similarity

Similarity has been an important research topic in several fields such linguistic, Artificial intelligence (in particular in the field of Natural Language Processing and Fuzzy Logic). The range of application of measure of similarity ranges from word sense disambiguation, text summarization, information extraction and retrieval, question answering, automatic indexing and automatic correction of codes. There are two types of similarity: syntactically and semantic similarity. Syntactic similarity can be defined as functions over terms. For instance, the hamming distance (used in Information Theory). This similarity is defined as the number of positions with different characters in two terms with the same length. Whilst semantic similarity can be defined as Miller and Charles (Miller et al 1991) as a continuous variable that describes the degree of synonymy between two words.

When evaluating similarity in a taxonomy the most natural way to access similarity is to evaluate the distances between the two concepts being compared. Therefore the shorter is the path from one to another means that they are more similar. This approach has been used as measure of similarity. However, one of the main drawbacks is that it relies on the notion that links in a taxonomy represent uniform distances (Resnik 1995; 1998). Our own view is that similar entities are assumed to have common features[2]. For instance (university, research_institute) but it is also the case that dissimilar entities may also be semantically related by the relation meronym or holonym such as (student –person; bicycle-wheel) .

[2] The common features in two entities might not be so discriminative as the features which are different in them.

We have designed and implemented a similarity algorithm to assess concept similarity and relation similarity (Vargas-Vera and Motta 2004, Vargas-Vera et al 2003a,b). Our algorithm compares extended graph obtained from the user query (using entities in the query itself plus informative classes from ontology) and a graph which represents a subset of the ontology (relevant to the query). As an inter-media stage it creates the intersection upon nodes between the two graphs (the **graph of the query** and the **graph obtained from the ontology)**. In short, our similarity algorithm assess concept similarity and relation similarity using the Dice Coefficient using the most informative classes from the ontology.

Our similarity algorithm differs from other similarity algorithms in that it uses ontological structures and also instances. Instances provide evidential information about the relations being analyzed. This is an important distinction between the kind of similarity which can be achieved using only WordNet or the similarity which can be achieved using distances to super-classes followed by Wu et al. (Wu et al 1994). In the first one WordNet brings all the synsets found even the ones which are not applicable to the problem being solved. Whilst in the latest the idea of a common super-class between concepts is required (i.e. two concepts are similar if they share a common ancestor).

We present an explanation of our algorithm when arguments in the user query are grounded (instantiated terms) and they match exactly (at the level of strings) with instances in the ontology. A detailed description of the algorithm and example can be found in (Vargas-Vera et al. 2003b).

The algorithm uses all grounded terms in the user query. It tries to find them as instances in the ontology. Once they are located a portion of the ontology (G_2) is examined including neighborhood classes. Then an intersection [3] (G_3) between augmented query (using knowledge from the ontology) G_1 and G_2 is performed to assess structural similarity. It could be the case that in the intersection G_3 several relations can include the grounded arguments. Then the similarity is computed for all relations (containing elements of the user query) using Dice Coefficient. Finally, the relation with the maximum Dice Coefficient value is selected as the most similar relation.

Our similarity algorithm for relations is defined as follows:

SimilarityBase algorithm:

Case 1: X_1 and X_2 are grounded arguments.

1. Translate the question to First Order Logic i. e. predicate_name(X_1, X_2)

2. $\exists$ relation connecting C_1 and C_2 Λ $\exists$ $C_1 \supset X_1$ Λ $\exists$ $C_2 \supset X_2$ such that

relation(C_1,C_2) where relation is an ontological relation between C_1 and C_2.

3. $\exists$ relation connecting C_1 and C_2 Λ $\exists$ $C_1 \supset X_1$ Λ $\exists$ $C_2 \supset X_2$ such that

[3] Intersection means to find a sub-graph in G_2 which contains all concepts contained in graph G_1 using subsumption relation.

relation(C_1, C_2) Λ $\exists$ $S_1 \supset (U_{11} \subset U_{12} \subset \ldots \subset U_{1n}) \supset C_1 \Lambda \exists S_1 \Lambda (U_{11} \subset U_{12} \subset \ldots \subset U_{1n}) \supset C_2$

4. $\exists$ relation connecting C_1 and $C_2 \Lambda \exists C_1 \supset X_1 \Lambda \exists C_2 \supset X_2 \Lambda S_1 \supset X_1 \Lambda S_2 \supset X_2 \Lambda \exists (U_{11} \subset U_{12} \subset \ldots \subset U_{1n}) \subset C_1 \Lambda \exists (U_{21} \subset \ldots \subset U_{2n}) \subset C_2$
where Uij is a subclass of U_{ij+1}

5. Find the intersection, G_3, of G_1 and G_2 based upon the node labels.

6. Let be A and B vectors containing the features used to compare similarity.

Compute:

Concept _similarity = 0 no common concepts

Concept _similarity = 1 same set of concepts , otherwise

Concept_similarity = sim_dice(A,B) = $2*\sum_1^n aibi / \sum_1^n ai^2 + \sum_1^n bi^2$
vectors A and B are filled with the number of concept nodes of graph G_1 and G_3 respectively.

7. Compute **Relation_ similarity** = d_i = sim_dice(A,B) = $2*\sum_1^n aibi / \sum_1^n ai^2 + \sum_1^n bi^2$
vector A and B are filled with the number of arcs in the immediate neighborhood of the graph G_1 and G_2 respectively.

8. maximum(d_i) where i=1,n

The algorithm builds graphs from the user query G_1, obtain a fragment from the ontology containing the relevant nodes G_2 and build an intersection (G_3) between G_1 and G_2.
Step 2 describes the construction of the graph G_1 for the query. This is created using subject (X_1), relation, object (X_2) and the most representative classes for X_1 and X_2 respectively.
Step 3 describes how the algorithm finds sub-hierarchy containing grounded[4] arguments/concepts from the user question (i.e. the neighborhood containing the grounded arguments).
Step 6. the similarity between G_1 and G_3 using the Dice coefficient is computed. The vectors A and B are filled with the number of concept nodes of graphs G_1 and G_3 respectively then

Concept _similarity = 0 no common concepts

Concept _similarity = 1 same set of concepts , otherwise

Concept _similarity = $2*\sum_1^n aibi / \sum_1^n ai^2 + \sum_1^n bi^2$

Step 7. the similarity between G_1 and G_2 using the Dice coefficient is computed.

[4] Grounded argument means instantiated argument.

$$\text{Relation\_similarity} = d_i = 2*\sum_1^n aibi \,/\, \sum_1^n ai^2 + \sum_1^n bi^2$$

vector A and B are filled with the number of arcs in the immediate neighborhood of the graph G_1 and G_2 respectively.

A procedure called **SimilarityTop** uses the **SimilarityBase** algorithm (defined above), WordNet synsets, and feedback from the user. The SimilarityTop procedure checks if there is similarity between the name of the relation/concept in the query and the relation/concept in the selected ontology. If there is no similarity then it offers all the senses which are found in the WordNet thesaurus to the user. It is **SimilarityTop** that AQUA uses, with the selected sense, to rewrite the logic formulae. The main steps are defined as follows:

SimilarityTop procedure:

- Call our **SimilarityBase** algorithm (defined above)
- If **ontological_relation** $\neq \Theta$ then

  evaluate_query(ontological_relation(β_1,β_2))

- Else
  To obtain synsets for relation_question using WordNet thesaurus
  To ask user to select sense from the ones that WordNet thesaurus provided
  Call evaluate_query(selected_sense(β_1, β_2))

As a fist instance we have implemented on AQUA our similarity algorithms (which perform mappings between relation in a user query and ontological relations using neighborhood of classes). However, in future we would analyze other mappings algorithms which could be explored such as the one described by Compatangelo and Meisel (Compatangelo and Meisel 2003) or the mapping algorithm implemented in the system *Glue* which learns from examples of mappings (Doan et al., 2001).

4. Related Work

Levy et al. have suggested an architecture and question answering algorithm used in the information Manifold (Levy et al 1996). The information manifold have some features . First contents of information sources are described by query expressions that can uses classes and roles as well as predicates of any arity. Secondly, the query planning algorithm uses the source descriptions to determine precisely which information sources can produce answers to a given query. In our multi-ontology framework, the integration system is close in spirit to Levy (Levy et al., 1996; Levy 1998). The difference is that approach works over an ontological space and introduces the notion of meta-ontology.

Florescu et al suggested the use of probabilistic knowledge in mediator systems (Florescu et al., 1997). According to Florescu et al. probabilistic information is useful for ordering access to

information sources. They have suggested several algorithms for ordering access to data sources However a greedy algorithm seems to work better performance. In future, we might want to explore if by ordering access to data sources the AQUA performance can be improved.

5. Conclusions

We have extended the idea of Levy et al. (Levy et al., 1996) of having a data integration system for databases but in our work we deal with ontologies/knowledge bases. It is clear that a great deal of work needs to be done to deploy this technology. However, this is one stone to our goal is to have an automatic system which can deals with queries from users written in NLP (Natural Language Processing) using different resources. These resources are heterogeneous and use different data models and vocabularies. Therefore, to build such system is a challenge. The main advantages are described as follows:

- free users from the task of finding the sources where the answer can be found and
- interact with each one separately.

We have outlined a solution to the problem of data integration over an ontological space. This solution relies on a Meta-ontology which contains descriptions of each individual ontology in the ontological space. This meta knowledge is used in conjunction with similarity measures to determine relevant sources to a given query. We argue that by using meta-knowledge and similarity algorithms the data integration task can be performed automatically on behalf of users. Preliminary results are encouraging. However, further research is needed in similarity measurements.

The main application of the data integration system is in AQUA. Currently, AQUA deals with a single ontology at the time. Therefore, we expect that by means of data integration system AQUA will deal on behalf of the user (in a transparent way) with many sources.

Acknowledgments

This work was funded by the Advanced Knowledge Technologies (AKT) Interdisciplinary Research Collaboration (IRC), which is sponsored by the UK Engineering and Physical Sciences Research Council under grant number GR/N157764/01.

References

Arens Y. Chee C. Y. Hsu C. and Knoblock C.A. (1993): Retrieving and integration data from multiple information sources. J. of Intelligent and Cooperative Information Systems, 2(2):127-158,1993.

Batini C. and Lenzerini M. and Shamkant B. and N. A (1986): Comparative analysis of methologies for database schema integration. ACM computing Surveys, 18(4):323-364, 1986..

Cattarci T. and Lenzerini M. (1993): Representing and using interschema knowledge in cooperative information systems. J of Intelligent and Cooperative Information Systems, 2(4):375-398, 1993.

Compatangelo E., and Meisel H. (2003): "Reasonable" support to knowledge sharing through schema analysis and articulation. *Neural Computing and Applications,* volume 12, number 3-4, pages 129-141. © Springer-Verlag, 2003.

Doan A., Domingos P., and Halevy A. (2001): Reconciling Schemas of Disparate Data Sources: A Machine Learning Approach. In Proceedings of the ACM SIGMOD Conference, 2001

Florescu D. , Koller D. , Levy A.(1997): Using Probabilistic Information in Data Integration Proceedings of the 23rd VLDB Conference, Athens, Greece 1997 .

Goasdoue F. and Rousset M-C(2000): Rewriting conjunctive queries using views in Description Logics with existential restrictions. In Proc. of the 2000 Description Logic Workshop (DL 2000) pages 113-122, 2000.

Levy A. (2000): Logic-based techniques in data integration. Logic Based Artificial Intelligence , Kluwer Academic Publishers, Ed J. Minker, 2000.

Levy A. The Information Manifold Approach to Data Integration. (1998): IEEE Intelligent Systems, vol. 13, pp. 12 - 16, 1998.

Levy A., Rajaraman, J. Ordille. (1996): Query Answering Algorithms for Information Agents. In Proc. AAAI Conference, 1996.

Miller G. A. and Charles W. G. (1991): Contextual correlates of semantic similarity. Language and Cognitive Processes , 6(1), 1-28, 1991.

Resnik P. (1995): Using information content to evaluate semantic similarity in a taxonomy. In Proceedings of IJCAI-95, pages 448-453, Montreal, Canada, 1995.

Resnik P. (1998): Semantic Similarity in a Taxonomy: An Information-Based Measure and its Application to Problems of Ambiguity in Natural Language. Journal of Artificial Intelligence research, 1998.

Resnik P and Diab M. (2000): Measuring Verb Similarity. Proceedings of the 22nd Annual Meeting of the Cognitive Science Society (COGSCI2000), Philadelphia, USA, 2000.

Tversky A. (1977): Feautures of similarity. Psychological Review, 84, 327-352, 1977.

Vargas-Vera M and Motta E. (2004): AQUA - Ontology-based Question Answering System. Third International Mexican Conference on Artificial Intelligence (MICAI-2004), Lecture Notes in Computer Science 2972 Springer Verlag, (eds R. Monroy et al), April 26-30, 2004.

Vargas-Vera M., Motta E. and Domingue J. (2003a): AQUA: An Ontology-Driven Question Answering System (2003a): AAAI Spring Symposium, New Directions in Question Answering, Stanford University, March 24-26, 2003.

Vargas-Vera M., Motta E. and John Domingue J. (2003b): An Ontology-Driven Question Answering System (AQUA). KMI-TR-129, Knowledge Media Institute,The Open University, June 2003.