An Incremental and User Feedback-based Ontology Matching Approach

Fernando Wagner
Universidade Federal do Ceará
Av. da Universidade, 2853
Fortaleza – CE – Brazil
+55 (85) 3366 7300
fernandow@lia.ufc.br

Jose A. F. Macedo
Universidade Federal do Ceará
Av. da Universidade, 2853
Fortaleza – CE – Brazil
+55 (85) 3366 7300
jose.macedo@lia.ufc.br

Bernadette Lóscio
Universidade Federal de Pernambuco
Av. Prof. Moraes Rego, 1235
Recife – PE – Brazil
+55 (81) 2126.8000
bfl@cin.ufpe.br

ABSTRACT
Ontologies are being used in order to define common vocabularies to describe the elements of schemas involved in a particular application. The problem of finding correspondences between ontologies concepts, called ontology matching, consists in the discovery of correspondences between terms of vocabularies (represented by ontologies) used by various applications. The majority of solutions proposed in the literature, despite being fully automatic, has heuristic nature and may produce non-satisfactory results. The problem intensifies when dealing with large data sources. The goal of this paper is to propose a method for generation and incremental refinement of correspondences between ontologies. The proposed approach uses filtering techniques, as well as user feedback to support the generation and refinement of such matches. For validation purposes, a tool was developed and some experiments were conducted.

Categories and Subject Descriptors
D.2.12 [**Interoperability**]: *Data mapping*
H.3.3 [**Information Search and Retrieval**]: *Information filtering*

Algorithms, Management, Performance, Experimentation, Human Factors, Verification.

Keywords
Ontologies, incremental matching, user feedback, filtering.

1. INTRODUCTION
With the advent of the Semantic Web, the concept of ontology has been discussed and used in several works in the field of data integration. Ontologies are being used in order to build shared vocabularies, which are intended to describe concepts and relationships of the terms contained in the data sources of a given application. The idea behind the use of common vocabularies consists in an attempt to reduce the heterogeneity among distinct data sources, thus facilitating integration between these.

The discovery of semantic correspondences among terms of vocabularies is an important step to accomplish the integration between data from different applications. In the ontology context, this operation is known as ontology matching. Formally, the process of ontology matching is a function that takes two input ontologies, a set of initial correspondences (or alignment) between them and returns a new alignment [5].

The generation of correspondences between ontologies has been identified as one of the main bottlenecks in data integration solutions. Generating correspondences between terms, as well as the maintenance of those, requires great effort and time consumption, especially when dealing with large schemas. In this scenario, a manual construction of correspondences becomes infeasible. Automated tools use heuristic algorithms, and can generate results containing a considerable percentage of false and/or missing correspondences. In addition, the generation of a full alignment between two ontologies is not always required. An example is the identification of correspondences among multiple domain ontologies (for example, DBPedia[1] and Yago[2]) where only some parts of them share a common conceptualization. Another example is in the bioinformatics field. The Gene Ontology[3], the resulting product of collaborative researcher's effort, is an ontology designed to support the annotation of genes. Consider the task alignment between the Gene Ontology and another ontology that is more specific and describes, for example, genes of only one species. In both cases, the alignment will cover just some portions of both ontologies.

In this context, the main contribution of this work is the proposal of an iterative and semi-automatic method for generation and incremental refinement of correspondences between ontologies describing large heterogeneous data sources. Such an approach will be based on filtering input ontologies in order to extract their most valuable parts, as well as gathering user feedback to validate generated correspondences. We claim that the participation of the user in the process of defining correspondences supported by automatic matching algorithms is key for achieving better results. More specifically, we believe that user feedback is fundamental to select important concepts to be aligned as well as to refine generated alignments. In addition, the approach uses existing automatic algorithms for generating correspondences, in order to assist the user in getting a resulting alignment with good quality.

The remainder of this paper is organized as follows: Section 2 discusses the main techniques of ontology matching; Section 3 mentions some related work; Section 4 introduces some definitions and the proposed approach; Section 5 highlights some

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

iiWAS2011, 5-7 December, 2011, Ho Chi Minh City, Vietnam..

Copyright 2011 ACM 978-1-4503-0784-0/11/12...$10.00.

[1] http://www.dbpedia.org

[2] http://www.mpi-inf.mpg.de/yago-naga/yago/

[3] http://www.geneontology.org/

advantages and limitations of the approach; and Section 6 presents conclusions and future work.

2. Ontology Matching

Currently, there are several algorithms based on different techniques for automatically generating correspondences between ontologies. As example, we mention syntactic and lexical algorithms, which use information from external sources (e.g. WordNet[4]), algorithms that analyze the similarity between the structures of the concepts and algorithms that make use of semantic techniques, such as logical deduction. Such algorithms can be combined with the goal to generate better results. This is because each of these approaches has distinct characteristics and can be used in complementary character. For example, a matcher based on WordNet is able to find correspondences that a syntactic matcher would have difficulty to identify.

In general, ontology matching solutions are automatic, i.e. they compute the set of correspondences between the input ontologies in a single iteration (single-shot). Due to this characteristic, the outcome of this process may include a considerable number of uncertain correspondences, in addition to a large degree of incompleteness, especially when dealing with large schemas. An alternative that allows the improvement of the final result is the user's involvement in the process of alignment [4]. The main justification for the semi-automation is that the user has specific requirements and a prior knowledge about the domain of the involved ontologies, and could contribute for obtaining a final alignment with better quality. Therefore, matching solutions should allow user intervention in order to confirm desirable correspondences and to eliminate the undesirable ones. Furthermore, it should be possible to allow the combination of different algorithms in order to find a solution that satisfies the requirements of the user.

Another important issue in ontology matching is that, generally, solutions consider the whole set of concepts of the given ontologies during the correspondences identification. However, in cases of matching ontologies with a large number of concepts, filtering the input ontologies, i.e. extracting some of their most valuable parts, can be critical to achieve better quality results. For example, suppose the case of alignment between specific domain ontologies (e.g., MusicOntology[5]) and multiple domain ontologies (e.g., DBPedia), typical scenario on the Web of data. In this case, may be necessary to filter the multiple domain ontology to eliminate concepts that are not relevant for the matching operation.

3. Related Work

The work described in [4] introduces incremental matching concept, presenting a prototype, which receives two XML schemas, and for each element of the source schema, the user has an option to use a matcher algorithm in order to suggest candidate correspondences, linking it to the target schema elements. The user is able to confirm (or create) the correct correspondences for each element. Despite cognitive facilities, using this approach becomes complicated for large schemas, because the matching is performed element by element, and the user may waste a lot of time during this process.

Danny Chen et al proposes in [3] a matching tool called OntraPro, which has a simple interface and uses iterative algorithms to compute the candidate correspondences between input ontologies. Then, the user is able to reject or create correspondences. The results are stored and reused in order to generate new correspondences. When dealing with large schemas, the probability of errors by the user increases, given the large number of candidate matches that are created. This overloads the user, increasing chances of human errors and "rippled effect happens", reducing the quality of the final result.

The PROMPT [6] is a semi-automatic tool that provides support to the matching and merging task between ontologies. The PROMPT has an extension called COGZ, which assists the user in performing such tasks involving large ontologies. In addition to visual features, COGZ has a filtering mechanism for search terms, in order to highlight relevant ontology terms and hide the irrelevant ones. Such a mechanism increases the user's focus to certain parts of ontologies, reducing occurrences of error. However, the engine filters the ontologies only for usability purposes. Automatic Algorithms available in PROMPT do not consider filtering, generating results of lesser quality.

With the goal of filtering schemes and generate partitions containing related concepts, Villegas et al. [2] suggests a measure to calculate the proximity between concepts. The idea is to consider the schema as a graph where nodes are concepts and edges are relationships. Given an input concept, the definition of distance between nodes is applied to this and other schema concepts, being returned the closest concepts. This strategy seems to be quite promising, since it allows for a greater focus by the user, in the relevant partitions in ontologies.

4. Proposed approach

In this section, we present the I3M (*Incremental, Interactive and Iterative Matcher*) approach for ontology matching. Initially, we present some definitions, which are relevant for the understanding of our proposal.

Definition 1 (*Ontology*): An ontology is a tuple $\boldsymbol{O} = <\boldsymbol{C}, \boldsymbol{R}, \boldsymbol{I}>$ where $\boldsymbol{C}$ is a set of concepts, $\boldsymbol{R}$ is a set of relationships between these concepts and $\boldsymbol{I}$ is a set of instances that belong to one or more concepts.

Definition 2 (*Proximity*): Given concepts $\boldsymbol{e_1}$ and $\boldsymbol{e_2}$ from an ontology $\boldsymbol{O}$, the proximity between $\boldsymbol{e_1}$ and $\boldsymbol{e_2}$, denoted by $prox(\boldsymbol{e_1},\boldsymbol{e_2})$, may be defined as the minimum number of relationships linking these concepts. This definition is similar to distance's definition in graph theory. Suppose, for example, that the graph of Figure 1 illustrates an ontology, where nodes are concepts and arcs are relationships. The proximity between concepts *A* and *B* is 1, because a single relationship links *A* and *B*. However, the proximity between concepts *C* and *G* is 4, because the minimum "path" between these nodes consists of four relationships (arcs).

Definition 3 *(Ontology partition):* A partition of an ontology $\boldsymbol{O}$ may be defined as a sub-ontology $\boldsymbol{S}$ derived from $\boldsymbol{O}$. The highlighted concepts (with relationships that connect them) presented in Figure 1 are an example of an ontology partition.

Definition 4 (*Correspondence*) Is a 5-tuple **A** = <**id**,**e1**,**e2**,**r**,**p**> where **id** is a unique identifier for this, **e1** and **e2** are concepts from source and target ontologies respectively, **r** is the type of relation and **p** is the strength value between [0,1].

[4] http://wordnet.princeton.edu/

[5] http://musicontology.com/

Definition 5 (***User Feedback***): Is a tuple **UF** = <**A**, **F**> where **A** denotes a correspondence and **F**, a boolean value. If **F** is false, then, the correspondence **A** is false, else, **A** holds.

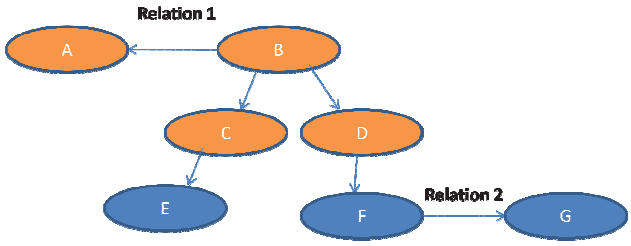

Fig 1. Example of ontology with a highlighted partition.

The general idea of our approach consists in extending the traditional process of ontology matching (*single-shot*), adding the following properties:

- **Incremental**: in each interaction, the I3M process considers just ontology partitions instead of the whole ontologies.
- **Interactive**: the I3M process takes into account the user feedback in order to improve precision and recall of matching results.
- **Iterative**: for each partition, the user will work in iterations, where, in each iteration, a different matcher may be used for the generation and refinement of matches. Moreover, correspondences obtained from each iteration are stored and can be reused in subsequent iterations.

Figure 2 depicts the I3M process, which is composed of the following tasks:

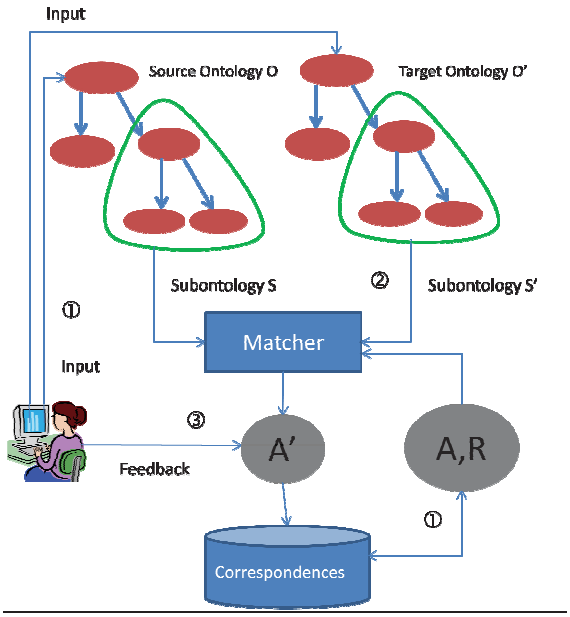

Fig 2. Overview of the I3M process

Task 1: Ontology partition generation. The user selects two ontologies, a concept **c** and an integer **k** (for each ontology), which constitutes the input set for this task. Next, both ontologies are filtered according to their respective input set, generating two partitions (sub-ontologies). The set of concepts that compose each partition is defined automatically, taking into account the order of proximity between **c** and other ontology concepts {**c1,…, cn**}, such that *prox(**c, cj**)* = i, 1≤ i ≤ k and j ≤ n.

Task 2: Matcher selection. During this task, ontology partitions are given as input to automatic algorithms for generating correspondences between them. The idea is to provide a set of matchers with distinct characteristics in such a way that the user may select the most suitable one, in order to contribute for the improvement of the quality of the final result. These algorithms may also take into account the correspondences previously confirmed/rejected. The result of this step is an alignment between two ontology partitions;

Task 3: User feedback gathering. After the execution of a matcher, the user has the choice of rejecting false correspondences as well as the opportunity of adding undiscovered ones. Users may also undo pruning and additions of correspondences. This is justified by the fact that user can make mistakes. The outcome of this task is a new set of correspondences, which is stored in a repository and can be used in subsequent interactions.

It is important to note that Tasks 2 and 3 may be repeated until the user decides to stop the matching process.

To illustrate the use of the I3M approach, consider the following scenario: a user wants to establish correspondences between the schemas of a given proprietary ontology and the DBPedia ontology. The proprietary ontology is relatively small, contains a few numbers of concepts and refers to biological life domain, including concepts such as animals, plants and related terms. The DBPedia ontology is a multi-domain ontology and contains hundreds of concepts. Figure 3 shows the interface of the I3M prototype, developed to validate the proposed approach, where the input ontologies must be specified. The user may choose between filtering the ontologies or using them in their entirety.

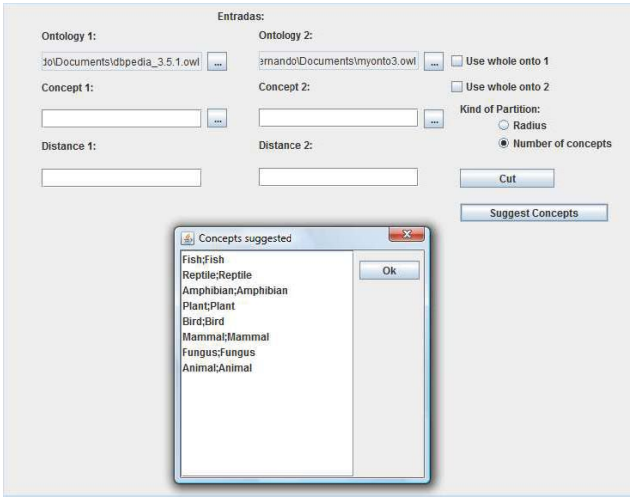

Fig 3. Defining the partitions

Considering the proposed scenario, the user chooses to filter the DBPedia ontology, due to its large number of concepts. To do this, a concept name and an integer to delimit the partition size must be specified. In order to facilitate the choice of this concept, a light-weighted matcher can be used to suggest initial correspondences between the input ontologies. Based on such correspondences, the user may choose the most suitable concept to be used for generating the partition for the DBPedia ontology.

Suppose that the user chooses “Animal” (last concept suggestion in Figure 3) and the number 15. So, DBPedia ontology is filtered and a partition is created, which consists of the "Animal" concept

with more 14 related concepts. As mentioned before, the proprietary ontology is relatively small, and is not filtered.

After the partitioning of the DBPedia ontology, the existence of prior matches involving concepts of partitions is verified in the database. Then, the user may select an automatic matcher available in the prototype. Next, the alignment generated by the matcher is displayed (Figure 4) and the user may eliminate false correspondences and/or add correct ones that were not found earlier. The changes will be recorded in a repository. The user may select another matcher aim at generating a new alignment that will be added to the alignment previously generated.

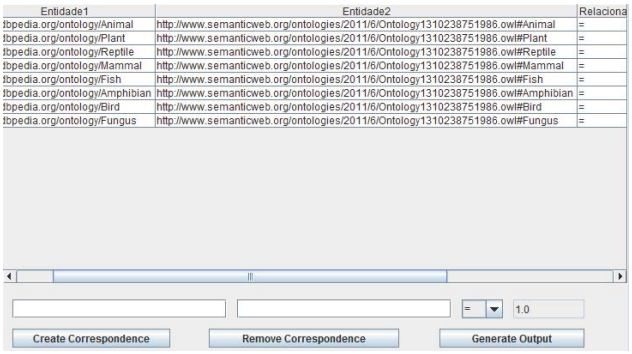

Fig. 4. Displaying the alignments

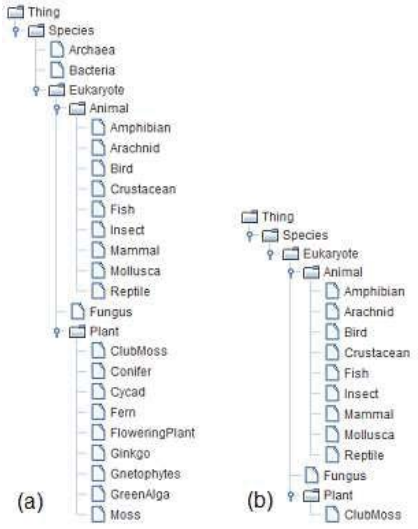

Fig 5. Partitions from DBPedia (sizes 25 and 15 respectively)

At the end of user feedback gathering, the user may increase the partition size. Suppose, for example, that the user looks at the sub-ontology structure of size 15 (Figure 5 (b)) and suspects that there exists more concepts related to biological life domain that should be considered in the ontology alignment. Then, in this case, the user may increase the integer input from 15 to 25, in order to increase the ontology partition. As a result, a new partition is created (Figure 5 (a)) and the user may use it, repeating steps 2 and 3 several times in order to discover new correspondences.

5. Advantages and Limitations

Some advantages and limitations of the proposed approach are discussed in this section. Firstly, the filtering feature brings benefits, both for user and automatic matcher. Some matchers can match filtered ontologies faster and providing less errors than using whole ontologies as input, implying in a drastic decrease in the number of false positives because many possibilities of generating incorrect correspondences may be eliminated. This also means that the user has less difficulty on correcting automatic generated alignments. In addition, filtering is cited by [1] as a strong cognitive help that facilitates the provision of user feedback, since user will handle with a smaller amount of concepts. Thus, even an automatic matcher provides similar results with and without filtering, chances of error may be reduced. Another advantage is in case of ontology evolution. Since the user knows what portion of the ontology has been changed, it is possible to consider just the partition that was changed, instead of working with whole ontology.

To achieve good results, it is desirable that the user has a good knowledge about the involved domains and that he will be able to spend considerable time during the matching process. So, as limitations, we highlight that this process may be time-consuming and require deep user domain knowledge.

6. Conclusions and future work

In this paper we present an interactive, iterative and incremental method for generation and refinement of correspondences between ontologies. The usage of ontology filtering facilitates the manipulation of matches by allowing the user to select only the ontology fragment that is interested for him/her. In addition, it also contributes to increase the efficiency of automatic algorithms, because the entries have a reduced size. Ontology partitions may be dynamically increased or decreased, according to the user needs. This feature allows the user to delimit accurately the scope of interest within the ontologies. On the other hand, we assume that: (i) the user is familiar with the domain(s) and (ii) the concepts belonging to a particular domain are structurally and semantically related close to each other.

Future activities include conducting experiments using refined matchers, such as contained in OAEI[6] evaluation. We also aim at reducing processing time in matching execution for large ontologies. Also, we will investigate how to improve usability in order to reduce user's effort during feedback gathering.

7. REFERENCES

[1] Falconer, S., Storey, M. A cognitive support framework for ontology mapping. In *Proceedings of the 6th International and 2nd Asian Semantic Web Conference*, Busan, Korea, 114-127, 2007.

[2] Villegas, A., Olivé, A. A Method for Filtering Large Conceptual Schemas. In *Proceedings of the 29th International Conference on Conceptual Modeling*, Vancouver, Canada, pp. 247-260, 2010.

[3] Chen, D., Lastusky, J., Starz, J., Hookway, S. A User Guided Iterative Alignment Approach for Ontology Mapping. In *Proceedings of the International Conference on Semantic Web and Web Services*. Las Vegas, USA, pp. 51-56, 2008.

[4] Bernstein, P. A., Melnik, S., Churchill, J. E. Incremental schema matching, In *Proceedings of the international conference on very large data bases*. Seoul, Korea, 1167-1170, 2006.

[5] Euzenat, J., Shvaiko, P. Ontology Matching. Springer. 2007.

[6] Noy, N. F., Musen M. A. The PROMPT suite: Interactive tools for ontology merging and mapping. 2003.

[6]http://om2010.ontologymatching.org/