Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 35 (2014) 504 – 513

Procedia Computer Science

18th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems - KES2014

A method of identifying ontology domain

Dan Wu*, Anne Håkansson

Software and Computer Systems, ICT School, Royal Institute of Technology KTH, Sweden

Abstract

Metadata, such as the domain description and the purpose of an ontology, can be used to describe the context of ontologies for ontology integration. However, these metadata are not always available in ontologies. To solve the problem, a method is developed to automatically discern the domain of an ontology. This method uses a so-called core domain ontology, rules and an ontology reasoner to identify the domain. The core domain ontology is a light weight ontology that consists of the essential concepts of a domain. Rules and the ontology reasoner are used to test if the core domain ontology is consistent with an ontology for which the domain needs to be identified. If the two ontologies are not in violation, then the method confirms the consistency between them, that is, the test ontology shares the same domain as the core domain ontology. If the core domain ontology shows inconsistency with the test ontology, they do not share the same domain and then the ontology can be used to compare with another core domain ontology. Experiments on the core domain ontology for the conference domain show good results. Ten ontologies of mixed domains are compared with the core conference ontology. Eight ontologies' domain are correctly identified, out of which, four ontologies are identified as sharing the same ontology domain.

Keywords: domain identification of ontology; metadata of ontology and ontology match and integration

1. Introduction

Metadata of ontologies are stored in an ontology repository [1-3] and used to improve the use of ontologies on the semantic web. In our previous research, metadata of ontologies are extracted to support building context rules to provide information for ontology integration [4]. However, many ontologies lack such information [1; 3]. To

* Corresponding author.
E-mail address: dwu@kth.se

tackle the problem, a method is developed and applied to identify domain information of an ontology. If the domains of ontologies are available, other ontologies that are in the same domain may be used as the contextual information for the semantic ontology integration. Therefore, extra information beyond the integrating ontologies can be used to refine the ontology integration [4].

Context is considered important for building adaptive systems [5;6]. In order to build context and provide contextual information for ontology integration, the metadata of ontologies, such as domain description and purpose, is needed. This paper proposes a method for identifying the domain of an ontology. The following text is organized in six sections: section two describes the related work; section three describes the proposed method, the core domain ontology and the rules used by the method; section four explains the experiment; presents the result and discusses the method and the experiment; and finally, section five concludes the work and points out future work.

2. Related work

To our best knowledge, there is no related work in the area of identifying the domain of an ontology to build context for ontology integration. However, the ontology domain as metadata and algorithms for metadata generating, have been studied in related areas, namely ontology reuse.

Simperl *et al.* [1] investigate how metadata are used for research in ontology engineering, ontology repository and the semantic web tools. The commonly used metadata are the description of ontology domain, availability and licensing conditions, development status, the release date, and also the structure data of ontologies. Two ontology search engines Swoogle[†] and Watson[‡] provide a small amount of ontology metadata, such as URI, ontology format, the size of a file, comments, number of classes, properties and individuals, can be automatically extracted. Beside these metadata, Swoogle extracts data about encoding, ontology ratio, number of statements and information about the ontology discovery process. Watson, on the other hand, shows user reviews, locations and imports. These metadata are quite different from the metadata of ontology domain.

To overcome the difficulty of metadata acquisition, Simpler *et al.*[1] proposed an approach for automatic metadata acquisition, called OMEGA. First, a parser harvests metadata elements from the tags of an ontology. Secondly, several ontology metadata vocabulary (OMV) elements are derived with the help of Google APIs[§], or Wikipedia Categorical Index[**]. Furthermore, an inference engine, such as Pellet[††], can be used to extract metadata automatically. Finally, existing metadata information is reused from online ontology repositories, such as the DAML Library, and search engines Swoogle and Watson.

The OMEGA acquisition tool contains an algorithm that automatically returns *hasDomain*. To discover the domain of an ontology, keywords and key classes are extracted from the ontology, which are compared with DMOZ[‡‡] web category. The weight of a keyword k depends on the highest category level that k appears and it's frequency as the category name. If one keyword appears in sever sub-categories, a maximum likelihood algorithm is applied. For each keyword, the algorithm computes full category paths of the sub-categories that match the keyword, and then it counts the number of distinct top categories to calculate the likelihood of the related domain. The highest likely top DMOZ category based on all the keywords represents the domain. DMOZ is a large and comprehensive human-edited directory of the web, and the domains are represented by words in a hierarchy.

[†] http://swoogle.umbc.edu/
[‡] http://watson.kmi.open.ac.uk/
[§] http://code.google.com
[**] http://en.wikipedia.org/wiki/Portal:Contents/Categorical_index
[††] http://www.mindswap.org/2003/pellet/
[‡‡] http://www.dmoz.org/

Our method is also developed for domain acquisition. However, we rely on core domain ontologies, not the DMOZ web category. Each domain is represented by a core domain ontology that consists of hierarchy of classes and class relations. The difference between comparing keywords and ontologies distinguishes the OMEGA method from our domain identifying method.

3. The domain identification method

In the method, core domain ontologies (CDOs) are used to represent different domains. More specifically, each core domain ontology (CDO) represents one domain by using concepts (classes) and concept relations (properties) of the domain. The CDOs are compared with other ontologies by applying rules and an ontology reasoner. The rules are developed to integrate ontologies and to test for contradiction between the core domain ontology and another ontology, in which the domain is not annotated (hereafter called the test ontology). The ontology reasoner reasons with the content of the CDO and the test ontology to produce a class hierarchy. The class hierarchy can only be created when the classes of ontologies are consistent.

To compare the ontologies, the domain identification method applies reasoning that follows *SROIQ* [7], an extended Description Logic (DL). *SROIQ* is a result from *SHOIN*, which is an extension of description logic underlying OWL-DL, with additional expressive means on classes and properties, such as complex role inclusion axioms and constructs. Hence, the method can handle ontologies that contain relations, such as subsume (⊑), equivalence (≡), conjunction (⊓), disjunction (⊔), negation (¬), and universal restriction (∀), existential restriction (∃) and inverse properties (⁻). Two special classes are included: owl:Thing (⊤), which represents the set of all individuals, and owl:Nothing (⊥), which represent an empty set. For a property R, two relations are included, domain and range. If *R* has domain *C* and range *D (*denoted as *R⊑C℃D)*, it means two individuals (*c, d*) are related by *R*, and *c*ϵ*C* and *d*ϵ*D*. A limitation of the method, though, is that a part of SROIQ logic is excluded, e.g., reasoning on individuals, i.e., instances of classes and properties, is not included.

As the result of a comparison, the method returns *True* or *False*. If the core domain ontology is found "not contradicting" the test ontology, the method returns *True*, which means that the test ontology shares the same domain as the core domain ontology. Otherwise, the method returns *False*. The following sections explain the whole procedure, the core domain ontology, the threshold and the rules in details.

3.1. The core domain ontology

A core domain ontology catches the central concepts (classes) and relations (properties) over a domain. Even ontologies about the same domain can be heterogeneous because of various interests, authors' perspectives, different purposes and applications' context. It is expensive to create a complete ontology covering a domain because of economy, time and other resource consumption, as well as the ever-changing world. Therefore, we use a core domain ontology in the method. A core domain ontology does not cover a complete definition of classes and properties of a domain, but it contains the essential and basic classes of a domain. Hence, a core domain ontology is a light weight ontology that contains only classes when being designed and developed. It, however, can be extended later with some classes and properties from confirmed domain ontologies, i.e., they have been tested and confirmed as sharing the same domain.

As an example, we use a core domain ontology of *conference* with seven classes, i.e., *Document, Paper, Abstract, Review, Person, Author and Reviewer*. The relations between classes are hierarchical and distinct from each other, see Table 1.

Table 1. The core conference ontology

Abstract $\sqsubseteq$Paper; Paper $\sqsubseteq$Document; Review $\sqsubseteq$Document; Author $\sqsubseteq$Person; Reviewer $\sqsubseteq$Person; Review $\sqcap$Paper $\sqsubseteq \perp$; Person $\sqcap$Document $\sqsubseteq \perp$

The table shows that *Abstract* subsumes under *Paper* and *Paper* is subsumed to *Document* and so on. And the last two parts show classes *Review* and *Paper* and classes *Person* and *Document* are distinct and do not overlap with each other.

To describe the domain, the core domain ontology should also use properties. These properties are included with the help of classes, to thereby extend core domain ontologies with other domain ontologies. When a test ontology is confirmed sharing the same domain as the core domain ontology, the similar classes are used to extend the core domain ontology, the properties that are connected to the similar classes are included. For example, if a property *hasAuthor* is in a conference domain ontology, and the property is related to class *Paper,* as the domain, and *Author,* as the range, it is then included in the core domain ontology, since both *Paper* and *Author* are similar classes between the domain ontology and the core domain ontology. In such way, core domain ontologies are kept basic and essential.

3.2. The procedure of the domain identification method

The domain identification method tests if the core domain ontology contradicts the test ontology, see Figure 1. If the core domain ontology contradicts the test ontology, it returns *False,* which means that domain of the test ontology is different from CDO's domain. If the core domain ontology does not contradict the test ontology, it returns *True,* which means that the test ontology shares the domain as *CDO*. The comparison direction is from the core domain ontology to the test ontology. The inputs of the procedure are a test ontology *O* and a core domain ontology *CDO,* and the output is *True* or *False*.

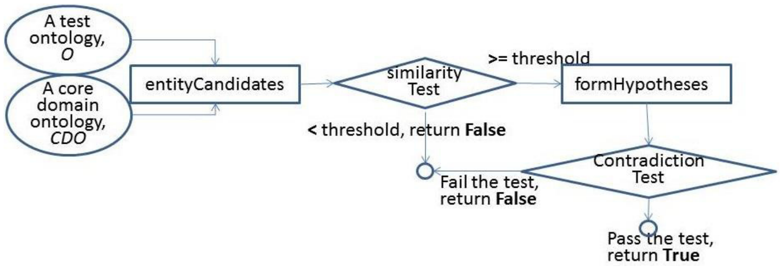
A test ontology, O
A core domain ontology, CDO
entityCandidates
similarity Test
>= threshold
formHypotheses
< threshold, return False
Contradiction Test
Fail the test, return False
Pass the test, return True

Fig. 1: the domain identification procedure

There are four steps in the procedure: extracting entity candidates, conducting a similarity test, forming hypotheses and testing for contradictions of ontologies. In the first step entityCandidates, the candidates, i.e., classes and properties are extracted from ontologies, O and CDO, which results in two sets of classes and two sets of properties. These sets are the input to the next step, similarityTest. Classes are compared separately from properties. Considering the comparison direction, we need to find the similar classes in *O* for each class of *CDO.*

Similar classes and properties, i.e., string-identical, synonyms and substring-contained classes and properties are distinguished and matched. The percentage of similar classes of *CDO* to *O* is calculated and compared with a pre-configured number, so-called threshold value. The threshold of the similar class is the least expected percentage of similar classes shared between the core domain ontology and the test ontology. If the percentage exceeds the threshold value, the procedure continues to the next step. Otherwise, it returns False and the procedure stops.

If one class C_{CDO} of the core domain ontology matches more than one similar classes in the test ontology, e.g. $Co_{i,} ..Co_j$, the distances between C_{CDO} and each similar class in ($Co_i, ... Co_j$) are measured. A score of 1, 2, 3 or 4 is assigned to each pair of the similar class, e.g., (C_{CDO}, Co_i) and (C_{CDO}, Co_i), according to the following conditions:

- If string identical classes, 1 is assigned, meaning the shortest distance;
- If synonyms, 2 is assigned, implying that (C_{CDO}, Co_i) are synonyms;
- If substring-contained classes, 3 or 4 are assigned. Substring-contained similar classes are divided into two groups: one includes string identical names or synonyms of classes in the class hierarchy; the other group is combined with other words. The first group is assigned with 3 and the last group is assigned with 4. For example, a class *"Abstract"* in the core domain ontology, two classes, "*abstract_of_paper*" and "*abstract_submission_date*" are found in the test ontology. Then, "*abstract_of_paper*" has a shorter distance than "*abstract_submission_date*", since "*paper*" is in the *"Abstract"* class hierarchy, but not "*submission date*". Therefore, the distance between "*Abstract*" and "*abstract_of_paper*" is 3 and to "*abstract_submission_date*" is 4.

If several substring-contained classes contain several other words, the similarity distance is compared among them. The class with the shortest distance will be assigned with a number in order to minimize the computation. The similarity is calculated according to the function below, where *t* is the longest shared substring of c_1 and c_2,$f\colon \mathfrak{H} \times \mathfrak{H} \rightarrow [0,1], \forall c1, c2 \in \mathfrak{H}$.

$$f(c1, \quad c2) = 2|t|/(|c1| + |c2|)$$

The class with the highest score returned from the above function is the most similar class and is assigned 4. However, there are situations when one class matches several equally scored classes, for example, when there is more than one synonym for one class. All these equally scored classes need to be tested, and hence, all the alternatives of similar classes will be tested in the next step, separately.

In the step of formHypotheses, hypotheses for classes are developed. A hypothesis is an equivalence relation between classes. For example, if class C_{CDO} is similar to class C_O, then, the hypothesis is $C_{CDO} \equiv C_O$. From the previous step, pairs of similar classes are returned. For each pair of similar classes, a hypothesis is developed. If a class matches to more than one similar class, each match is developed into a hypothesis and is tested separately. For instance, class C_{CDO} matches to C_{O1} and C_{O2}, two hypotheses are generated: $C_{CDO} \equiv C_{O1}$ *and* $C_{CDO} \equiv C_{O2}$. And the two hypotheses are tested separately.

A set of hypotheses is, then used in the next step, contradictionTest. The hypotheses, the core domain ontology and the test ontology are tested with a reasoner. The contradiction test checks that the definitions of classes and properties are satisfiable. In another word, an interpretation I of model is searched to satisfy the union of assertions of the test ontology *O*, the core domain ontology (*CDO*) and the hypotheses (*H*) and rules (*R*), i.e., $I \models O \sqcup CDO \sqcup H \sqcup R$. The hypotheses are falsified if conflicts are raised form the test; otherwise, we say that the contradiction test has passed.

If the test fails, the core domain ontology contradicts the test ontology, the contradiction test returns ***False***, otherwise it returns ***True***. If there are several alternatives of similar classes in the step similarityTest, these alternatives will be formed as sets of hypotheses in the step, formHypotheses, which are tested in the contradiction test. If one test of a set of hypotheses succeeds in the contradiction test, a true value is returned. Hence, the goal of identifying the domain of the test ontology is reached and the procedure stops. Rest of the hypotheses will not be tested since they do not contribute more information in discerning the domain of the test ontology.

3.3. Threshold for comparing the similar classes

The similar classes that are shared by the compared ontologies are extracted in the similarityTest of the method, as mentioned above. String-identical classes (classes' names that are identical), synonyms of classes and string-contained classes (one class's name is a subsumed string of another class) are similar classes. WordNet is used for the synonym check.

For each class, in the core domain ontology's class set (denoted as SC_{CDO}), synonyms are found in WordNet. The synonyms, the string identical classes and the string-contained classes are identified in the test ontology's class set (denoted as SC_O) and saved in an array. Thereby, $SC_{CDO} \rightarrow SC_O$ represents the set of similar classes from SC_{CDO} to SC_O. The percentage of the similar class is calculated according to the function below:

$$(|SC_{CDO} \rightarrow SC_O|/|SC_O|)*100$$

The result of the function is compared with a default threshold, which is a pre-defined number ranging from 0-100%, depending on the expected result. The higher the threshold is set to, the more similar classes are required. If the threshold is increased, the comparison of the core domain ontology with the test ontology will be stricter. In the worst case, this can lead to so-called false negative cases, where the test ontology is wrongly classified as not sharing the domain as the core domain ontology. For example, if the threshold is set to 90%, the test ontology needs to contain at least 90% similar names of classes as the core domain ontology.

If the threshold is too low, the comparison can lead to false positive cases, where a test ontology is falsely classified as sharing the domain with the core domain ontology. For example, if the threshold is set to 10%, the test ontology needs to contain only 10% similar names of classes as the core domain ontology, which can lead to wrong conclusion. Since it is a configurable number, the threshold should be adjusted according to situations. In our experiment, the threshold is set to 50%.

3.4. Rules

Rules are used to resolve conflicts and to generate the desired results during the contradiction test. An example of conflict is, when 1) class A_1 is defined distinct to class B_1 in ontology O_1, i.e., $A_1 \sqcap B_1 \sqsubseteq \bot$, and 2) A_2 is subsumed under B_2 in ontology O_2, i.e., $A_2 \sqsubseteq B_2$, and 3) when classes A_1 and A_2, B_1 *and* B_2 are found similar, they are set equal, then these axioms will cause conflicts in the reasoner.

Such conflicts can be resolved with rules. Only the explicit inconsistencies in two ontologies make conflicts. If one ontology has more or less strict definitions of a class than another ontology, they do not generate conflicts. Thus, we assume that they are created with different perspectives and for different purposes, and therefore do not necessary contradict each other. Here rules, used for our experiment, are presented and explained. Hereafter, O_1 represents the CDO, and O_2 represents the test ontology.

Rule 1: The more specific ontological definition of classes does not contradict the less specific definition.

Rule 1-1: There are definitions in O_1: A_1 subsumes B_1 and B_1 subsumes C_1, and in O_2: A_2 subsumes C_2 and B_2 subsumes C_2. And classes A_1 and A_2, B_1 and B_2 and C_1 and C_2 are similar classes. Then they are made equal, i.e., $A_1 \equiv A_2$, $B_1 \equiv B_2$ and $C_1 \equiv C_2$, which are formed as hypotheses. With these conditions in the rule, it unites the similar classes and generates subsumed relations among the classes, $A_1 \sqcup A_2 \equiv A$ and $B_1 \sqcup B_2 \equiv B$ and $A \sqsubseteq B$ and $C_1 \sqcup C_2 \equiv C$ and $B \sqsubseteq C$, see below:

If in O_1: $A_1 \sqsubseteq B_1$ and $B_1 \sqsubseteq C_1$; and in O_2: $A_2 \sqsubseteq C_2$ and $B_2 \sqsubseteq C_2$; and $A_1 \equiv A_2$, $B_1 \equiv B_2$ and $C_1 \equiv C_2$
then $A_1 \sqcup A_2 \equiv A$ and $B_1 \sqcup B_2 \equiv B$ and $A \sqsubseteq B$ and $C_1 \sqcup C_2 \equiv C$ and $B \sqsubseteq C$

Rule 1-2: This rule describes two classes that are defined distinctly from each other in an ontology. The similar classes in another ontology defines only the classes without any relation. As a result, the classes are united and the relation is kept as distinct, $A_1 \sqcup A_2 \equiv A$ and $B_1 \sqcup B_2 \equiv B$ and $A \sqcap B \sqsubseteq \bot$, where $\bot$ means empty class, see below:

If in O_1: $A_1 \sqcap B_1 \sqsubseteq \bot$; and in O_2: A_2, B_2 and $A_1 \equiv A_2$, $B_1 \equiv B_2$
then $A_1 \sqcup A_2 \equiv A$ and $B_1 \sqcup B_2 \equiv B$ and $A \sqcap B \sqsubseteq \bot$

Rule 2: Property comparison. If the name of two properties R_1 and R_2 in two ontologies O_1 and O_2 are string-identical or synonymic, the relations of their correspondent classes are checked and the relation between R_1 and R_2 are deduced according to the relations of classes.

Rule 2-1: Class A_1 is the domain of R_1, and class B_1 is the range of R_1 in O_1; and class A_2 is the domain of R_2 and the range class is B_2 in ontology O_2. And if A_1 and A_2 are equal, and B_1 equals to B_2, then R_1 is equal to R_2. This is presented as:

If in O_1: $R_1 \sqsubseteq A_1 \circ B_1$ and in O_2: $R_2 \sqsubseteq A_2 \circ B_2$, and $A_1 \equiv A_2$ and $B_1 \equiv B_2$, ***then*** $R_1 \equiv R_2$

Rule 2-2: If A_1, B_1 and R_1 is defined in O_1 and A_2, B_2 and R_2 are defined in O_2, and the domain class A_1 of R_1 is subsumed under the domain class A_2 of R_2, and the range class B_1 of R_1 is subsumed under the range class B_2 of R_2, then the relation of R_1 is kept, as illustrated below:

If in O_1: $R_1 \sqsubseteq A_1 \circ B_1$ and in O_2: $R_2 \sqsubseteq A_2 \circ B_2$, and $A_1 \sqsubseteq A_2$ and $B_1 \sqsubseteq B_2$,
then $A \equiv A_1 \sqcap A_2$ and $B \equiv B_1 \sqcap B_2$ and $R \sqsubseteq A \circ B$

Rule3: This rule is specially designed to adjust the result of the threshold value. The problem is that even though a test ontology may share many similar classes as the core domain ontology, it does not describe any relations between those classes. In such situation, the method will let the test ontology pass the contradiction test since there will not be any explicit contradictions in the test. But this is a false positive case. To tackle this problem, rule 3 checks if those similar classes, in the test ontology, have relations with each other. If 50% of the discovered similar classes are not related to each other in the test ontology, this test ontology does not share the same domain as the core domain ontology, although the percentage of similar classes exceeds the threshold.

In order to calculate the percentage of the similar classes in the test ontology relate to each other, the function of ClassUsage in Protégé[§§] is applied. For each class *C*, if at least one other similar class appears in the ClassUsage, it counts that the class has a relation to other classes. If none other similar class appears in the ClassUsage, it counts that the class has no relation to other classes. Then, the percentage of the related similar classes (denoted as R*SC*) are calculated by summering all the classes that have relations to others, and divide the sum with the total number of the similar classes *TSC,* i.e., *(RSC/TSC)*100*. The rule is presented below:

If in O_1: $(C_{1\text{-}1}, C_{2\text{-}1}, \ldots, C_{i\text{-}1})$, and in O_2: $(C_{1\text{-}2}, C_{2\text{-}2}, \ldots C_{j\text{-}2})$ and $C_{1\text{-}1} \equiv C_{1\text{-}2}$, and $C_{2\text{-}1} \equiv C_{2\text{-}2}$ and … and $C_{i\text{-}1} \equiv C_{j\text{-}2}$ and *(RSC/TSC)*100 <=50%*
Then contradictionTest returns ***False***

An example of applying rule 3, if the core conference ontology defines class Paper as subsumes under class Document, while no relations in the test ontology are defined, and if more than 50% of similar classes lack of relations in the test ontology, then the test ontology does not define the conference domain.

4. The experiment

In order to examine how well the method works, a core domain ontology for conference is created, as presented in Table 1 in section 3.1. Ten ontologies of various domains are downloaded from the Semantic Web. Four ontologies, downloaded from the ontology alignment evaluation initiative (OAEI)[***], describe the conference domain. Six ontologies are fetched with the search engines, Swoogle[†††] and Watson[‡‡‡] by using the keyword *conference*. Out of these ontologies, two ontologies describe the conference domain and four ontologies describe other domains. The annotation of ontologies is used for determining the domain. If the domain information is clearly stated in the annotation, we use the annotations directly; otherwise, the domain is

[§§] http://protege.stanford.edu/
[***] http://oaei.ontologymatching.org/2012/conference/index.html
[†††] http://swoogle.umbc.edu/
[‡‡‡] http://watson.kmi.open.ac.uk/WatsonWUI/

decided subjectively by analysing the classes and properties of the ontologies. The test ontologies are presented in Table 2.

The last row in Table 2 describes the numbers of classes and properties in each ontology, which illustrate the sizes of the ontologies. The threshold in the experiments is set to the default value of 50%.

Table 2. The test ontologies

Ontology Name	Cmt	Confious	Sigkdd	ekaw	Music onto	Biblio graph	unspsc	university	webconf	conf
Source	OAEI	OAEI	OAEI	OAEI	Watson	Watson	swoogle	swoogle	swoogle	Watson
Ontology Domain	Conf erence	Conf erence	Conf erence	Conf erence	music	Biblio graphy	Product &service	university	Conf erence	Conf erence
Numberof Classes&Prop erties	85	109	66	107	197	107	6343	97	45	118

4.1 The results

The number of similar classes, the trigged rules, and the results returned from the domain tests are recorded in Table 3. In total, 80% of the test ontologies are correctly identified, while 20% are wrongly identified. Out of ten test ontologies, four test results returned True value and six returned False. Ontologies from OAEI describe the conference domain and tests of these ontologies returned all *true* value, which are correct answers. Of the six *false* values, two of them are incorrect answers, ontology 9 and 10. They are falsely classified because there are not enough similar classes are identified in them, compared to the default threshold. If the threshold is decreased to 40%, ontologies 5, 6, 9 and 10 will pass the threshold. But the test results of ontologies 5 and 6 will remain as false, since ontology 5 will trigger rule 3 that always return false and ontology 6 contains contradictions with the core conference ontology and, hence, will fail the contradictionTest.

Table 3. The test result

Ontology Number	***Ontology Name***	***similarClass (in percentage)***	***TriggedRules***	***ResultReturned***
1	cmt	7 (100%)	Rule 1	**True**
2	cofious	5 (71%)	Rule 1, 2	**True**
3	sigkdd	6 (86%)	Rule 1	**True**
4	ekaw	7 (100%)	Rule 1	**True**
5	musicO	3 (43%)		**False**
6	Bibliograph	3(43%)		**False**
7	Unspsc	5(71%)	Rule3	**False**
8	university	2 (29%)		**False**

9	webconf	3(43%)		**False**
10	watsonconf	3 (43%)		**False**

All rules work correctly in the tests. In ontology 7, rule 3 is used to falsify an ontology that shares 71% similar classes with the core conference ontology. Although the number of similar classes pass the threshold of 50%, the content of the ontology does not contain the required relations between classes, which is discovered by rule 3.

Another result is the extension of the core conference ontology with the confirmed domain ontologies, i.e., cmt, confious, sigkdd and ekaw. In Table 4, the right column shows the added definitions of properties and classes into the core conference ontology from the confirmed domain ontologies. This extension is not used in current tests. However, it can add value for the future tests.

Table 4. The result of extending the core conference ontology

OntologyName	Changes in the core conference ontology
cmt	Abstract≡PaperAbstract, writtenBy⊑Review°Reviewer, writtenBy≡writeReview^{-}, readPaper⊑ Reviewer°Paper, readByReviewer≡readPaper^{-}, submitPaper⊑ Author°Paper, markConflictOfInterest⊑ (Author or Reviewer)°Paper, writePaper⊑Author°Paper, writePaper≡hasAuthor^{-}, hasBeenAssigned⊑ Reviewer°Paper, hasBeenAssigned≡assignedTo^{-}, markConflictOfInterest⊑ Person°Document
confious	None
Sigkdd	submit⊑ Author°Paper
ekaw	authorOf⊑ Person°Document, authorOf≡writtenBy^{-}, hasUpdatedVersion⊑ Document°Document, updatedVersionOf≡hasUpdatedVersion^{-}, hasReview⊑ Paper°Review, reviewOfPaper≡hasReview^{-}, Author≡PaperAuthor

4.2 Discussion

Although we choose ontologies from different sources and of varied domains, the test results are difficult to generalize. There are some valuable issues worth discussing. Ontologies have about 100 axioms each, except ontology 7. Ontology 7 has more than six thousands axioms where 71% of its classes are identified as similar classes, which exceeds the threshold. Although rule 3 works well in this case, it is important to consider the size of the core domain ontology and the test ontology. The size proportion between the core domain ontology and the test ontology should be reasonable. This can be considered as an additional constraint in the method in the future.

We deliberately choose ontologies of related domains to conference, such as domains of bibliography and university. There are other close related domains that are not considered in the test, e.g., domain of journal and bookshop. In those similar domains, classes and properties may be similar or overlapping, which can cause the core domain ontology to work less efficiently. However, for a multi-domain system, several core domain ontologies will be used. Each test ontology will be tested against every domain ontology. In this way, the differences of several core domain ontologies will help to improve the result. Since a core domain ontology is used to define the essential classes of a domain, it constraints the content of the test ontology. Then, for each domain core ontology, the result of the test ontology will be further tuned. Theoretically, in a multi-domain system, this method adds more constraints in the test by using core domain ontologies for different domains, which can improve the test result.

In general, the method works better on OAEI ontologies, which are used for comparing ontology match systems. These ontologies are better controlled and considered to be objective. Other ontologies from the semantic web are less controlled. Although the authors of this paper checked the soundness of the ontologies, it is difficult to judge the quality without background information. The domains of ontology 8 and ontology 10 are subjectively judged by the authors. The method may show different results on other ontologies than used in the test.

It is noticeable that the threshold (50%) works fine considering the result, five ontologies (nr. 1,2,3,4 and 7) pass the threshold, all returned correct result; five ontologies (nr.5,6,8,9,and 10) are beneath the threshold, two of them are error cases. One weakness of this method is the ignorance of checking cross-representations between classes and properties. This indicates a need for an improvement of the similarity test. For example, class *Author* is defined in an ontology; whereas in another ontology, *author* is expressed as a property. The *author* property can relate class *Document* with class *Person*. If such similarities are not considered, some semantic similarities presented in classes and properties may be missed.

Another weakness is the lack of the possible perspectives of a domain, which is not covered in the core domain ontology. For example, classes *Author and Reviewer* are important concepts in a conference domain. However, there are conference ontologies on the semantic web that do not define them at all, for example in ontology 9 in the experiment. This causes a false negative result. The problem can be tackled in the future with extending core domain ontologies with perspectives. By extending a core domain ontology with confirmed domain ontologies, the diversity perspectives may be covered in core domain ontologies.

5. Conclusions and the future work

In this paper, a method is proposed and developed to identify the domain of an ontology. The method uses a core domain ontology and rules to distinguish if an ontology shares the same domain as the core domain ontology. A four-step-procedure is constructed for checking the similarities between classes and properties, as well as the contradiction of the contents of the compared ontologies. For evaluation, the method is applied on ten test ontologies, with the defined core conference ontology and rules. The results show that eight ontologies are correctly identified, among which, four share the conference domain and the other four are different from the conference domain; and two ontologies are wrongly identified. The results also illustrate an extension of the core conference ontology with properties and similar class definitions, which can improve the method.

In the future, it is interesting to study how the method works in a multi-domain environment, i.e., if several core domain ontologies are used, which means more constraints are introduced in the method. The two error cases are caused by the failure to recognize similar classes. It is, therefore, important to examine the level of the threshold to improve the result. The qualities of core domain ontologies are very important for the method; therefore, it is interesting to apply method like OntoClean to create domain ontologies to see the effects on the method.

References

1. A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of ContextAware Applications. Dey, Anind K., Salber, D. and Abowd, Gregory D. 2001. (2-4), 2001, Human-Computer Interaction (HCI) Journal, Vol. Volume 16, pp. pp. 97-166.
2. Formal Ontology and Information Systems. Guarino, N. 1998. Trento, Italy : IOS Press, 1998. Proceedings of FOIS'98. ss. 3-15.
3. Hartmann, J., Palma, R., Gómez-Pérez, A. 2009. Ontology Repositories. [red.] S., Dr. Studer, R. Staab. Handbook on Ontologies. Berlin, Heidelberg : Springer-Verlag, 2009, ss. 551-571.
4. Ontology metadata for ontology reuse. Simperl, E., Sarasua, C., Ungrangsi, R. and Bürger T. 2011. 2011, Int. J. Metadata, Semantics and Ontologies, s. vol (6) no(2).
5. Palma, Raúl; Hartmann, Jens; Bontas, Elena; Haase, Peter. 2007. OMV - Ontology Metadata Vocabulary for the Semantic Web. u.o. : OMV Consortium, 2007.
6. Porzel, Robert. 2011. Contextual Computing. Berlin Heidelberg : SpringerVerlag, 2011.
7. The Even More Irresistible SROIQ. Horrocks, Ian.; Kutz, Oliver; Sattler, Ulrike. 2006. The Lack District of the UK : u.n., 2006. Tenth International Conference on Principles of Knowledge Representation and Reasoning.
8. Wu, Dan. 2013. Ontology Integration with Non-Violation Check and Context Extraction. Stockholm : KTH Royal Institute of Technology, 2013.