

A Platform for Semantic Web Studies

Mathieu d'Aquin, Carlo Allocca and Enrico Motta

Knowledge Media Institute, The Open University

Walton Hall, Milton Keynes, MK7 6AA, UK

{m.daquin, c.allocca, e.motta}@open.ac.uk

ABSTRACT

The Semantic Web can be seen as a large, heterogeneous network of ontologies and semantic documents. Characterizing these ontologies, the way they relate and the way they are organized can help in better understanding how knowledge is produced and published online. It also provides new ways to explore and exploit this large collection of ontologies. In this paper, we present the foundation of a research platform for characterizing the Semantic Web, relying on the collection of ontologies and the functionalities provided by the Watson Semantic Web search engine. We more specifically focus on formalizing and monitoring relationships between ontologies online, considering a variety of different relations (similarity, versioning, agreement, modularity) and how they can help us obtaining meaningful overviews of the current state of the Semantic Web.

1. INTRODUCTION

With millions of semantic documents and ontologies made available online, the Semantic Web can be seen as a large-scale, distributed and evolving repository of interlinked knowledge. Several systems have emerged in the last few years that intend to give efficient access to such knowledge, including Semantic Web search engines such as Swoogle¹, Sindice², Falcon-S³ and Watson⁴. In particular, Watson provides a complete API supporting applications in exploiting dynamically knowledge from the Semantic Web [4]. This API has been used for example to discover automatically from online ontologies existing semantic relationships between terms [15], or to aggregate candidate senses for query terms in a word-sense disambiguation process [10].

However, a Semantic Web search engine such as Watson is not only a service supporting the development of Semantic Web applications. It also represents a unprecedented resource for researchers to study the Semantic Web, and more specifically, how formalized knowledge and data are produced, shared and consumed online. In this paper, we describe ongoing work on creating a platform making possible the realization of studies over a variety of aspects (including

ontology agreement, as well as evolution, consistency, connectedness, redundancy) at the scale of the Semantic Web. We realize this platform by integrating Watson with techniques for large-scale clustering, formalizations of ontology relations, as well as mining mechanisms, to make emerge recurrent structures and patterns in the way Semantic Web knowledge is organized.

More precisely, we focus here on monitoring and analyzing a variety of relationships that exist between ontologies on the Web. We look at different levels of similarity that can be computed between ontologies, to see how they can be clustered in general areas of high density, or coverage. These similarity measures are also used as part of process of tracking different versions of ontologies, providing a basis to study their evolution. An important aspect of ontologies is that they represent a consensus within a specific community, but might disagree, i.e., represent knowledge in conflict with other ontologies, built for other communities. By assessing these (dis)agreements, we can obtain an overview of the ontology landscape in a particular area, distinguishing particular trends and opinions. Finally, we also look at the more common relationship of dependency, showing how we can derive from the analysis of dependency graphs representing modular ontologies, common patterns in the way these ontologies are organized.

2. PREVIOUS WORK

Initial studies intending to “characterize knowledge on the Semantic Web” were realized a few years ago, relying on Watson [5], Swoogle [6], or manually collected ontologies [17], mostly to look at the way Semantic Web technologies such as RDF and OWL were employed in online ontologies.

For example, to give an account of the way semantic technologies are used to publish knowledge on the Web, of the characteristics of the published knowledge, and of the networked aspects of the Semantic Web, [5] presented an analysis of a sample of 25,500 semantic documents collected by Watson. This analysis looked in particular into the use of Semantic Web languages and of their primitives. One noticeable fact that can be derived from analyzing both the OWL (1) Species and the description logics used in ontologies is that, while a large majority of the ontologies in the set were in OWL Full (the most complex variant of OWL 1, which is undecidable), most of them were in reality very simple, only using a small subset of the primitives offered by the language (95 % of the ontologies were based on the $\mathcal{ALH}(D)$ description logic). This is consistent with conclu-

¹<http://swoogle.umbc.edu/>

²<http://sindice.com>

³<http://iws.seu.edu.cn/services/falcons/>

⁴<http://watson.kmi.open.ac.uk>

sions obtained in [17].

More recently, researchers in the Semantic Web community have started to consider specific aspects of this knowledge. For example, [9] tries to answer the question “what four million mappings can tell you about two hundred ontologies”. Also, an initial experiment [1] recently investigated the use of information present in the URIs of the ontologies to encode versioning information, which can be extracted to trace the different versions of ontologies. It appears that many different, more or less popular conventions are used to encode such version information, from the use of version numbers (e.g., `v1.2, rev=3.6`) to the use of time-stamps and dates (using two or three numbers, in big endian or little endian orders). This work is being extended in this paper, showing how the combination of recognizing these patterns in URIs and of machine learning algorithms can help detect ‘chains’ of ontology versions, providing an insight on how ontologies evolve on the Web.

3. STUDYING ONTOLOGIES THROUGH ONTOLOGY RELATIONS

Ontologies and semantic documents online are not isolated artifacts: they are, explicitly or implicitly, related with each other through a large variety of semantic relations [11]. Indeed, studies have for example targeted ontology comparison in order to identify overlaps between ontologies [13] or to find differences between versions of an ontologies [14, 12]. In [8] ontology integration is defined as the construction of an ontology C that formally specifies the union of the vocabularies of two other ontologies A and B . The most interesting case is when A and B commit to the conceptualization of the same domain of interest or of two overlapping domains. In particular, A and B may be related by being *alternative ontologies*, *truly overlapping ontologies*, *equivalent ontologies with vocabulary mismatches*, *overlapping ontologies with disjoint domain*, *homonymically overlapping ontologies*.

In this paper, we consider that characterizing online ontologies requires a general study of these relations between ontologies, providing a formal base for defining, manipulating and reasoning upon the links that relate ontologies online, explicitly or implicitly. For this reason, we defined in [2] the DOOR ontology: a Descriptive Ontology of Ontology Relations. This ontology includes relations covering various aspects of ontology relationships, such as inclusion, similarity, versioning and compatibility, at various levels (lexical, syntactic, semantic). It defines the relations through their formal properties, and through their taxonomical organization.

Apart from being the result of a formal study of ontology relationships, the main contribution of this ontology is to provide the core of a platform to study large collections of ontologies and semantic documents, such as the Semantic Web, supporting a formal approach to characterizing, exploring and reasoning upon these relations. In the next section, we present a number of mechanisms that are used to efficiently detect various relations between ontologies, to populate DOOR, and what we can learn about the Semantic Web from identifying these relations.

4. RESULTS OF ANALYZING ONTOLOGY RELATIONS

While ontology relations seem to be an important aspect of the Semantic Web landscape, there have not been many studies trying to characterize these relations, and to learn from them. Here, we describe a number of studies of different aspects (similarity, versioning, agreement, modularity) of ontology relationships that occur in a large collection of ontologies. We apply such analyses on the set of ontologies crawled by the Watson search engine, as it corresponds to a large, representative subset of the Semantic Web.

4.1 Ontology Similarity

Similarity is a difficult relation to assess and therefore to study, first because it does not have a formal definition, but more importantly because different similarity measures can be considered, depending on the task at hand. This is especially true for ontologies, for which similarity might happen at many different levels. Ontologies can be similar because they cover the same domain, including concepts describing common terms in this domain. They can also be similar in their structure, i.e., in the way they represent these common terms through various axioms. Finally, in a context when reasoning with ontologies is important, ontologies can be semantically similar, as they allow to derive similar inferences. For this reason, we define here a collection of similarity measures, at the basis of the similarity relations present in our ontology relation ontology and that take each into account one of these three aspects of ontologies.

Lexicographic Similarity is the simplest of our similarity measures. It basically states that two ontologies are similar if they share a significant portion of their vocabularies. The vocabulary $Voc(O)$ of an ontology O is defined as the set of normalized names of named entities (classes, properties and individuals) that appear in the ontology. On this basis, we straightforwardly define the lexicographic similarity measure $lSim$ between two ontologies O_1 and O_2 as

$$lSim(O_1, O_2) = \frac{|Voc(O_1) \cap Voc(O_2)|}{\max(|Voc(O_1)|, |Voc(O_2)|)}$$

representing the idea that ontologies are lexicographically similar if they share most of their vocabularies. When computing this relation, we use a simple method for normalizing the names of entities in $Voc(O)$, replacing separators by a common character and putting everything to lower case. While this measure is relatively simple (the most complex part being computing the intersection, with can be done in $O(n)$, with n being the size of the biggest vocabulary), the major difficulty is to efficiently extract the vocabularies of ontologies. Here we rely on the indexes built by the Watson search engines, as well as on various optimizations to ensure that, when computing similarity measures in a large collection of ontologies, this operation is not repeated unnecessarily.

Syntactic Similarity goes a step further by considering the structure of the ontologies. Here, we consider an ontology O as a set of axioms (or statements) ultimately representing relations between the various entities present in the ontology. We also normalize the representation of these axioms in the same way as above, by considering only the normalized names of the entities they mention. With this taken into account, we define the syntactic similarity

function *synSim* between two ontologies as

$$\text{synSim}(O_1, O_2) = \frac{|O_1 \cap O_2|}{\max(|O_1|, |O_2|)}$$

here again straightforwardly representing the idea that two ontologies are similar syntactically if they share a large number of their axioms. In the same way as for *lSim*, the major difficulty in computing *synSim* for a large collection of ontologies is in the operation of extracting a normalized set of axioms. As a pre-processing step, we therefore build an index of the normalized, sorted lists of axioms for each ontology, used to reduce the time needed for the actual computation.

Semantic Similarity is intended to take into account the logical foundation of ontologies, in particular as a way to derive implicit knowledge. Indeed, while two ontologies might share a large proportion of their axioms, and so be syntactically similar, the differences between them might be amplified when looking at the statements they entail. Also, there might be many different ways to represent knowledge so that the same conclusions can be derived, even if the syntactic definitions of the ontologies are different. A method analogous to the two previous measures could be envisaged to assess how much the sets of inferred statements from the two ontologies overlap. However, in a language such as OWL, these sets of inferred statements are often infinite. To work around this problem, we define $LC(O_1, O_2)$ as the set of axioms from an ontology O_1 which can be inferred from an ontology O_2 . On this basis, we define *semSim*, the semantic similarity measure between two ontologies O_1 and O_2 as

$$\text{semSim}(O_1, O_2) = \frac{|LC(O_1, O_2) \cap LC(O_2, O_1)|}{\max(|O_1|, |O_2|)}$$

providing an approximation of the initial idea of comparing the sets of logical consequences (inferences) of the two ontologies. Semantic similarity is a very complex measure to assess, as it heavily relies on the use of ontological reasoning. Pre-processing cannot be applied on ontologies individually, since the inferences to be compared depend on both the ontologies, leading to about 2^n combinations of parameters for the LC function to consider, with n the number of ontologies. There is currently no reasoner able to realize such an operation in a time that would allow the use of this function in a large collection of ontologies.

We applied the *lSim* and *synSim* measures to discover similarity relations between the ontologies of the Watson collection. The results of these computations are used in particular to detect the presence of a versioning relation between ontologies, as presented in the next section. However, such measures are also interesting by themselves, as they provide us with the ability to discover particular clusters of ontologies, ‘areas’ of high density where many similar ontologies exist. To demonstrate this, we applied purpose-built, scalable clustering mechanisms using the results of *lSim* on a subset of the Watson collection containing 160,000 semantic documents, grouping together ontologies with a lexicographic similarity value of more than 0.3. We obtained more than 70,000 different clusters, with only 6,100 of them containing more than one element. These clusters contain on average 16 different ontologies, with the biggest one con-

taining almost 50,000 different semantic documents, showing that such a similarity function can be effectively used to discover large, dense groups of ontologies covering, at least in appearance, the same topics. In particular, a number of the large clusters that we identified through this method correspond to large collections of documents, such as FOAF descriptions from livejournal.com or talkdigger.com. A visualization of these clusters would help focusing on particular areas of interest, while ignoring such large sets if judged irrelevant.

4.2 Evolution of Ontologies on the Web

Another aspect of ontologies on the Web is that they are dynamic objects, changing and evolving with time. Looking at these evolutions can help us better understand how knowledge is modeled over time in a particular community. Actually, mechanisms exist, for example as part of the OWL language, to keep track of ontology versions, declaring that an ontology is related to another through a temporal, versioning relation. However, these mechanisms are rarely used by ontology engineers, and such relations can be lost in the process of collecting ontologies to create a large repository like the one of Watson.

An observation initially made in [1] is that, while not formally represented in the content of an ontology, versioning information is often encoded in its URI. For example, the Watson collection contains the following three ontologies:

1. <http://dev.w3.org/cvsweb/2000/10/swap/log.rdf?rev=1.2>
2. <http://dev.w3.org/cvsweb/2000/10/swap/log.rdf?rev=1.4>
3. <http://dev.w3.org/cvsweb/2000/10/swap/log.rdf?rev=1.5>

In this example, it is easy to recognize a particular pattern showing that these three ontologies in fact represent a sequence of versions of the same ontology. Empirically analyzing a sample of nearly 1000 ontologies, we discovered that this phenomenon was common and identified 6 different patterns in the way the numbers contained in ontology URIs differ, which can potentially represent a directed versioning relation (including version numbers like above, but also dates using 2 or 3 numbers, in big endian (YYY/MM/DD) or little endian (DD/MM/YYYY) notations).

By applying these patterns on a set of about 250,000 ontologies and semantic documents from the Watson collection, we identified around 27,000 candidate versioning relations (i.e., pairs of versions of ontologies) representing about 1,400 ‘evolving ontologies’. However, as discussed in [1], the precision of this approach in identifying versions is relatively low, as the employed patterns are also often used to represent other types of temporal relations and time-stamps are commonly used to name large quantities of automatically generated semantic descriptions, regardless of the fact that they are not time-dependent.

We therefore extended this study adding other indicators to provide clues on whether two ontologies are versions of each other. In particular, one of our hypothesis is that the similarity between two ontologies (as calculated above) could help. Indeed, a naive idea is that, a new version of an ontology being a modification of the previous one,

they should overlap to a large extent. However, naive approaches of exploiting this idea (using a threshold to filter out insufficiently similar ontologies) did not lead to significant improvements in our dataset. For this reason, in order to derive useful models characterizing ontology versioning relations, and therefore the way to detect them, we experimented with the use of machine learning classifiers.

In our case, the use of a classifier is needed to detect whether a pair of ontologies (O_1, O_2) represents a versioning relation or not, depending on a number of ‘features’ of this pair of ontologies. The considered features here include:

The pattern used to detect this pair as a candidate versioning relation. We use as input pairs of ontologies that are derived from the initial method using information encoded in the URIs of the ontologies. Six different patterns are used in this method and it is expected that some of them would perform better than others.

The similarity between O_1 and O_2 . Which we calculate using the measures of lexicographic and syntactic similarity described in the previous section.

The length of the chain of (candidate) versions. Each candidate ontology pair is part of a chain of such pairs, which, if valid, represents the versioning history of a particular ontology, e.g., $[(O_1, O_2), (O_2, O_3), \dots, (O_{n-1}, O_n)]$. Empirically, we found that there seems to be a correlation between the length of such a chain and the validity of the versioning relation, with long chains generally representing other kinds of relations.

The idea here is to manually classify a number of these pairs of ontologies as being either examples of versioning relations, or counter-examples. Using supervised learning algorithms, we can then build a model of how the above features seem to impact on the results, not only obtaining a more accurate approach to detect versions of ontologies, but also models of the way such versioning relationships are characterized in terms of these features. We consider three different types of classifiers, included in the Weka toolkit [18]:

A Naive Bayes Classifier relies on Bayes’ theorem that a conditional probability can be derived from its inverse, knowing the probabilities of the involved events.

Support Vector Machine models consider each entry as a p-dimensional vector and try to separate such points through a p-1-dimensional hyperplane that not only separates the points in one class or the other, but for which the separation, or *margin*, is as large as possible.

A Decision Tree tries to derive an efficient procedure to decide whether an object should be classified in one class or the other, through a succession of attribute-based tests with a branch for each possible outcome.

We divided our set of 1,200 manually classified pairs of ontologies into three subsets of equal sizes. The *training set* is used as input of the learning algorithms to build a model. The *test set* is used to check the performance of the learning algorithms, by validating their outputs against manually classified results. The *validation set* is used in the same way as the test set, to verify the stability of the models

Classifier/Precision	Test set	Validation set
Naive Bayse	70%	74%
Support Vector Machine	84%	89%
Decision Tree	80%	87%

Table 1: Performance of the three classifiers in detecting versioning relations.

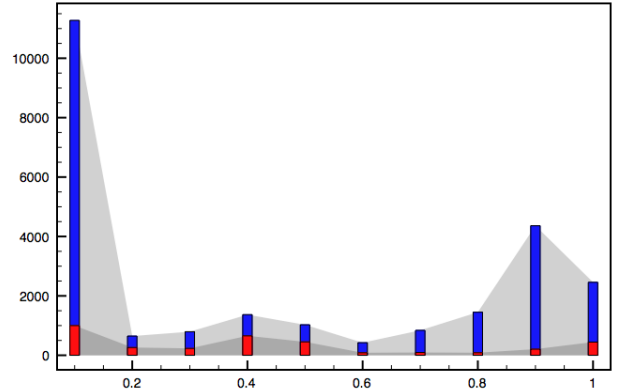


Figure 1: Distribution of the lexicographic similarity in the dataset of ontology pairs (blue), and for the pairs classified as versioning relations by Support Vector Machine (red).

in terms of performance. The results of this experiment are summarized in table 1.

As can be seen from the table, this approach of using machine learning classifiers to detect versioning relations yield good performance, with up to 89% of the pairs of ontologies detected as representing versioning relations being correctly classified. The model generated from the use of Support Vector Machine clearly shows higher performance than the other two, while all the classifiers demonstrate reasonable stability in performance, from the test set to the validation set (the test set being, however, marginally harder to classify for all of the classifiers). Applying this model to the entire dataset from the Watson collection of ontologies, we obtained more than 8,000 ontology pairs being detected as versioning relations, 6,500 of which we can expect to be correct.

Moreover, in addition to providing a reasonably accurate approach to detecting ontology versions, the use of such classifiers allows us to better characterize versioning relationships, based on the features used. Indeed, it appears clearly for example that our initial hypothesis that the longer a chain of ontologies is, the less likely it is to be made of version relations has been validated by our model. Indeed, chains of ontology versions are rarely longer than 10 steps, while our dataset contained candidate chains of 800 ontologies. Also, the graph in Figure 1 shows that, while indeed a certain level of (lexicographic) similarity should be expected in ontology versions, ontologies that are very similar are less likely to be versions of each other. In other terms, both a sufficient overlap and a sufficient amount of changes should be expected in ontology versions.

4.3 Agreement and Disagreement in Ontologies

Ontologies are knowledge artifacts representing particular models of some particular domains. They are built within the communities that rely on them, meaning that they represent consensual representations inside these communities. However, when considering, the set of ontologies distributed on the Web, many different ontologies can cover the same domain, while being built by and for different communities. Knowing which ontologies agree or disagree with others can therefore be very useful in many scenarios.

For this reason, [3] defines two basic measures for assessing agreement and disagreement of an ontology O with a statement $s = \langle \text{subject}, \text{relation}, \text{object} \rangle$:

$$\begin{aligned} \text{agreement}(O, s) &\rightarrow [0..1] \\ \text{disagreement}(O, s) &\rightarrow [0..1] \end{aligned}$$

Two distinct measures are used for agreement and disagreement so that an ontology can, at the same time and to certain extents, agree and disagree with a statement. These two measures have to be interpreted together to indicate the particular belief expressed by the ontology O regarding the statement s . For example, if $\text{agreement}(O, s) = 1$ and $\text{disagreement}(O, s) = 0$, it means that O fully agrees with s and conversely if $\text{agreement}(O, s) = 0$ and $\text{disagreement}(O, s) = 1$, it fully disagrees with s . Now, agreement and disagreement can vary between 0 and 1, meaning that O can only partially agree or disagree with s and sometimes both, when $\text{agreement}(O, s) > 0$ and $\text{disagreement}(O, s) > 0$. Finally, another case is when $\text{agreement}(O, s) = 0$ and $\text{disagreement}(O, s) = 0$. This basically means that O neither agrees nor disagrees with s , for the reason that it does not express any belief regarding the relation encoded by s .

Considering that ontologies are made of statements, extending the measures above to compute a relation of agreement and disagreement between two ontologies is relatively straightforward, using the mean of each measure for each statement of an ontology against the other ontology, in both directions and making this a normalized measure. However, while relatively simples, the two measures of agreement and disagreement between ontologies provide an interesting way to obtain an overview of a set of ontologies. Indeed, this experiment looked at the 21 ontologies returned by Watson when querying for semantic documents containing a class with the term *SeaFood* in its ID or label, and computed the agreement and disagreement relations for all pairs of ontologies in this set. The results are shown in Figure 2 where ontologies are numbered according to their rank in Watson (valid on the 20/09/2009).

Analyzing these diagrams, it appears that there is a certain level of ‘coherence’ in the results. In particular, homogeneous clusters can be built from the agreement and disagreement values. Indeed, the ontologies O1, O2, O3, O4, O5, O6, O7, O11, O12, O13, O16, O17, O18, O19 and O20 all fully agree with each other and, at the same time, partially agree and disagree with O14 and O15. O14 and O15 also form a cluster since they agree with each other, and consistently disagree with the same set of ontologies (the reason being that O14 and O15 are the ontologies considering that SeaFood is a subclass of Meat, but agree on all the other related statements). O21 is also particular, since it disagrees with most of the ontologies of the first cluster, sometimes

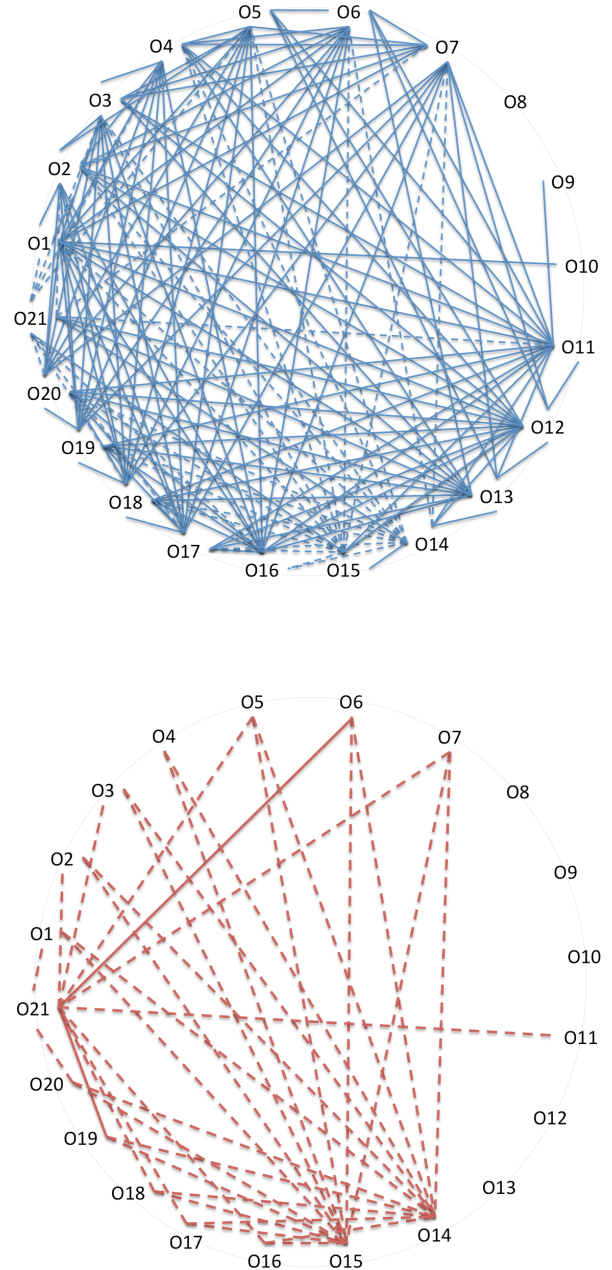


Figure 2: Agreement (top) and disagreement (bottom) relations among the 21 test ontologies. Plain lines represent full dis/agreement (measures' values = 1). Dashed lines represent partial dis/agreement (measures' values greater than 0).

fully. Indeed, it also considers SeaFood to be a subclass of Meat, and additionally disagrees on several other statements with some of the other ontologies (for example, it considers that tuna is a subclass of fish while several other ontologies consider tuna as an instance of fish). O8, O9 and O10 are particular since there is only a very small overlap between them and the other ontologies. For example, O9 only agrees with O11 that Vegan is a subclass of Vegetarian. In other terms, the results of this experiment show that such relations of agreement and disagreement can provide a way to identify ‘camps’: groups of ontologies that relate with each other because of having the same understanding of common terms. It provides a way to cluster ontologies which is complementary to the coverage view provided by similarity (see section 4.1), looking more closely at how various communities understand the semantics of common terms differently.

4.4 Ontology Modularity

While the relations we considered above generally necessitate to be made explicit, we consider here a simpler type of links which appears to be common in Semantic Web ontologies: the relation of dependency, expressed through the `owl:imports` primitive in the OWL language. While this relation is in appearance very simple, it relates to a complex and subjective area of ontology design, as it is the main mechanism used nowadays to introduce *modularity* in ontologies [16].

Here again, the availability of the Watson collection allows us to construct clusters, or graphs, of ontologies related with each other through import. We build a recursive process that starts from a set of ontologies containing import statements, download all the imported ontologies, check if the imported ontologies relate to other, already known ontologies, and repeat the same process with the newly found ontologies until the entire space has been explored. The idea is that, through this process, small dependency graphs can be built that each potentially corresponds to a modular ontology and in which each of the nodes represents an ontology module. In order to reduce the noise corresponding to ontologies being heterogeneously imported from many others, we filter out the graphs containing nodes with URIs in different domains.

We obtain as a result a set of 76 online, *modular ontologies*, each of them containing on average 5 modules (sub-ontologies), with the biggest containing 36 modules.

Looking closely at the graphs from this set, a first observation that can be drawn is that modular ontologies present on the Web generally rely on simple structures. Most of the graphs contain only 1 or 2 levels, and none of them more than 6. Another observation is that the size of modules, which is a factor favored by many automatic ontology modularization techniques (see [16]), does not seem to be applied consistently as a criterion to create ‘real-life’ modular ontologies. Indeed, while most of the ontologies contain relatively small modules on average, there is often a large variation in the sizes of modules within one modular ontology, with the size of the biggest module commonly being 10 or even 100 times larger than the one of the smallest module.

A visual representation of the graphs representing modular ontologies also makes appear interesting common patterns, which can be seen as empirical equivalents to the carefully defined architectural patterns considered in the area of ontology design [7]. An example of such a common situa-

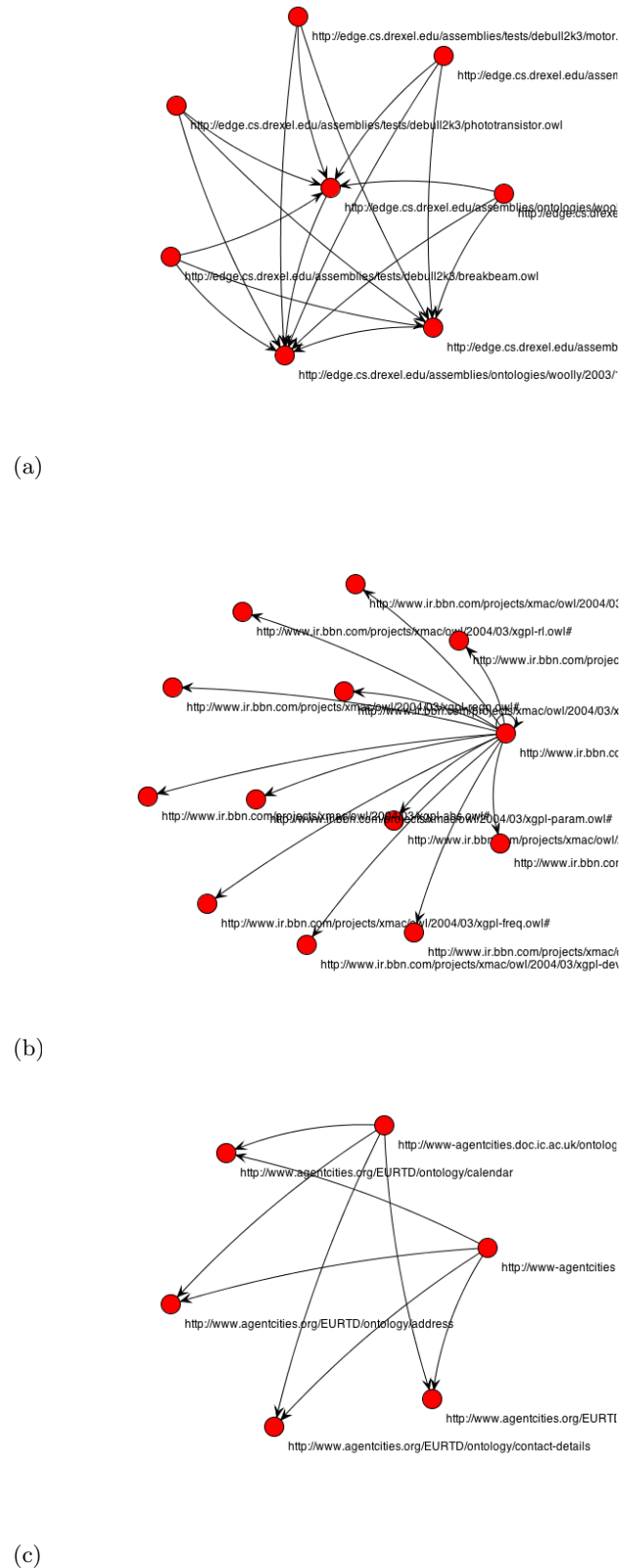


Figure 3: Example of the application of patterns in modular ontologies. (a) Use of upper-level ontologies. (b) The Shell Pattern. (c) Combined use of 2 shell patterns.

tion is when most of the modules in a modular ontology are dependent from a few (typically 2 or 3) ‘top’ ontologies (see Figure 3(a) for an example with three top modules). Such a pattern corresponds to the situation where *upper-level ontologies* were used to derive a number of more specific domain or application ontologies. An even more common pattern corresponds to the situation where a number of mostly independent modules are being used in one integrated ontology, representing a focus point in the dependency graph, where the features of all the different modules are being imported. Because of the shape of the corresponding graph, we call such a pattern a *shell-pattern* (see Figure 3(b) for an example of a ‘pure’ shell-pattern).

This last analysis shows that even a relatively simple relation such as `owl:import` can help obtaining a better understanding of how knowledge is published on the Web. In particular, identifying patterns in dependency graphs can be useful to ‘reverse engineer’ modular ontologies, detecting interesting combinations of patterns. As an example, Figure 3(c) shows a case of combining the use of two shell patterns, effectively creating two ontologies from the different compositions of the same set of elementary modules. This could also help validating and diagnosing the design of modular ontologies by detecting common bugs in the organization of modules (i.e., anti-patterns), such as co-inclusions and cycles.

5. CONCLUSIONS

In this paper, we have presented the foundation of a platform to study and characterize ontologies made available on the Semantic Web. This platform relies on the collection of ontologies gathered by the Watson Semantic Web search engine, as well as the access functionalities provided by the Watson engine as a core component. Several works exist on characterizing ontologies as individual objects. Here, we consider as a basis to the study of online ontologies the formalization, monitoring and analysis of the relationships that exist between them. We investigated in more detail four different types of relations (similarity, versioning, (dis)agreement and modularity), looking at what we can learn from detecting them in a large collection of ontologies such as the one of the Watson repository. The next step in building this platform is to consider the integration of these different relations, relying on our ontology of ontology relations, in order to apply reasoning upon these relationships, discover new ones and help users explore this collection of ontologies more efficiently.

6. REFERENCES

- [1] C. Allocca, M. d’Aquin, and E. Motta. Detecting different versions of ontologies in large ontology repositories. In *International Workshop on Ontology Dynamic, IWOD*, 2009.
- [2] C. Allocca, M. d’Aquin, and E. Motta. Door: Towards a formalization of ontology relations. In *Proc. of International Conference on Knowledge Engineering and Ontology Development (KEOD)*, 2009.
- [3] M. d’Aquin. Formally measuring agreement and disagreement in ontologies. In *Proceedings of the Fifth International Conference on Knowledge Capture, K-CAP*, 2009.
- [4] Mathieu d’Aquin. Building semantic web based applications with watson. In *In proceedings of the 17th Annual WWW conference - WWW2008*, 2008-04.
- [5] Mathieu d’Aquin, Claudio Baldassarre, Laurian Gridinoc, Sofia Angeletou, Marta Sabou, and Enrico Motta. Characterizing knowledge on the semantic web with watson. In *EON*, volume 329 of *CEUR Workshop Proceedings*, pages 1–10. CEUR-WS.org, 2007.
- [6] L. Ding and Tim Finin. Characterizing the semantic web on the web. In *Proceedings of the 5th International Semantic Web Conference*, 2006.
- [7] A Gangemi, A Gomez-Perez, V Presutti, and M.C Suarez-Figueroa. Towards a catalog of owl-based ontology design patterns. In *Conference of the Spanish Association for Artificial Intelligence, CAEPIA*, 2007.
- [8] A. Gangemi, D. M. Pisanelli, and G. Steve. An overview of the onions project: Applying ontologies to the integration of medical terminologies. *Technical report. ITBM-CNR, V. Marx 15, 00137, Roma, Italy*, 1999.
- [9] Amir Ghazvinian, Natalya F. Noy, Clement Jonquet, Nigam Shah, and Mark A. Musen. What four million mappings can tell you about two hundred ontologies. In *The Semantic Web - ISWC 2009*, volume 5823 of *Lecture Notes in Computer Science*, pages 229–242. Springer, 2009.
- [10] Jorge Gracia and Eduardo Mena. Overview of a semantic disambiguation method for unstructured web contexts. In *K-CAP*, pages 187–188. ACM, 2009.
- [11] A. Kleshchev and I. Artemjeva. An analysis of some relations among domain ontologies. *Int. Journal on Inf. Theories and Appl*, 12:85–93, 2005.
- [12] B. Konev, C.Lutz, D.Walther, and F.Wolter. Cex and mex: Logical diff and logic-based module extraction in a fragment of owl. *Liverpool University, UK and TU Dresden, Germany*, 2008.
- [13] A. Maedche and S. Staab. Comparing ontologies-similarity measures and a comparison study. *Proc. of EKAW-2002*, 2002.
- [14] N. F. Noy and M. A. Musen. Promptdiff: A fixed-point algorithm for comparing ontology versions. *18th National Conf. on Artificial Intelligence (AAAI)*, 2002.
- [15] Marta Sabou, Mathieu d’Aquin, and Enrico Motta. Exploring the semantic web as background knowledge for ontology matching. *Journal on Data Semantics*, 11:156–190, 2008.
- [16] Heiner Stuckenschmidt, Christine Parent, and Stefano Spaccapietra, editors. *Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization*. Springer, 2009.
- [17] Taowei David Wang, Bijang Parsia, and James Hendler. A survey of the web ontology landscape. In *Proc. of the 5th Int. Semantic Web Conference (ISWC 2006)*, Athens, Georgia, 2006.
- [18] Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques (Second Edition)*. Morgan Kaufmann, 2005.