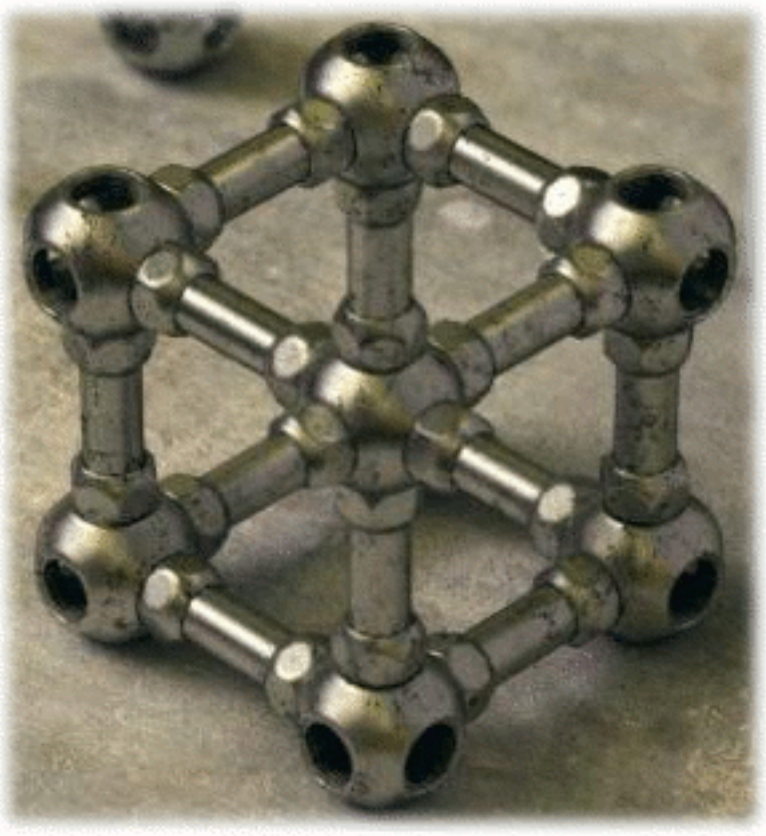

Ontology Merging: on the confluence between theoretical and pragmatic approaches [1]

Author(s):
Raphael Cóbe
Renata Wassermann
Fabio Kon

[1]This work was supported by Fapesp Project LogProb, grant 2008/03995-5, São Paulo, Brazil.

Ontology Merging: on the confluence between theoretical and pragmatic approaches

Raphael Cóbe[*], Renata Wassermann, Fabio Kon

[1]Department of Computer Science – University of São Paulo (IME-USP)

{rmcobe,renata,fabio.kon}@ime.usp.br

Abstract. *In recent years, researchers have focused on merging knowledge bases in both pragmatic and theoretical points of view. In this paper, we enumerate a few attempts to deal with inconsistencies while merging knowledge bases. We focus on ontology merging and show that pragmatic and theoretical approaches are not integrated and that both could benefit from a closer relationship. We extended an existing theoretical algorithm for Description Logics and applied it for the ontology merging problem. We describe here an implementation of this algorithm as an open source Protégé plugin.*

1. Introduction

There has been a rapid increase in availability of (semantic) information on the web. Nevertheless, there is no standard way of reusing knowledge, creating a challenge of building new knowledge bases for specific domains. This has forced users to build knowledge bases from scratch instead of being able to reuse previously established knowledge.

Ontologies have been considered as a mean for expressing and sharing semantic knowledge among systems [Gruber 1993] specially in the context of the Semantic Web. Their underlying structures allow machine-processing, providing a common vocabulary for expressing metadata about each web resource. Also, they are based on first order logic, allowing the usage of reasoners that are able to infer relationships between concepts based on their logical description. In that sense, W3C proposed the OWL[1] standard specification language to express ontologies.

The main challenge faced by knowledge integration research is solving inconsistencies by removing the minimum amount of information so that the remaining stays consistent. Since we are talking about removing part of the knowledge from the base, it is important to clarify the difference among three kinds of knowledge integration: (a) merging, (b) revising, and (c) updating. Revesz [Revesz 1995] claims that revision and update operators are characterized by the inclusion of knowledge that is either more or less relevant (or trustful) than the knowledge previously defined, while merge, in contrary, does not prefer any piece of knowledge over another.

That kind of concern resulted in works like [Konieczny and Pérez 1999] where the authors have dealt with the merging problem for the propositional logic case by means of a model-based approach. This kind of work has inspired most of the research conducted in the theoretical field of first order logic merging, such as the work of Gorogiannis et al. [Gorogiannis and Hunter 2008] and Qi et al. [Qi et al. 2006]. Unfortunately, pragmatic approaches did not follow the same evolution pace as theoretical ones and just a

[*]The author would like to thank FAPESP for sponsoring his research.

[1]www.w3.org/TR/owl-features/

few tools have been developed to provide knowledge base integration. This might have happened because pragmatic research has focused on the ontology mapping activity. If we think about the whole knowledge integration as a process: first of all, we have to compute whether there are similar concepts and how similar such concepts are - this is the *mapping activity* - and each concept correspondence is called concept *match*. After mapping the concepts, the merging activity takes place. During the merging, the concepts from all the knowledge bases are copied into the output base. Thus, mapping is the activity that most of the time comes before the merging (at least in the ontology integration) [Falconer et al. 2007].

In this paper, we present our current work on ontology merging, including the implementation of a plugin for the Protégé[2] editor.

This paper is organized as fallows. Section 2 presents a brief summary of pragmatic and theoretical works that aim to deal with ontology inconsistencies. Section 3 explains our efforts in bridging the gap between both points of view and present the merging plugin we developed. Finally, at Section 4, we present a few conclusions taken from our work and discuss what we plan to do in the future.

2. Ontology Merging

In this section, we intend to show the common approaches used to deal with the ontology merging and inconsistency handling problem. We have divided this section in two to show that these two fields of study are not dealing with the same problems.

2.1. Theoretical Approaches

Only a few studies in the literature deal directly with description logics based knowledge integration and inconsistency management. We classify these works, like [van Harmelen et al. 2005], into two main categories: syntactic and semantic-based approaches. The syntactic-based approaches sees ontologies as a set of axioms, which are syntactic objects, while semantic-based approaches sees ontologies as a set of models, which are semantic objects that are represented by a finite set of axioms.

In the context of syntactic-based approaches, we would like to cite the research conducted by Thomas Meyer and his colleagues. They proposed an algorithm for finding maximally consistent sets from inconsistent knowledge bases [Meyer et al. 2005]. This algorithm is a modification of the *conjunctive maxi-adjustment* algorithm for propositional knowledge integration and is called *CMA-DL*. In that sense, such work is similar to the work developed by van Harmelen et al. [van Harmelen et al. 2005], but instead of looking for maximally consistent subsets their goal is to build minimally inconsistent subsets, which they call diagnoses. The main difference between these two approaches is that the CMA-DL algorithm takes into account the order of the bases to be merged. Each base is called *strata* and the set of all strata is called *stratified knowledge base*. This set is ordered by preference, which means that the first ontology is preferred over the second one during the merging activity. We have proposed a small modification to this algorithm that gives to the user all possible merging precedence order. We have used this algorithm to implement our Protégé merging plugin.

Most of the semantic-based approaches have been inspired by model-based propositional logic inconsistency solving like what is presented at [Konieczny and Pérez 1999].

[2] `http://protege.stanford.edu`

In that context, Gorogiannis et al. [Gorogiannis and Hunter 2008] propose an approach to deal with inconsistencies by means of *Dilation Operators* that are, basically, an strategy to iteratively relax the formulas to remove inconsistencies. The authors have first proposed the use of Dilation Operators to deal with inconsistencies in propositional logics and showed the equivalence of their approach to the one from Konieczny and Pérez [Konieczny and Pérez 1999]. Finally, they took the idea of using Dilation Operators further and proposed an operator that iteratively transform first order formulas by changing universal quantifiers into existential ones. This approach may solve a few inconsistencies but we figured out that it would be hard to translate it to an ontology context. For instance, the description logic formula equivalent to $\forall x.p(x) \rightarrow z(x)$ would be $p \sqsubseteq z$ but we were unable to define a way to dilate the description logic formula so it would be equivalent to the dilated first order logic formula, i.e. $\exists x.p(x) \rightarrow z(x)$. Qi and colleagues have also proposed model-based operations to solve inconsistencies in ontologies. In [Qi et al. 2006] they proposed a model-based operator named *weakening* and showed that its results are semantically equivalent to those of CMA in stratified knowledge bases.

2.2. Pragmatic Approaches

In this section, we present a few tools which purpose is to manage multiple ontologies to combine and promote the reuse of knowledge. We will discuss the PROMPT approach for ontology merging and specially inconsistency handling in more detail as it was the only one that we have found that deals with inconsistency. We have tried a few other tools but, unfortunately, none of them provided any inconsistency handling mechanism. For instance, we have tried Watson For Knowledge Reuse[3], which is a tool that allows the user to query a web service that contains ontologies and ask it for suggestions on new concepts to be added. It may suggest to include relationships and concepts that may break the ontology consistency. So, it is not specially concerned with keeping the ontology consistency. We have also tried the OWLDiff[4] tool. It intends to work just like the common Unix *diff* command, providing an easy-to-use interface that shows the differences between two ontologies. It also allows the user to copy ontology fragments between ontologies but does no consistency checking after doing so.

We have studied the PROMPT tool for merging and it does provide support for inconsistency management. Unfortunately, the inconsistencies dealt with PROMPT strategies are not logic inconsistencies and they arise due to the fact that its merging algorithm sometimes fail in merging concepts and properties. We figure that PROMPT does not deal with logical inconsistencies because when the authors proposed its idea [Noy and Musen 2000], the Protégé OWL tool did not provide support for more expressive logics constructions like the disjoint clause.

PROMPT deals with 4 kinds of inconsistencies: a) Name Conflict: this inconsistency happens when the algorithm includes two different concepts with the same name in the merged ontology, so the system advises the user to rename them; b) Dangling References: this inconsistency happens when the image of a given property is missing in the merged ontology, so the system suggests that the user includes such concept into the merged ontology; c) Redundant Hierarchy: this inconsistency happens when there is more than one path connecting a concept to one of its ancestors, so the system suggests that the

[3] `http://neon-toolkit.org/wiki/1.x/Watson_for_Knowledge_Reuse`
[4] `http://krizik.felk.cvut.cz/km/owldiff/`

user removes one of these paths; d) Slot Constraint Violation: this inconsistency happens when some property has its cardinality violated at the merged ontology, e.g. a property that should have only one individual as its images is used to connect two different pair of individuals. The systems then suggests that the user removes one of these individuals.

3. On the confluence of theoretical and pragmatic approaches

The main focus of the work that we are currently developing is to bridge the gap between the theoretical and pragmatic approaches for ontology integration. Unfortunately, pragmatic approaches have very little to offer since the only conflict solving approach described (by PROMPT) does not deal with logical inconsistencies. Also, the theoretical works most of the time cannot be directly applied to Ontology merging, their algorithms were designed for using with first order logics like [Gorogiannis and Hunter 2008].

We chose to use the CMA-DL algorithm designed by Meyer et al. [Meyer et al. 2005] to solve merging inconsistencies in description logics and applied it to ontology merging. The algorithm proposed the generation of maximally consistent subsets of the axioms present at each ontology at the inconsistent knowledge base in an iterative way. We chose to use such algorithm as the starting point for our research because it is a syntactic-based approach that can clearly build maximally consistent ontologies, not like the algorithm from [van Harmelen et al. 2005], which relies on a *Connectedness* notion and the proposed *Direct Structural Connection* function cannot detect axioms that cause inconsistencies that are not structurally connected.

The CMA-DL algorithm takes into consideration that, at the inconsistent knowledge base, the ontologies are sorted by order of preference. We believe that, sometimes, this is the case and we believe that this approach is close to the knowledge revision, but sometimes we cannot classify the ontologies according to their relevance. Let us take a look at the example (taken from [Meyer et al. 2005]) for instance:

Example 1: Consider the knowledge base $K = (S_1, S_2)$ and that the ontology S_1 is composed of the following axioms $bird(tweety), \neg flies(tweety), bird(chirpy)$ and that the ontology S_2 is composed of the axiom $bird \sqsubseteq flies$. It is easy to see that this knowledge base is inconsistent, sice S_1 states that *tweety* is a bird that cannot fly and S_2 states that every bird flies.

The CMA-DL algorithm gives preference to the knowledge present in S_1 and the result for its processing is the ontology O composed of the axioms exclusively from S_1, i.e., $O = \{bird(tweety), \neg flies(tweety), bird(chirpy)\}$. Although this is an ontology free of inconsistencies, we argue that at this case the discarded axiom seems to be very important to the result. It is a constraint that applies to all individuals of the bird concept.

We have proposed a modification to the CMA-DL algorithm that takes into account all possible ordering combinations of the input knowledge base. Such algorithm can be seen at the Listing 1. It accumulates the results of the possible combinations and leaves to the user the choice of which one to use. The algorithm presented relies on a set of specific operations. It uses a *PowerSet()* operation to calculate all possible subsets of a given set and *Permutations()* to calculate all possible permutations of a given set. It also relies on the *Ontology()* operation to generate a new Ontology from a set of axioms given and, conversely, it uses the *Axioms()* operation to retrieve a set containing all axioms of a given ontology.

Listing 1. Modified CMA-DL

```
1  Algorithm M-CMA-DL:
2    Input: A set of ontologies 𝒟 := (D_1, D_2, ..., D_n)
3    Output: A Set of Consistent Ontologies
4    ℬ := {An Empty Ontology}
5    for all Permutation (O_1, O_2, ..., O_n) in Permutations(𝒟) do
6      for i := 1 to n do
7        𝒞 := ℬ
8        for all Ontology C in 𝒞 do
9          𝒪 := Axioms(O_i)
10         j := ‖𝒪‖
11         𝒮 := ∅
12         repeat
13           𝒳 := {X ‖ X ∈ PowerSet(𝒪) and ‖X‖ = j}
14           for all Axiom Set X in 𝒳 do
15             if Ontology(Axioms(C) ∪ X) is consistent then
16               𝒮 := 𝒮 ∪ {X}
17             endif
18           endfor
19           j := j − 1
20         until 𝒮 is not empty
21         for all Axiom Set S in 𝒮 do
22           ℬ := (ℬ \ {C}) ∪ {Ontology(Axioms(C) ∪ S)}
23         endfor
24       endfor
25     endfor
26   endfor
27   return ℬ
```

If we run our version of the algorithm using as input the same knowledge base presented in Example 1, we would get the following set as output: $R = \{O_1, O_2, O_3\}$, where $O_1 = \{bird(tweety), \neg flies(tweety), bird(chirpy)\}$, $O_2 = \{bird(tweety), bird(chirpy), bird \sqsubseteq flies\}$, and $O_3 = \{bird(chirpy), bird \sqsubseteq flies\}$. One can easily see that our approach gives more power of choice to the user and at two different options he/she is able to keep the axiom that states that every bird flies, which was our primary goal.

We have developed a Protégé view plugin (Figure 1). that allows users to merge two ontologies at each time. The plugin can use both the classic and our version of the CMA-DL. It uses the HermiT[5] reasoner to check the consistency and OWLAPI[6] to access and manipulate ontologies. It is distributed under the GPL v3.0 license and is available at `http://ccsl.ime.usp.br/en/onair/ontology-merging`. The plugin allows the user to pick two ontologies from his/her filesystem and choose the destination where the resulting ontologies are going to be stored. Lastly, the user chooses whether he/she wants to use the classic CMA-DL or our modified version by checking the option "Merge ontologies using the first one as more important (Classic CMA-DL)".

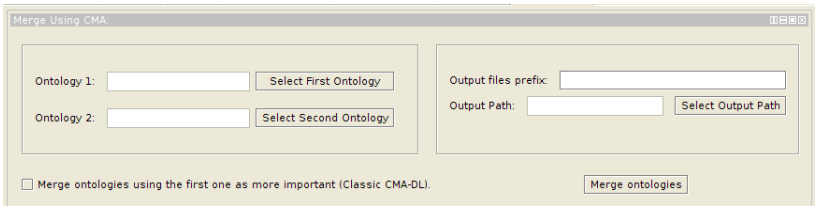

Figure 1. Protege Merging Plugin

[5] `http://hermit-reasoner.com/`
[6] `http://owlapi.sourceforge.net`

4. Conclusions and Future Work

In this paper, we presented a brief overview of theoretical and pragmatic research in the ontology merging field. We showed that there is a big gap between the pragmatic and theoretical approaches for ontology merging. We believe that both sides would benefit from a higher degree of integration. Also, we believe that the pragmatic field is a little bit stagnant when it comes to dealing with inconsistent merging results. In that context, we have chosen the CMA-DL algorithm as a starting point for our research because it clearly solves inconsistencies and could be directly applicable to ontologies. After a few experiments, we have proposed a small modification for this algorithm to provide more power of choice to the users. Now, the user can choose which ontology ordering suits better his/her needs.

Currently, we are working on building a software library to manage inconsistencies. An initial version of it is available at `http://ccsl.ime.usp.br/en/onair/ontology-merging`. Also, we intend to implement a web version for this merging mechanism and integrate it to the OnAIR - Ontology Aided Information Retrieval system[7], which is an ontology-based search tool for multimedia bases. OnAIR has an ontology-based query expansion feature and the merging mechanism would help the experts to build better ontologies, improving the retrieval results quality.

References

Falconer, S., Noy, N., and Storey, M. (2007). Ontology mapping-a user survey. In *Proceedings of the Workshop on Ontology Matching (OM2007) at ISWC/ASWC2007*, Busan, South Korea.

Gorogiannis, N. and Hunter, A. (2008). Merging first-order knowledge using dilation operators. In *Proceedings of the 5th international conference on Foundations of information and knowledge systems (FoIKS'08)*, pages 132–150, Berlin, Heidelberg. Springer-Verlag.

Gruber, T. (1993). A translation approach to portable ontology specifications. *Knowledge acquisition*, 5(2):199–220.

Konieczny, S. and Pérez, R. P. (1999). Merging with integrity constraints. In *Fifth European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU'99)*, pages 233–244.

Meyer, T., Lee, K., and Booth, R. (2005). Knowledge integration for description logics. In *Proceedings of the National Conference on Artificial Intelligence AAAI'05*, volume 20, pages 645–650. AAAI Press.

Noy, N. and Musen, M. (2000). Prompt: algorithm and tool for automated ontology merging and alignment. In *Proceedings of the 17th National Conference on Artificial Intelligence (AAAI-00).*, pages 450–455.

Qi, G., Liu, W., and Bell, D. (2006). A revision-based approach to handling inconsistency in description logics. *Journal of Artificial Intelligence Review*, 26(1):115–128.

Revesz, P. Z. (1995). On the semantics of arbitration. *International Journal of Algebra and Computation*, 7:133–160.

van Harmelen, F., Haase, P., , Huang, Z., Stuckenschmidt, H., and Sure, Y. (2005). A framework for handling inconsistency in changing ontologies. *The Semantic Web–ISWC 2005*, pages 353–367.

[7] `http://ccsl.ime.usp.br/onair`