Towards Ontology Matching via Pattern-Based Detection of Semantic Structures in OWL Ontologies

Ondřej Šváb-Zamazal, Vojtěch Svátek

University of Economics, Prague, Dept. Information and Knowledge Engineering,
Nám. Winstona Churchilla 4, 130 67, Praha 3, Czech Republic
ondrej.zamazal@vse.cz, svatek@vse.cz

Abstract. Ontology Matching is nowadays a vivid area of Computer Science. There are several OM tools looking for correspondences between entities of ontologies. These correspondences are usual simple equivalence mapping pairs class-to-class or property-to-property. In our work we concentrate on diverse kinds of semantic structures in ontologies in terms of their detection and mutual matching. For this kind of matching not only equivalence relations as well as not only homogeneous correspondences are envisaged. The paper is a report from the first phase of the work aiming at ontology matching via pattern-based detection of semantic structures in OWL ontologies. In this initial phase we mainly pay attention to n-ary relations and their discovery in OWL ontologies. Subsequent phases lie in description of conditions of semantic matching between semantic structures.

1 Introduction

Ontology Matching (OM) is nowadays recognised as a crucial task in many diverse applications such as ontology evolution, schema integration, P2P information sharing, web service composition, web service browsing and so on [3]. Currently there are many OM tools that address the problem of automatic discovery of correspondences between entities in different ontologies[1] (OM task). However, most of these tools do not leverage deep-level aspects of ontologies and merely use superficial aspects of ontologies. There are a few approaches with logic-based approach [5] and the use of background knowledge [2]. Furthermore, most OM tools match homogeneous entities, ie. class-to-class, property-to-property, instance-to-instance, however there are several situations when so-called *heterogeneous matching* does make sense [4]. Next, OM tools usually match individual entities. These entities are however parts of more sophisticated structures [6, 11] that result from the conceptualization choice of ontology designers. Therefore, it can be useful to match those structures as a whole. Last but not least, most OM tools only discover equivalence relationships between entities being matched, more rarely subsumption relationships, and theoretically further relations are mentioned such as overlapping or disjointness. There are no approaches that consider semantic relations (i.e. relationships with specific domain semantics) between entities. Still, such capabilities of matchers could be beneficial for an application usage.

[1] In our work, we purely concentrate on *domain* ontologies expressed in OWL [1].

In our work we concentrate on diverse kinds of semantic structures in ontologies in terms of their detection and mutual matching, which should address abovementioned points. This work comprises three interlinked tasks: first, the description of meaningful *semantic structures* in ontologies wrt. the OM task. Second, the description of symptoms of semantic structures through *patterns*. Finally, the description of conditions of *semantic matching* between semantic structures. Semantic matching means that not only equivalence and subsumption are in play.

Our patterns originally consist of name and graph aspects of ontologies [12]. As a work in progress, we have to extend this notion of pattern with respect to detection of semantic structures. It will also lead to extension of our original formalization of these patterns where we need to incorporate properties (domain/range) and logic axioms (such as disjointness axioms and so on) from OWL ontologies. Currently, *patterns* comprise three aspects (three kinds of building blocks) of semantic structures: naming patterns, structural patterns, and logic axioms patterns.

The paper describes the initial phase of our work on an OM approach that would allow for heterogeneous matching, matching of whole structures and matching by 'rich' semantic relations. In this phase we are specialized in diverse kinds of structures discovered by deep-level analysis of ontologies. Those structures usually reflect conceptualization choices of ontology designers. In Section 2 we describe several semantic structures using use cases. Then we pay special attention to the discovery of n-ary relations (in the Section 3). Sections with related work and conclusions wrap up the paper.

2 Semantic Structures

In our case, semantic structures are limited to those potentially useful for OM task. Currently, we consider the following semantic structures: *parallel taxonomies* [12], *value partitions*, *part-whole relations*, and *n-ary relations*. The first three patterns are briefly presented (with use cases) in the following subsections. N-ary relations as semantic structures are described in more detail in Section 3, which also includes experimental results.

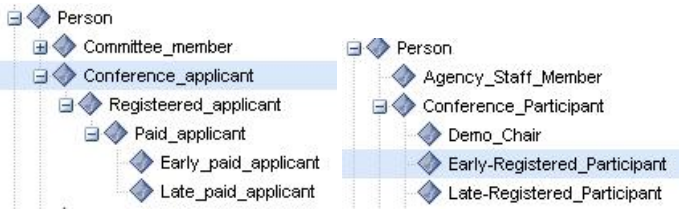
2.1 Parallel Taxonomies

Parallel taxonomies are two or more structural taxonomies (each constituting an interconnected fragment of concepts from an ontology) that share a significant proportion of their *distinct token set*. The distinct token set is a set of tokens that are part of names of some but not all concepts from the structural taxonomy; these tokens typically amount to distinguishing adjectives or appositives, while the tokens *shared* within the taxonomy refer to the underlying entity itself. We formally described this structure in [12].

Parallel taxonomies use case. An example of parallel taxonomies from the domain of conference organization, namely, from the OntoFarm collection,[2] are in Figure 1. On the left-hand side (conference.owl) there is a taxonomy dealing with the 'applicant'

[2] `http:/nb.vse.cz/~svatek/ontofarm.html`

concept (in total there are five concepts). On the right-hand side, there is a taxonomy dealing with 'Participant' concept (in total there are three concepts). These taxonomies have been recognised as parallel taxonomies across ontologies, because they share their distinct token set[3]. If we compare the two head nouns from those parallel taxonomies ('applicant' and 'Participant'), it seems to be clear that they are related—perhaps 'applicant' is the previous step of 'Participant' in the process of registration. The whole taxonomies can be aligned with some named semantic relation, eg. 'relatesTo', or 'becomes'.



(a) conference.owl (b) ekaw.owl

Fig. 1. Example of parallel taxonomies across ontologies (conference.owl,ekaw.owl)

2.2 Value Partitions in OWL

There are at least two ways how to represent specified collections of values expressing 'qualities', 'attributes', or 'features' (as it was described at `http://www.w3.org/TR/swbp-specified-values/`):

- *enumeration of the individuals*, where qualities are instances,
- *values as subclasses partitioning a 'feature'*, where qualities are classes

Detection of these two alternatives can lead to heterogenous matching (class-to-instance).

Value partitions use case. The 'Topic' concept can be divided into many more specific topics, see Figure 2. In the ontology O1, the partitions of 'Topic' concept are considered as subclasses of 'Topic' that are pairwise disjoint. Furthermore, it can be additionally specified that the 'Topic' concept is completely defined (exhaustive description). On the other side, the 'Topic' concept can also be considered as an enumeration of individuals, see the ontology O2. Again, it can be a fully specified concept (exhaustive description). Syntactically, it is possible to use an OWL collection and 'oneOf' axiom (according to the SWBPD[4] 'specified values' pattern [9]). This example occured in the ontologies 'edas.owl' and 'MICRO.owl' from the OntoFarm collection, where the 'hasTopic' property has also been specified.

[3] This sharing is not of 100% but majority-based.
[4] W3C Working Group for Semantic Web Best Practices and Deployment.

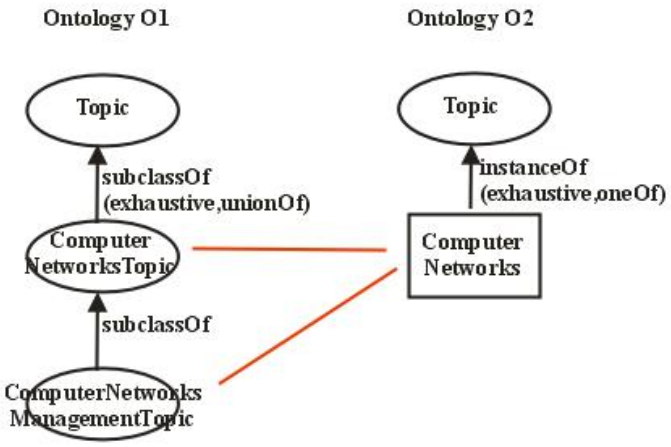


Fig. 2. Example of value partitions

2.3 Part-Whole Relations

State-of-the-art OM tools are able to find several types of correspondences—equivalence, subsumption, overlapping and disjunction—although equivalence is considered much more often than others. It could however also be useful to search for named semantic relations; relations between entities then would have to be detected depending on the context of a particular matching (correspondence).[5] One of characteristic problems to be addressed by named semantic relations is the modelling of part-whole relations. This relation is actually quite important during modelling, as most ontology engineering methodologies warn designers to carefully distinguish between part-whole relations and kind-of relations.

There are two basic SWBPD patterns for representing the semantic structure of part-whole relations [10]:

- *representing part-whole for individuals*,
- *representing a part-whole hierarchy* using classes.

However, there are much more various kinds of part-whole relations ([8]), eg. place-area, member-bunch, member-partnership, material-object etc.[6] On the other hand, part-whole relations have a relatively limited scope of domains: spatial, temporal or structural (e.g. company departments). We plan to investigate symptoms of diverse kinds of those part-whole relations in OWL ontologies.

Part-whole relations use case. In this example (Figure 3), we consider the concepts 'Committee','ProgramCommittee' and its part as 'ProgramCommitteeMember' in the

[5] An example of named semantic relation was also referred to in the parallel taxonomies use case.

[6] Part-whole relations are a subject of interest in philosophy as *mereology* and in linguistics as *meronymy*.

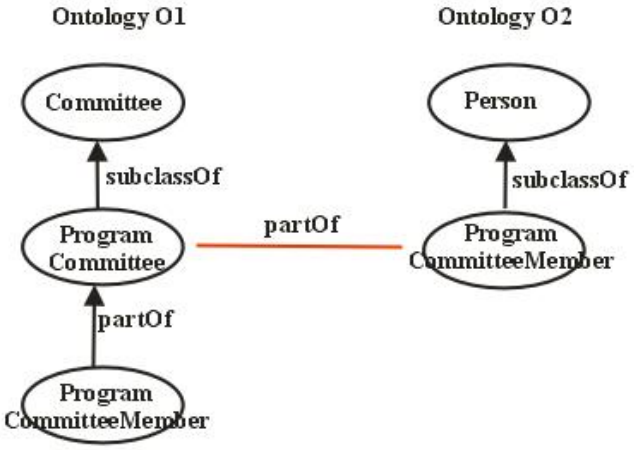


Fig. 3. Example of part-whole relation

ontology O1, while in the ontology O2 we also have 'ProgramCommitteeMember' as subclass of 'Person'. In this case, we suggest the named semantic relation 'partOf' due to equivalence correspondence between the 'ProgramCommitteeMember' concepts. This pattern leads to triangular matching, where we can enrich ontology O2 with the part-whole relation while the ontology O1 with the kind-of relation.

3 Reified N-ary Relations

In OWL a property is always a binary relation. However, sometimes it is a natural need to link more than two individuals in one relation. In that situation it is appropriate to approximate an n-ary relation, i.e. a relation connecting an individual to more than just one individual or value. This topic is again covered by a SWBPD note [7], on which we partly base our consideration about this semantic structure. We think that the most suitable pattern for expressing an n-ary relation is to introduce a new class for a relation. It means that this class is the constituent element for expressing a relation, thus it can be matched with a property from a different ontology. There are several use cases illustrating situations where this introduction of artificial (reified) class can be useful: for expressing *additional attributes describing a relation*, for expressing *different aspects of the same relation*, or for expressing a relation with *no distinguished participant*.

N-ary relations use case We again use an example from the conference organisation domain. A reviewer writes a review for a particular paper (see Figure 4). This is a typical example of n-ary relation. A designer can model this situation in many ways. Two of them are in Figure 4 (a) and (b). Situation (a) is modelled according to the SWBPD n-ary relation pattern. However, the designer need not exploit this kind of pattern and can model it intuitively as depicted in the (b) part. The (b) situation uses two distinct binary relations.

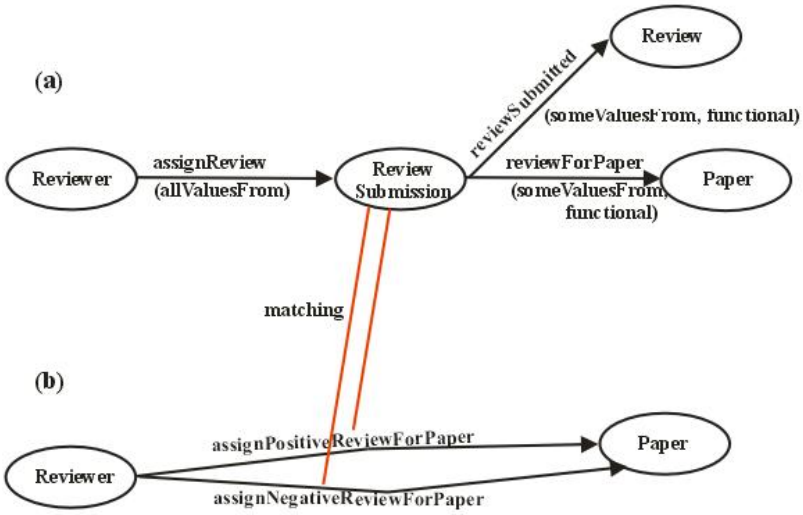


Fig. 4. Example of n-ary relation

3.1 Patterns for N-ary Relation Discovery

This motivation example also suggests a possible and useful heterogeneous matching. However, we foremost need to detect the n-ary relation itself. The detection of n-ary relation can exploit diverse symptoms. For an n-ary relation it should hold that it cannot be split up into binary relations, because they are constituents of it and these relations are all intertwined in some way. It means that all these relations should occur with appropriate concepts in the A-Box (i.e. instance-level knowledge). This would be very helpful, but instances of entities are often unavailable together with ontologies. Therefore we focus about two sources of information:

- *Structural pattern* that forms a structural bunch (see Figure 5) containing at least two relations having the concept Y in their the domain and at least one relation having the concept Y in its range.
- *Naming pattern*, materialised by the average token-based similarity measure c between the name of 'RelationX' and names of other entities from structural bunch.

3.2 Data Acquisition, Pre-Processing and Mining

In order to acquire a high number of ontologies, we applied the *Watson tool*[7] via its API. Then OWL API[8] is then called in order to obtain information about relations and their domains and ranges. A simple method of *tokenisation* was applied fully automatically. The algorithm for pattern-based discovering of n-ary relations within an ontology can be briefly described as follows:

[7] `http://watson.kmi.open.ac.uk/editor_plugins.html`
[8] `http://owlapi.sourceforge.net/`

1. Relations and their domains/ranges are first grouped into structural bunches according to the structural pattern described above.
2. The naming pattern is applied: for all different 'RelationX' an average token-based similarity measure c is computed as the ratio of the number of entities from a structural bunch that share a token with 'RelationX' over the number of all distinct entities from the structural bunch except X and 'RelationX'.
3. A structural bunch with average token-based similarity measure higher than a certain threshold σ is taken for an n-ary relation.

The naming pattern is based on the assumption that n-ary relations actually shift the ranges of the original relation (RelationX) having reified relation in its range further to connected relations having the reified relation as a domain. This is often reflected by the naming style through the n-ary relation approximation.

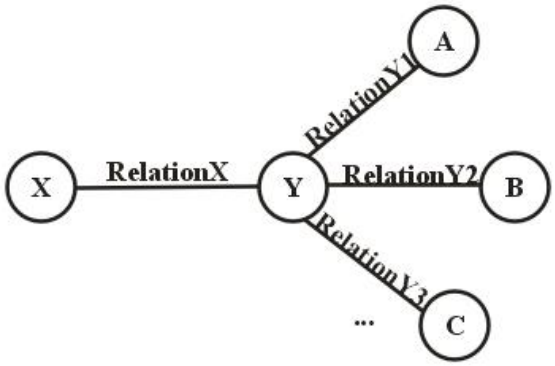


Fig. 5. Structural bunch of candidate for n-ary relation, where Y is a candidate for reified relation and 'RelationX' has a reified relation in its range.

3.3 Initial Experiment

So far we only carried out a preliminary experiment concerning the precision of n-ary relation discovery. 10 ontologies have so far been processed, with the σ threshold heuristically set to 0.4. The ontologies have been 'randomly' taken from the collection of 477 ontologies collected via the Watson tool. In Table 1 there are numbers of *true positive* (TP) examples, *false positives* (FP) examples, and *precision* (P) computed in the usual way.[9]

We can see that the application of the naming pattern after the structural pattern increased precision from 12% up to 50%, though obviously at the expense of recall (3 TPs were lost). The manual evaluation was rather coarse-grained and subjective,

[9] Note that computing the recall even for such a small number of ontologies would be much harder, as it would require to thoroughly and exhaustively analyse the ontology content.

	TP	FP	P
Structural pattern	6	51	12.2%
Naming pattern	3	6	50.0%

Table 1. Precision of detection for N-ary relations for structural pattern and for naming pattern applied after structural pattern.

because some domain ontologies have obscure conceptualization. However, it seems that the application of both patterns would be a better choice for most use cases; some tuning of the σ parameter could also help. In all, these preliminary results show that the detection of n-ary relation is doable in principle, at least for some cases.

For better insight we can look at one positive example (satisfying both the structural and naming pattern) and one negative example (satisfying the structural pattern only). Figure 6 depicts the situation where the Process can have certain Effect under certain Condition.[10] At the first sight it is a clear example of n-ary relation. In this case, the reified relation is ConditionalEffect and 'RelationX' is 'hasEffect', with $c = 0.6$. Figure 7 depicts the concept of 'Date' connected via binary relations with the concepts 'gYear', 'gMonth' and 'gDay'.[11] There is no specific relation among those concepts, and they are designed as set of binary relations. In this case, the false reified relation is 'Date', and 'RelationX' is 'date' with $c = 0.14$ (i.e. far below the heuristic threshold). We can also look back to the n-ary relations use case again. The algorithm will discover 'ReviewSubmission' from the situation (a) as a reifeid relation, since 'assignReview' as 'RelationX' has $c = 0.8$.

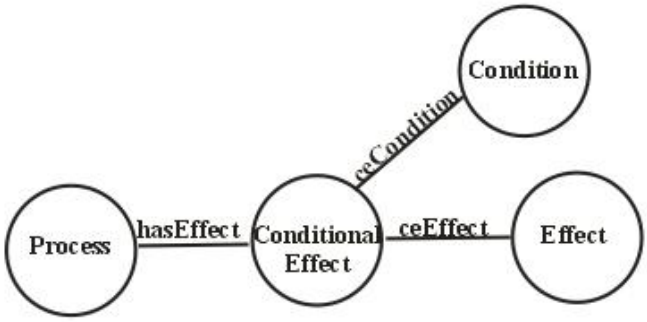


Fig. 6. TP example of detection of n-ary relation

[10] This example has been automatically discovered in the ontology residing at `http://www.mindswap.org/~bparsia/ontologies/sws/owls1.1/Process.owl`

[11] This example has been automatically discovered in the ontology residing at `http://oaei.ontologymatching.org/2004/Contest/103/onto.rdf`

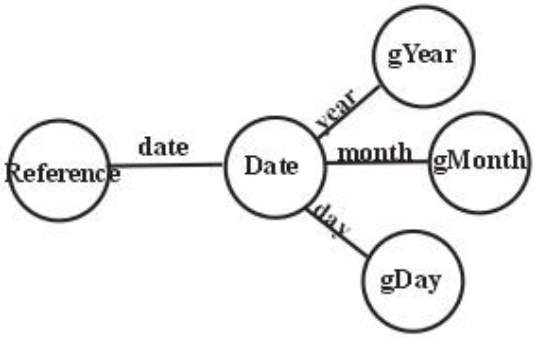


Fig. 7. FP example of detection of n-ary relation

3.4 Matching N-ary Relations Across Ontologies

We can imagine the situation (say the case 1) where both ontologies have n-ary relations, or (the case 2 where) the ontology 1 includes an n-ary relation, while the ontology 2 contains binary relations corresponding to the particular n-ary relation from the ontology 1. In both cases, it makes sense to create an alignment; either of the reified relation to the reified relation (the case 1), or of the reified relation to binary relation/s (the case 2). The latter case corresponds to the n-ary relations use case situations (a) and (b).

Let us outline a method of creating such an alignment. Regarding the matching and the case 1, assume that we have discovered the reified relation in the ontology 1 (A) and another reified relation (B) at the same time in the ontology 2. Then we might be able to reconstruct all components of those reified relations by taking appropriate concepts and relations from the corresponding structural bunch. First, we can compute a string_based similarity measure for each pair of concepts[12] of two reified relations (A and B). If there are, for all concepts from A, satisfactory (i.e. similar above a certain, high, threshold) concepts from B, we should also try to compute a string-based similarity measure for each pair of relations from the reified relations. If even this is succesfully done, A and B are matched as reified relations as a whole structures.

Regarding matching and the case 2, assume that we have discovered a reified relation (A) in the ontology 1 and there is no reified relation discovered in the ontology 2 with our abovementioned method. Still, we can try to find out binary relations from ontology 2 corresponding to the reified relation from the ontology 1. Similarly as in the previous case, we will reconstruct all components of the reified relations from the ontology 1 by taken appropriate concepts and relations from the corresponding structural bunch. In this case 2 for ontology 2, we are not limited to several concepts as in the case 1. First, we have to find out the most (string-wise) similar concepts from the ontology 2 (used in the domain/range of some relation r) to the concept that is in the domain of

[12] I.e. the concepts from the domains and ranges of the structure, except the reified relations themselves.

the reified relation.[13] Next, we also have to compute a string-based similarity measure of the concept from the domain/range of relation r to the concept from a range of reified relation A. If this similarity is satisfactory (i.e., above a certain, high, threshold), we should also try to compute a string-based similarity measure between the relation r from ontology 2 and the reified relation A from ontology 1. If even this is succesfully done (the similarity measure is high enough), A and r are matched as reified relation to binary relation (heterogeneous matching).

As string-based similarity measure, any common measure such as Jaccard, Char-Jaccard, Levensthein etc.[14] can be used. We can also exploit results of some off-the-shelf OM tools that even possibly combine multiple measures. It means that after having discovered the reified relation(s) in (two) ontology/(ies) we can directly use the results of the OM tool for comparing the similarity measures of concepts/relations. The question of the level of threshold is still open and it is the matter of future experimentation.

4 Related Work

Our work is strongly related to [11] where correspondence patterns are described as helpers to model ontology alignment. They are classified, according to their usage, into generic patterns and application-specific patterns. Furthermore, they are classified based on the type of entities they are to be applied on. These correspondence patterns are published in a pattern library on the web.[15] Our semantic structures represent parts of potential correspondence patterns. From this point of view, our work can potentially extend the pattern library with new patterns. On the other hand, as one of our goals is to match whole semantic structures, it also deals with methods for instantiating these correspondence patterns.

Most of the OM tools find out homogeneous mapping, i.e. class-to-class, property-to-property, or instance-to-instance. However, the notion of matching heterogeneous entities (such as class-to-property or instance-to-class) has also been addressed [4] as a theoretical issue. Obvious reason of this kind of matching lies in modelling issues, as the same concept can be modelled using a class or a property (class-centric or property-centric design approach). Most of semantic structures described above could lead to heterogenous matching.

In [6] authors propose 'block matching'. They argue that sometimes it makes sense to match whole structures and not only isolated entities. This consideration is in accordance with our work. They authors of [6] worked out an algorithm for finding these blocks, and then they try to match them. In contrast with this approach, we do not partition the whole ontology. In our case, we try to detect semantic structures that represent snippets (semantic structures) of ontologies that can be potentially matched with other snippets from different ontologies.

[13] Actually, we can alternatively start with the range. The domain of a reified relation means that this concept is in the domain of a relation having the reified relation in its range. Analogous situation holds for the notion of range of a reified relation.

[14] For an overview of measures commonly used in OM see e.g. [3].

[15] `http://www.omwg.org/TR/d7/patterns-library/`

As we focus on ontology matching via pattern-based detection of semantic structures, this approach can also be seen as an OM that uses background knowledge. Usually, background knowledge for an OM task is some ontology that can be used for matching two other ontologies from the same domain (e.g. in [2]). As we mentioned in the introduction, a few systems take advantage of logic aspects of ontologies. S-Match [5] system does so-called *semantic matching*, which is based on analysing the set-theoretic meaning of concepts in ontologies.

As we already mentioned, our current work is based on [12], where we concentrated on patterns potentially indicating semantic structures (e.g. see section 2.1) and, in particular, possible errors in *individual* ontologies. The detection of these patterns can be helpful for ontology evaluation and refactoring. Later on, we succesfully verified that refactoring operations applied on errors detected via such patterns can improve results of OM systems [13]. This constitutes an approach to pattern-based improvement of OM that is parallel to the approach described in the current paper.

5 Conclusions and Future Work

Our long-term interest is in ontology matching via pattern-based detection of semantic structures. In this early phase we outlined different kinds of semantic structures in which we are interested, and we paid specific attention to one of those semantic structures, n-ary relations. Since n-ary relations are not directly supported in the OWL language, ontology designers have to choose a way how to represent such relations. They can take advantage of the 'best practices' formulated by the W3C SWBPD working group, namely, an n-ary relation can be reified as a concept linked by binary relations with other concepts from the n-ary relations semantic structure. Furthermore, there are also recommendations regarding the naming style. Alternatively, they can also represent it intuitively on their own. In our approach we tried to detect n-ary relations represented not only according to the 'SWBPD' patterns. We evaluated our approach, in terms of precision, on 10 ontologies. The preliminary small-scale results are promising, however, there is an ample space for further improvements. Three other semantic structures (parallel taxonomies, value partitions and part-of relationships) have been briefly mentioned and described using use cases.

Naturally, as it is an initial phase of research, there is a lot of future work to be done. Regarding value partitions and part-whole relations, we plan to deeper investigate these semantic structures with regard to their *occurrence* in OWL ontologies. For all semantic structures, we have to work out ways to discover their *instances* in OWL ontologies. Next, it should also be specified under what conditions the same kind of semantic structures can be *matched* (e.g., what threshold should be used for matching n-ary relations). These goals also include: *formal description* (as an extension of [12]) of semantic structures relevant to the OM task, scalable implementation of *automatic detection* of such semantic structures, and scalable implementation of *automatic ontology matching* using diverse semantic relations (i.e. not only the equivalence relationships). Another specific goal is to track the possible subsequent usage of discovered matching of semantic structures in the context of diverse types of *applications*. There are specific requirements of

each of these types of applications, which can increase or decrease the usefulness of semantic structure matching.

Acknowledgments

The research has been partially supported by the IGA VSE grant no.20/08 “Evaluation and matching ontologies via patterns”.

References

1. Web Ontology Language (OWL). Online http://www.w3.org/2004/OWL/.
2. Aleksovski Z., ten Kate W., van Harmelen F.: Exploiting the Structure of Background Knowledge Used in Ontology Matching. In: Ontology Matching workshop (OM-2006) at ISWC-2006.
3. Euzenat J. and Shvaiko P.: *Ontology matching*. Springer-Verlag. 2007. ISBN 3-540-49611-4.
4. Ghidini, C., Serafini, L.: Reconciling concepts and relations in heterogeneous ontologies. In: Proc. ESWC-2006, Budva, Montenegro, 2006.
5. Giunchiglia F., Shvaiko P., Yatskevich M.: S-Match: an Algorithm and an Implementation of Semantic Matching. In: Proc. ESWS-2004.
6. Hu W., Qu Y.: Block Matching for Ontologies. In: Proc. ISWC-2006.
7. Noy, N., Rector, A. (eds.): Defining N-ary Relations on the Semantic Web. W3C Working Group Note 12 April 2006, online at http://www.w3.org/TR/swbp-n-aryRelations/.
8. Odell J. J.: Six different kinds of composition. In: *Advanced Object-Oriented Analysis & Design using UML*. Cambridge University Press, Cambridge. 1998.
9. Rector, A. (ed.): Representing Specified Values in OWL: ”value partitions” and ”value sets”. W3C Working Group Note, 17 May 2005, online at http://www.w3.org/TR/swbp-specified-values/.
10. Rector, A., Welty, C. (eds.): Simple part-whole relations in OWL Ontologies. W3C Editor’s Draft 11 Aug 2005, online at http://www.w3.org/2001/sw/BestPractices/OEP/SimplePartWhole.
11. Scharffe F., Fensel D.: Correspondence Patterns for Ontology Alignment. In: Proc. EKAW-2008.
12. Šváb-Zamazal O., Svátek V.: Analysing Ontological Structures through Name Pattern Tracking. In: Proc. EKAW-2008.
13. Šváb-Zamazal O., Svátek V., Meilicke C., Stuckenschmidt H.: Testing the Impact of Pattern-Based Ontology Refactoring on Ontology Matching Results. In: Ontology Matching workshop (OM-2008) at ISWC-2008.