Refining Ontologies by Pattern-Based Completion

Nadejda Nikitina and Sebastian Rudolph and Sebastian Blohm

Institute AIFB, University of Karlsruhe
D-76128 Karlsruhe, Germany
{nikitina, rudolph, blohm}@aifb.uni-karlsruhe.de

Abstract. Constructing richly axiomatized ontologies for real-world knowledge-intensive applications is a time-consuming and difficult task. For this reason, the future relevance of ontologies in practice depends on the availability of advanced semi-automatic methods for ontology learning and refinement. In this paper we propose a method to enrich ontologies with complex axiomatic information by completing partial instantiations of ontology design patterns.

Keywords: Ontology design patterns, ontology refinement.

1 Introduction

Richly axiomatized ontologies are essential for powerful, knowledge-intensive applications, since they allow the application of advanced reasoning. However, ontology modeling and construction is a difficult and time-consuming task. Considering the fact that real world applications require large-scale knowledge bases, there is a need for automatic methods to support ontology construction. Unfortunately, automatically constructed ontologies tend to lack expressivity, e.g., axiomatic information about transitivity or symmetry of relations.[1] For example, the relations "before" and "after" found in DBPedia[2] are not specified to be transitive while transitivity would clearly be an expected characteristic of these relations. This kind of information is difficult to acquire from unstructured resources, since it often does not explicitly occur in text, but can only be deduced using external information sources. However, we think that exploiting additional information sources can help to bring forward the ontology enrichment.

As pointed out in [1], typical conceptual patterns arise during the design of ontologies for different domains and different tasks. Ontology design patterns [2] – modeling solutions to solve a recurrent ontology design problem – were introduced to support the reuse of formalized knowledge. We consider the ontology design patterns as a potential source to be exploited for ontology refinement.

The usefulness of ontology design patterns in semi-automatic ontology construction has already been demonstrated in [3], where they have been used to put automatically constructed ontology elements in context by extending the ontology

[1] Relations are also referred to as properties in related literature
[2] http://dbpedia.org/page/Angela_Merkel

with abstract concepts and relations. In this paper, we aim at the refinement of ontology's axiomatization using highly axiomatized knowledge contained in ontology design patterns. The key idea of the proposed approach is to search for components within an ontology which partially instantiate a given ontology design pattern. In this way, potential missing ontology elements can be identified.

If we consider the previously mentioned relations "before" and "after" contained in DBPedia as well as the ontology design pattern "precedence" introduced in [4] and if we assume the ontology part including these two relations to be a partial instantiation of the given ontology design pattern, we see that there are three axioms missing in DBPedia – the axioms expressing the inverseness of the relations "before" and "after" and their transitivity.

The ability to automatically recognize partial instantiations of an ontology design pattern would therefore allow for checking an ontology for potential missing elements and based on the outcome to automatically generate a list of suggestions for a refinement. In this way, using frequently occurring and richly axiomatized ontology design patterns as input could help to add a considerable amount of axioms to a sparsely axiomatized ontology.

In order to automatically recognize an ontology design pattern by the means of an algorithm, a set of indicative features of this pattern is required. We identify the instantiations of ontology design patterns by their structure and the meaning of their elements expressed by axioms and lexical characteristics of each element. Our matching algorithm is based on these types of features. In this paper, we presume an extended kind of ontology design patterns which contain additional lexical information. In the following, we are going to use the expression *ontology pattern* or *pattern* instead of ontology design pattern.

Our method is mainly independent from the employed concrete ontology representation language. However, we presume that the underlying ontology representation language of concerned ontologies supports complex axiomatizations.

The remainder of this paper is organized as follows: The next Section describes research work related to this paper. Our algorithm is introduced in Section 3. Section 4 summarizes and gives an outlook to further research.

2 Related Work

Semi-automatic ontology construction and refinement has been addressed by several approaches relying on different types of data sources. However, only few of them aim at the acquisition of complex axioms going beyond the modeling capabilities of RDFS.

There is a range of methods exploiting the information contained within natural language texts in order to acquire additional axioms (for an overview see [5]). [6] proposes a method for an axiomatization of glossaries such as WordNet based on parsing and converting of natural language descriptions into formal definitions. [7] also aims at the acquisition of complex axioms by the means of deep syntactic analysis of natural language definitions.

There is a range of approaches relying on multiple data sources such as [8] which aims at the acquisition of a particular type of axioms, namely disjointness axioms, by gathering syntactic and semantic evidence from different data sources. RELExO [9] combines learning complex class descriptions from textual definitions with the FCA-based technique of relational exploration in order to clarify the subclass relationship of concepts of an ontology. It generates hypotheses about class extension relationships which cannot be deduced or denied using the axioms already contained in the ontology. Then, it looks for counterexamples in the set of instances contained in the ontology and, if none could be found, it asks the expert to provide a counterexample or to approve the suggested hypothesis. RoLExO [10] relies on the same type of user interaction and hypothesis verification, but generates hypotheses about complex domain-range restrictions. A method proposed in [11] is another example of extracting hypothetical domain axioms based on a given set of entities. These approaches are complementary to ours, since they rely on other sources of information to acquire complex axioms.

Blomqvist [3] proposes a framework for pattern-based semi-automatic ontology construction and refinement. This work focuses on the refinement of 'lightweight' ontologies concerning the logical complexity and expressiveness, which are not intended to obtain a rich axiomatization. For this reason, ontology patterns are not used to enrich the ontology with complex axioms, but to put the automatically learnt ontology elements into context by connecting them with the more general concepts and relations of the pattern.

To the best of our knowledge we are the first to address the general use of ontology design patterns for semi-automatic enrichment of ontologies with complex axioms.

3 Matching Ontologies and Ontology Patterns

The proposed method is based on ontology matching. Matching of ontologies has been widely covered in literature. An overview of the existing approaches can be found in [12]. We use a modified ontology matching technique due to the specific requirements for matching ontology patterns with ontologies. The main particularity of pattern matching is the high average level of abstractness characteristical for the concepts of a pattern. The concepts contained in a pattern are usually abstract enough to match many different concepts in an ontology. Therefore, relations are often the major indicators for a pattern instantiation. Especially lexical information about relations is essential for a better performance of the matching algorithm. Thus, we consider it useful to invest additional effort to a-priori enrich patterns with lexical information. Our algorithm is designed to exploit provided additional lexical information.

Before presenting the algorithm, we state the underlying criteria for a high likelihood of pattern realization by an ontology part. Thereby, we reduce the problem of identifying partial instantiations of a pattern to the problem of identifying complete pattern instantiations. We rely on the following set of criteria:

A part O of an ontology ONTOLOGY is a potential instantiation of the considered ontology pattern P, if its structure can be matched completely with the structure of P in a way that

1. *Each concept C_P contained in P has exactly one corresponding concept C_O in O, which is equivalent to C_P or a subconcept of it;*[3]
2. *Each relation R_P contained in P has exactly one corresponding relation R_O in O, so that domain and range concepts of R_O are the correspondents (according to 1) of the domain and range concepts of R_P and R_O implies R_P;*
3. *Each axiom A_P of P can be deduced from ONTOLOGY when its concepts and relations are replaced by their correspondents according to 1 and 2.*

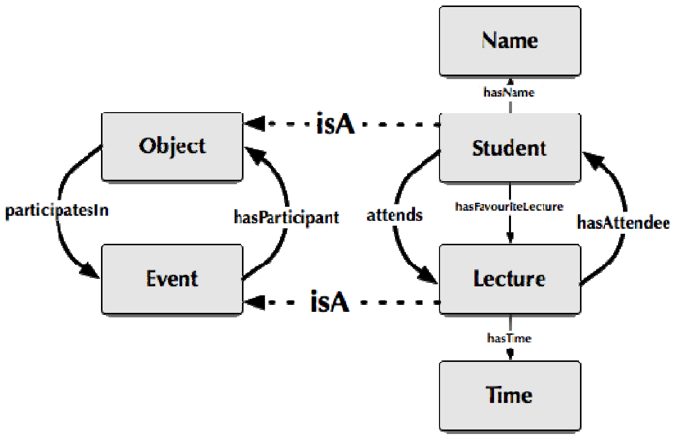

Fig. 1. Matching of an ontology part and a pattern

Figure 1 shows how the content ontology pattern “Participation” is matched with a part of an example ontology according to the stated criteria. The concept “Student” is a specific “Object”, the concept “Lecture” is a specific “Event”, and thus they correspond as required by condition 1. Relations “hasParticipant” and “hasAttendee” as well as “participatesIn” and “attends” are expressed by synonymous expressions and their domain and range concepts correspond to each other (condition 1). For this reason they imply each other and correspond to each other as required by condition 2. If the pattern also contains an axiom declaring “participatesIn” to be the inverse relation of “hasParticipant”, then according to condition 3 it must be possible to deduce it from the set of ontology axioms. In this case, an axiom declaring “attends” to be inverse to “hasAttendee” would suffice.

The criteria stated above provide the basis for our matching algorithm. It uses lexical properties of relations and concepts to verify whether concepts and relations correspond to each other as required in 1 and 2. We will briefly describe the matching of ontology elements based on lexical properties in the following subsection before discussing the algorithm in more detail.

[3] Equivalence, subconcept and implication relationships are correspondences established by the ontology matching method introduced later on.

3.1 Matching Lexical Properties

The goal of the lexical matching is to determine whether a concept is equivalent to or a subconcept of a particular concept and a relation is implied by a particular relation. For this purpose, we use the lexical information contained in ontologies and rely on the availability of particular lexical information in patterns. In the following, we describe these kinds of information.

Lexical information contained in ontologies differs in its detailedness and purpose. We distinguish between a label and a linguistic pattern (LP). A label of an element is a string used as a name for a concept or a relation whereas a LP is used to recognize the instances of a concept or a relation in text. LPs can range from simple regular expressions to more complex structures enriched with different kinds of linguistic information such as concept's part-of-speech type. Even though LPs would be very useful for matching due to their potential richness, the representation of LPs is not standardized in widely used ontology representation languages such as OWL. Therefore, we do not consider LPs in our approach and use only the labels of elements.

For the verification of conditions 1 and 2 using labels, we rely on a list of synonyms and hyponyms for each pattern concept and a list of synonyms and troponyms[4] for each pattern relation. We match each synonym and hyponym or troponym with each label of the potentially corresponding pattern element in the ontology based on string-similarity.

3.2 Matching Algorithm

The matching algorithm in its simplified form can be stated as shown in Fig. 2 and Fig. 3. The algorithm receives an ontology and an ontology pattern as input and generates a list of pattern instantiations as output. It first identifies pairs of lexically matching ontology and pattern elements. Then, to avoid unnecessary computations, it selects the pattern element, which has the fewest lexical matches in the ontology. Since the pattern can only be matched as long as all of its elements have a corresponding element in the ontology, considering only the occurrences of the pattern element with the fewest number of correspondents assures that the least number of ontology parts is analyzed. Each occurrence of the selected element is then analyzed using the recursive procedure growAlignments starting with the given pair of matched elements.

Due to possible hyponymy or troponymy between the elements of the pattern and the ontology, several valid alignments are possible. For a particular initial partial alignment the outcome can differ depending on the order in which elements are matched. For this reason, the algorithm tries all possible ways to construct an alignment and gathers all valid alignments.

[4] Troponyms are expressions for more restrictive relations

```
Algorithm 1 The alignment algorithm
Require: ontology $O$, ontology pattern $P$
  result ← $\emptyset$ {result: set of valid alignments}
  for all $p \in P$ do
    occurrences ← $\emptyset$
    for all $o \in O$ do
      if lexicalMatchingSuccessful($p$, $o$) then
        occurrences ← occurrences $\cup\{(p, o)\}$
      end if
    end for
  end for
  $l$ ← leastFrequentPatternElement(occurrences)
  $A \leftarrow \emptyset$
  for all $c$ with $(l, c) \in$ occurrences do
    $A \leftarrow A\cup$ growAlignments($\{(l, c)\}$)
    for all $a \in A$ do
      if axiomaticMatchingSuccessful($a$) then
        result ← result $\cup a$
      end if
    end for
  end for
  return result
```

Fig. 2. Alignment algorithm

```
Algorithm 2 growAlignments is a recursive procedure
Require: $a$: partial alignment
  result ← $\emptyset$
  markAllAlignedElementsAsGreen($a, O, P$)
  for all $p_g \in$ greenPatternElements do
    $o_g$ ← ontologyCorrespondentOf($p_g, a, O$)
    for all $p_r \in$ redNeighboursInPattern($p_g, P$) do
      for all $o_r \in$ redNeighboursInOntology($o_g, O$) do
        if $(p_r, o_r) \in$ occurrences then
          result ← result $\cup$ growAlignments($a \cup \{(p_r, o_r)\}$)
        end if
      end for
    end for
  end for
  return result
```

Fig. 3. Recursive procedure growAlignments

It marks the already matched pattern and ontology elements green and the remaining elements red. For each green pattern element *A* it calculates the remaining red neighbors and matches each of them with the remaining red neighbors of the ontology element corresponding to *A*. If the lexical matching was successful for a pair of elements, they are included into the current alignment which forms the input for another run of the described procedure. The resulting alignments are gathered in a set.

After collecting all valid alignments for the currently analyzed pattern occurrence, axiomatic matching is applied to each alignment to verify that axioms of the ontology pattern can be deduced from the axioms of the ontology, if concepts and relations in the pattern axioms are replaced by the concepts and relations of the ontology. For this task, a state-of-the-art reasoner such as Pellet[5] or HermiT[6] can be used.

4 Ontology Refinement Based on Partial Pattern Instantiations

The algorithm presented above can be used to find partial pattern instantiations by separating pattern elements into obligatory and optional elements and applying the algorithm to the set of obligatory pattern elements. Assuming the availability of a set of richly axiomatized patterns containing additional sets of synonyms and hyponyms for each concept and relation, the ontology engineer can compose a list of patterns for the ontology refinement by choosing the whole set at once or selecting some patterns manually if he or she only needs a particular type of patterns.

For each pattern in this list, the ontology engineer can select the obligatory elements and the level of accuracy, which is the acceptable extent of pattern incompleteness, expressed as the number of pattern elements relative to the total number of pattern elements. The level of accuracy can be set for each pattern or for the whole refinement process. It allows to limit the required user interaction and at the same time to influence the matching performance towards a higher recall or a higher precision. The ontology engineer can also use the default settings. Per default, the level of accuracy is greater zero, which allows considering potential pattern matches containing at least one pattern element. The default obligatory elements are the concepts and relations of each pattern. Axioms are however optional.

After the setup, the algorithm is run for each of the selected patterns. Found alignments are checked for axiomatic incompatibility with the optional pattern elements in order to avoid refinement suggestions which result in an inconsistent ontology. Finally, for each pattern, a list of refinement suggestions is generated and presented to the ontology engineer, who can select some suggestions for the integration into the ontology and start the automatic integration process.

During the integration, partial alignments are used to integrate the unmatched pattern elements into the ontology. Thereby, matched elements themselves are not integrated, but are replaced in axioms by their correspondents before integrating the axioms and unmatched pattern elements into the ontology (Table 1). Concepts and relations missing in the ontology can be optionally renamed by an expert in order to obtain less general names and in this way to better suit the level of abstraction present in the ontology.

[5] http://clarkparsia.com/pellet/
[6] www.hermit-reasoner.com/

Table 1. Integration of pattern elements into an ontology.

Type of unmatched pattern element	Action before inserting the element into the ontology
Concept	Optional renaming by an expert
Relation	Replacement of all matched pattern concepts contained in domain and range axioms by their correspondents, optional renaming by an expert
Axiom	Replacement of all matched concepts and relations by their correspondents

5 Feasibility Study

We conducted an experiment on the ontologies contained in the Watson Ontology repository[7] in order to assess the potential of the proposed method. In the experiment, we used the previously described example consisting of the transitive relations "before" and "after" to examine how well the proposed method can perform for axioms involving transitivity and inverseness of relations.

In the experiment, we used only the relation label itself for the lexical matching. The relations were considered as obligatory pattern elements whereas the axioms about their transitivity and inverseness were considered to be optional.

We used the Watson Search Engine [13] to identify the ontologies containing an *ObjectProperty* definition for at least one of the relations. Thereby, 14 documents were identified and matched against the pattern with results as displayed in Table 2.

Table 2. Experiment results: matching of the after-before-pattern with ontologies indexed by the Watson Ontology Search Engine.

Ontology URL	Result
morpheus.cs.umbc.edu/aks1/ontosem.owl	Inverse only
lists.w3.org/Archives/Public/www-rdf-logic/2003Apr/att-0009/SUMO.daml	"After" is missing
secse.atosorigin.es:10000/ontologies/SUMO.owl	"After" is missing
daml.umbc.edu/ontologies/cobra/0.3/daml-time	Inverse only
ai.sri.com/daml/ontologies/time/Time.daml	Inverse only
cs.umd.edu/~golbeck/daml/slaveOnt.daml	No transitivity and no inverseness
cs.vu.nl/~pmika/owl-s/time-entry-fixed.owl	Complete
isi.edu/~pan/damltime/time-entry.owl	Complete
pervasive.semanticweb.org/ont/2004/06/time	Complete
pervasive.semanticweb.org/ont/dev/time	Complete
isi.edu/~pan/damltime/time.owl	Complete
mogatu.umbc.edu/ont/2004/01/Time.owl	Complete
sweet.jpl.nasa.gov/sweet/time.owl	Complete
daml.umbc.edu/ontologies/cobra/0.4/time-basic	Complete

[7] http://watson.kmi.open.ac.uk/WatsonWUI/

Eight of 14 documents resulted in a complete match of the pattern including all axioms. Three of the ontologies did not include the inverseness axiom, but the transitivity axioms. Two documents did not contain a definition for the relation "after", but a definition for the relation "before" which was defined as transitive. One document did not contain any of the mentioned axioms. We manually examined the refined ontologies and found that the performed completions were semantically justified.

6 Summary and Outlook

In this paper, we presented an algorithm for the identification of ontology pattern instantiations in ontologies along with a method to transfer complex axioms contained in ontology design patterns into a target ontology. The results of our experiment demonstrate the potential of the reuse of formalized knowledge. However, in order to assess the impact of the method more precisely, we plan a large-scale evaluation involving a large set of pattern with different characteristics.

The availability of appropriate and complete ontology patterns is essential for the effectiveness of our approach. Hence, we are currently working on semi-automatic methods to acquire useful patterns as well as the necessary lexical information for each pattern. For the former, we are planning to exploit existing ontologies to identify frequently co-occurring characteristics of ontology elements and in this way to identify particularly useful ontology patterns for ontology refinement. For the latter, we expect existing broad-coverage data sets such as WordNet, BillionTriple-Challenge[8] and DBPedia to be valuable resources. We also intend address the acquisition of composed relation labels such as *followed_by* or *authorOf*, since they are typical in the existing ontologies and difficult to obtain from the usual grossaries. For this purpose, we intend to use the existing methods for the extraction of synonyms and hyponyms based on Harris' Distributional Hypothesis [14] such as [15].

Since the effectiveness of our approach is highly dependent on the quality of the lexical matching, we are currently working on the incorporation of disambiguation techniques as well as matching techniques based on LPs in our lexical matching approach.

Acknowledgments. This work is supported by the EU FP6 NeOn project http://www.neon-project.org.

References

1. Gangemi, A.: Ontology design patterns for semantic web content. In: International Semantic Web Conference, 262–276 (2005)

[8] http://vmlion25.deri.ie/

2. Presutti, V., Gangemi, A.: Content Ontology Design Patterns as Practical Building Blocks for Web Ontologies, In: Proc. of the 27th Int. Conf. on Conceptual Modeling, (2008)
3. Blomqvist. Blomqvist, E.: Semi-automatic Ontology Construction based on Patterns. PhD-Thesis. Linköping University, Department of Computer and Information Science (2009)
4. Presutti, et al.: A Library of Ontology Design Patterns: Reusable Solutions for Collaborative Design of Networked Ontologies. NeOn D2.5.1 (2008)
5. Buitelaar, P., Cimiano, P.: Ontology Learning and Population: Bridging the Gap between Text and Knowledge, volume 167 of Frontiers in Artificial Intelligence and Applications. IOS Press (2008)
6. R. Navigli and P. Velardi.: Ontology enrichment through automatic semantic annotation of on-line glossaries. In Proc. of the 15th Int. Conf. on Knowledge Engineering and Knowledge Management (EKAW), LNCS, vol. 4248, pp. 126–140. Springer, Heidelberg (2006)
7. Völker, J., Hitzler, P., Cimiano, P.: Acquisition of OWL DL axioms from lexical resources. In: Proc. of the 4th European Semantic Web Conference (ESWC'07) (2007)
8. Haase, P., Völker, J.: Ontology learning and reasoning - dealing with uncertainty and inconsistency. In da Costa, P.C.G., Laskey, K.B., Laskey, K.J., Pool, M., eds.: Proc. of the Workshop on Uncertainty Reasoning for the Semantic Web (URSW). 45–55 (2005)
9. Völker, J., Rudolph, S.: Lexico-logical acquisition of OWL DL axioms – An integrated approach to ontology refinement. In: Proc. of the 6th Int. Conf. on Formal Concept Analysis (ICFCA'08) (2008)
10. Völker, J., Rudolph, S.: Fostering web intelligence by semi-automatic owl ontology refinement. In Proceedings of the 7th International Conference on Web Intelligence (WI) (2008)
11. Baader, F., Ganter, B., Sertkaya, B., Sattler, U.: Completing description logic knowledge bases using formal concept analysis. In Veloso, M.M., ed.: IJCAI. 230–235 (2007)
12. Shvaiko, P., Euzenat, J.: A survey of schema-based matching approaches. Journal on Data Semantics, IV. 146–171 (2005)
13. d'Aquin, M., Baldassarre, C., Gridinoc, L., Sabou, M., Angeletou, S., Motta., E.: Watson: Supporting next generation semantic web applications. In Proc. of the WWW/Internet conference (2007)
14. Harris, Z.S.: Word. Distributional Structure 10. 146–162 (1954)
15. Suchanek, F. M., Sozio, M., Weikum, G.: SOFIE: a self-organizing framework for information extraction. In Proc. of the 18th Int. Conf. on World Wide Web (WWW '09) (2009)