RUFORUM Working Document Series (ISSN 1607-9345) No. 14 (3): 71 - 76.
Available from http://repository.ruforum.org

Research Application Summary

AgriMatch: A dynamic Ontology Matching System

Ochieng, P. & Kyanda, S.
Department of Computer Science and Information Systems Makerere University, P. O. Box 7062,
Kampala, Uganda
Corresponding author: kswaibk@cis.mak.ac.ug

Abstract

Multiple conceptualizations are inevitable, especially in a widely distributed large scale system like the web. Every information community share a given conceptualization which may be different from another information community. Despite this difference in views, people and organizations need to exchange information. Data from various systems, structured under different information models, need to be integrated and accessed uniformly for various purposes. A possible way of dealing with this problem is to consolidate multiple conceptualizations into a unified form. Corresponding elements of the multiple conceptualizations may be mapped and treated uniformly. The aligned conceptualization (known as ontologies) can be used to annotate a document to add more meaning to the document. A search performed over these pages returns more informative that has high precision and recall.

Key words: Information community, information models, web

Résumé

Les conceptualisations multiples sont inévitables, en particulier dans un système largement répandu comme le web. Chaque communauté d'information partage une conceptualisation donnée qui peut être différente d'une autre communauté d'information. Malgré cette différence perçue, le monde et les organisations auront toujours besoin d'échanger des informations. Les données provenant de divers systèmes, structurées sous différents modèles d'information, doivent être intégrées et accessibles uniformément à diverses fins. Un moyen possible de traiter ce problème est de consolider plusieurs conceptualisations sous une forme unifiée. Les éléments correspondants des multiples conceptualisations peuvent être cartographiés et traités uniformément. La conceptualisation alignée peut être utilisée pour annoter un document afin d'y ajouter plus de significations. Une recherche effectuée sur ces pages produit plus d'information avec précision.

Mots clés: Communauté d'information, modèles d'information, web

User scenario

Peter, a Ugandan Farmer at Mukono district, wants to find out how Kenyans treat Newcastle disease in layer chicken which has been killing his birds for years. Using the current social web (2.0) Peter decides to use Google search engine using the words "How Kenyans treat Newcastle". This returns multiple pages to him which have different combination of the words Newcastle, disease, layers. Peter has to open pages one at a time extracting the required information. It is worth noting that this search has some key limitation. For instance, web pages of the Kenya government website dealing with Newcastle disease have the word Avian pneumoencephalitis instead of Newcastle. Therefore this page which is crucial to Peter is completely missed in the search though it might contain crucial information he is looking for. The current web therefore has no mechanism of solving words with same meaning but different representation. If a page has no key words used in the search crucial information is left out, i.e., the web has no intelligence to resolve different conceptualizations. Irrelevant pages which contain the sets of the keywords used in the search may also be fetched by the query wasting Peter's time. Peter has to read multiple pages to get aggregated information he is looking for. All information cannot be found in one page. This therefore raises key technical issues that have to be addressed to enable data integration:

1. Can all information required by the user be aggregated using a single search?
2. Can irrelevant pages be sifted out of the search pages returned?
3. Can a search engine pick all relevant pages including those using different terms but same meaning?

State of the Art

Many diverse solutions have been proposed to perform ontology matching task, and the most recent survey of the ontology matching tools are discussed in Shvaiko and Euzenat (2013) and Otero-Cerdeira *et al.* (2015). The key challenges in ontology matching are also discussed in Pavel Shvaiko and Jerome Euzenat (2008). The objectives of this study were to: (i) create an ontology matching algorithm to match various agricultural domain ontologies, (ii) build a natural language (NL) processing framework that converts user queries written in natural language into SPARQL query, and (iii) create a user Interface for query input and displaying the results to the user

Methodology

The ontology matching model is shown in Figure 1. It has the following modules:

Ontology Loader module. Ontology loader is implemented such that the ontologies to be matched (source and target) are loaded into memory using University Manchester OWL API (Manchester, n.d.).

Feature Extractor module. In order to match entity A and B from two ontologies the Feature extractor module extracts all annotations (local names, synonyms and labels) of

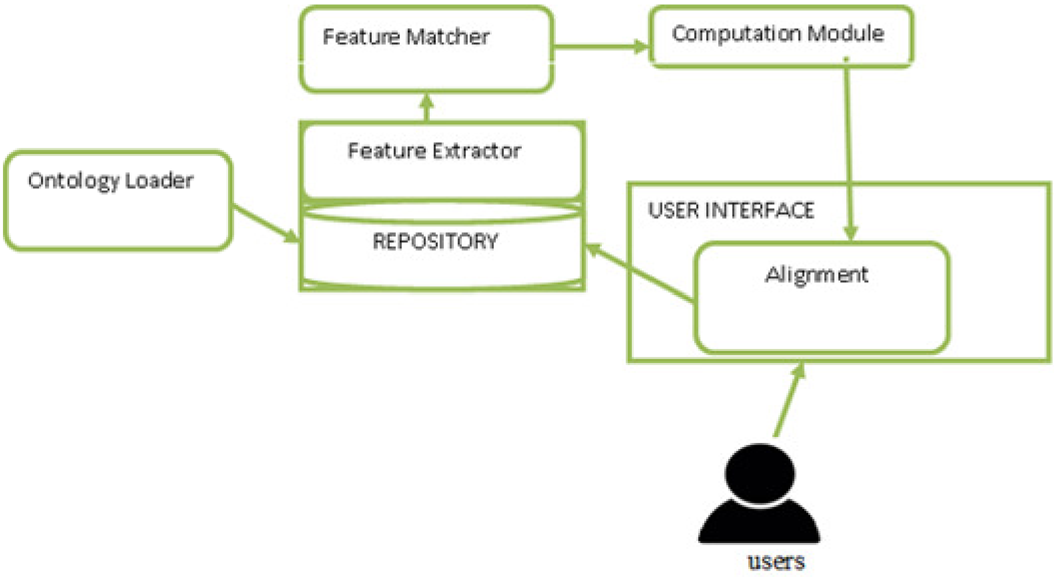

Figure1. AgriMatch ontology matching model

their children and the level of the child in the hierarchy from the entity to be matched. This information is stored in two Java HashMaps HashA and HashB, respectively. The HashMaps are implemented as Multimaps where the key is the class name of the child in the taxonomy and values are be the annotations extracted and the level of the child in the taxonomy from the parent. The goal of using HashMaps is to avoid against all comparison as proposed in Faria *et al*. (2013), since O(M× N) does not scale up well for large ontologies.

Feature Matcher Module. To get similar and dissimilar features shared between the two entities, the values of HashA is queried against HashB (note this can be in either direction) using entity (concept) name as the query value. If a similar feature is returned from HashB their level in both HashMaps are compared. If they are of the same level, a weight of 1 is assigned to the similarity. This is referred to as LEVEL weight. Otherwise a value of 0.1 is subtracted from 1 depending on the difference in levels. Also the type of matching features are assigned weight dynamically, this weight is referred to as FEATURE weight. The weight of the similarity feature is computed according to Equation 1. The similar feature and its weight is stored in a third HashMap HashCommon which has similarity feature as the key and weights as the values. The similar features are removed from HashA and HashB, otherwise no removal is done. This is done until all the features in HashA have been queried against HashB. Now all features that are similar will be stored in HashCommon and all dissimilar features will remain in HashA and HashB.

$$featureWeight = LevelWeight \ x \ FeatureWeight \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (1)$$

Similarity Computation Module. After taxonomically getting all the similar and dissimilar features of the two entities being compared, the probability of a given feature P(c) that appears in the common features set and probability of a given feature appearing in the dissimilar features set are calculated (Resnik, 1995). Frequency of a given feature is computed by counting the occurrence of all children noun of a given feature in the taxonomy according to the equation. We the use geometric mean as proposed in Sánchez *et al*. (2012).

Natural Language processing and SPARQL Generation

Ontology is written is OWL or RDFS, which is mostly queried using SPARQL which most normal users cannot write. Hence users are allowed to write their queries using their normal natural language that they understand. The system converts NL to SPARQL without user knowledge and returns answers in normal English. We implemented the system as shown in Figure 2.

Results and evaluation

We performed two sets of experiments. First to establish how matching algorithm perform in terms of Precision (a measure of the ratio of correctly found correspondences over the total number of returned correspondences). Recall (the ratio of correctly found correspondences over total number of expected correspondences) and F-measure were used to compared it with other leading ontology matching algorithms AML (Faria *et al.*, 2013) and YAM++ (Ngo and Bellahsene, 2012). Results are shown in Table 1.

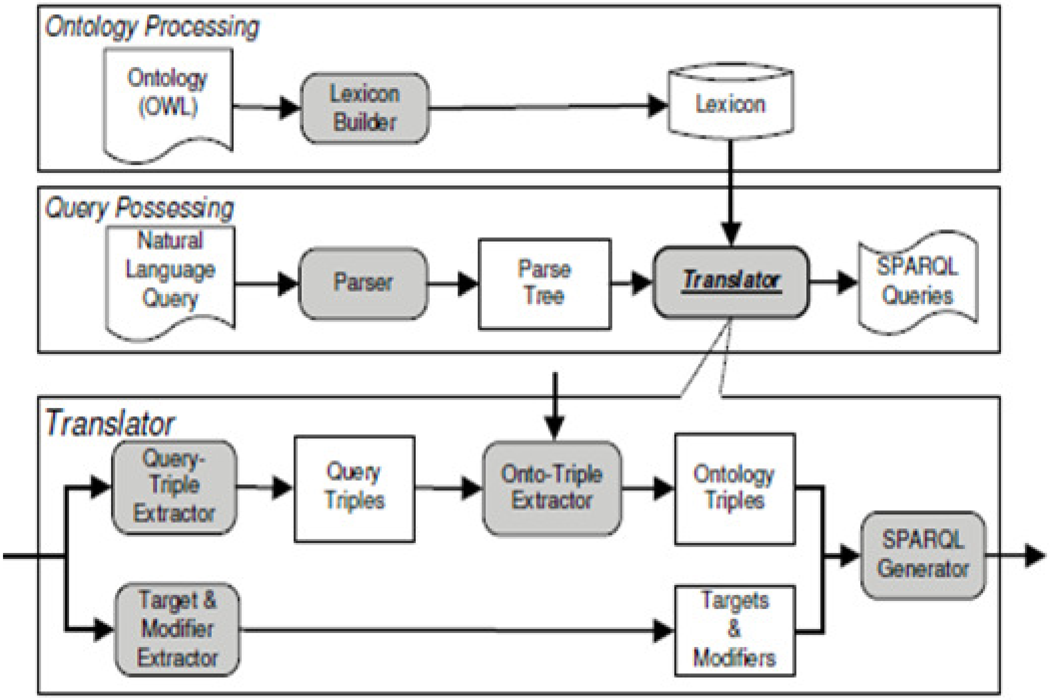

Figure 2. Overall system architecture combining Natural Language converter and Ontology Matching

Table 1. A comparison of the performance of Agrimatch with AML and YAM++

Method	Precison (%)	Recall (%)	F-measure (%)
YAM++	87.2	84.8	86.0
AML	84.7	71.8	78.0
AgriMatch	89.9	77.2	83.5

Table 2. Performance of Agrimatch in NLP processing

Method	Geography	Agriculture	Job
Original number of user queries	880	250	641
Number of selected Testing Queries	877	238	517
Precision	89.9	90.8	83.5
Recall	85.88	96.8	87.5

The second experiment was to assess how many of the translated queries correctly represent the semantics of the original natural language queries. We compared the output with the manually generated SPARQL queries. The metrics we used are precision and recall. For each domain, precision means the percentage of correctly translated queries in the queries that our system produced as output; recall refers to the percentage of queries that AgriMatch produced an output in the total testing query set. The results are shown in Table 2.

Conclusion

The aligned conceptualization (known as ontologies) can be used to annotate a document to add more meaning to the document. A search performed over these pages returns more informative that has high precision and recall.

Acknowledgement

This paper is a contribution to the 2016 Fifth African Higher Education Week and RUFORUM Biennial Conference.

References

Faria, D., Pesquita, C., Santos, E., Palmonari, M., Cruz, I.F. and Couto, F.M. 2013. Matching System. *Springer Berlin Heidelberg* 527-541.

Manchester, U. of. (n.d.). OWL API. Retrieved from http://owlapi.sourceforge.net/

Ngo, D. and Bellahsene, Z. 2012. Yam++: A multi-strategy based approach for ontology matching task. *Springer, Heidelberg* 7603: 421-425.

Otero-Cerdeira, L., Rodríguez-Martínez, F.J. and Gómez-Rodríguez, A. 2015. Ontology matching: A literature review. *Expert Systems with Applications* 42(2):949-971. http://doi.org/10.1016/j.eswa.2014.08.032

Pavel Shvaiko and Jerome Euzenat. 2008. Ten challenges for Ontology matching. In: *Proceedings of the 7th International Conference on Ontologies, DataBases, and Applications of Semantics (ODBASE)*. pp. 1163-1181.

Resnik, P. 1995. Taxonomy. In: *14th International Joint Conference on Artificial Intelligence* (Vol. 1). Montreal, Quebec, Canada: Morgan Kaufmann Publishers Inc.

Sánchez, D., Batet, M., Isern, D. and Valls, A. 2012. Ontology-based semantic similarity: A new feature-based approach. *Expert Systems with Applications* 39:7718-7728.

Shvaiko, P. and Euzenat, J. 2013. Ontology Matching: State of the art and future challenges. *IEEE Transactions on Knowledge and Data Engineering* 25 (X): 158-176. http://doi.org/10.1109/TKDE.2011.253.