Facilitating Ontological Benchmark Construction Using Similarity Computation and Formal Concept Analysis

Ondřej Zamazal and Vojtěch Svátek
Department of Information and Knowledge Engineering
University of Economics, W. Churchill Sq.4, 130 67 Prague 3, Czech Republic
{ondrej.zamazal|svatek}@vse.cz

ABSTRACT

The demo paper deals with preparation of ontological benchmarks as ontology sets selected from ontology repositories. We distinguish between two types of ontology sets according to coverage of ontology metrics values: a homogeneous set contains ontologies with very similar values of selected ontology metrics, while a heterogeneous set contains ontologies with different values of such metrics. Homogeneous sets can be built straightforwardly upon specification of the required metrics values by the user when interacting with repositories. However, building heterogeneous ontology sets is not covered in the current functionality of ontology repositories. We propose to employ search techniques leveraging on ontology similarity in this process. In similarity-based search one can obtain either homogeneous or heterogeneous sets, by varying the selection parameters. Further, we investigate whether Formal Concept Analysis could help generate sufficiently large homogeneous sets.

1. INTRODUCTION AND MOTIVATION

There are more and more ontologies on the semantic web along with new tools for their management and exploitation. New tools obviously need testing their functionality on ontology sets allowing to balance between 1) presence of specific features crucial for a particular functionality, and, 2) sufficient coverage of different cases the tool might encounter. With respect to the former, for instance, ontology repair tools [7, 4] can only be assessed on models with nontrivial concept expressions. In this case ontologies, in the respective ontological benchmark, are rather *homogeneous.* With respect to the latter, for instance, thoroughly testing ontology visualization techniques [5] demands diverse ontology aspects such as taxonomies of different depth, instances or various types of axioms. In this case ontologies, in the respective ontological benchmark, are rather *heterogeneous.* Homogeneous sets can be built straightforwardly upon specification of the required metrics values by the user when interacting with repositories. However, building heterogeneous ontology sets is not covered in the current functionality of ontology repositories.

We extend our previous work on ontological benchmark construction [14], relying on the Online Ontology Set Picker (OOSP)[1] tool, by adding support of similarity-based ontology search and by studying the potential of Formal Concept Analysis (FCA) for identifying interesting ontology sets. Section 2 presents the basic functionality of OOSP. Section 3 introduces ontology search based on similarity computation as an extension to OOSP. Section 4 presents the initial investigation of ontological benchmark construction based on FCA. Related work is presented in Section 5, and Section 6 wraps up the paper with conclusions and future work.

2. OVERVIEW OF OOSP

OOSP allows to select, from ontology repositories, a set of ontologies satisfying a user-defined sets of metrics [14]; supporting creation of homogeneous sets. It allows ontological tool designers to rapidly build custom benchmarks for testing different features of their tools. The web front-end allows to select ontologies by specifying a broad range of metrics and delivers benchmarks along with their statistics of metrics, including a graph view. Currently it includes the following snapshots from various repositories: the BioPortal[2] Feb. 2015 snapshot contains 317 ontologies, the *Linked Open Vocabulary* (LOV)[3] Feb. 2015 snaphot contains 461 ontologies, the LOV Jan. 2016 snapshot contains 509 ontologies, the *NanJing Vocabulary Repository*[4] (NJVR) Jan. 2016 snapshot contains 1403 ontologies, the NJVR merged Jan. 2016 snapshot contains 225 ontologies, and the OntoFarm[5] Jan. 2016 snapshot contains 16 ontologies.

3. SIMILARITY COMPUTATION IN OOSP

Ontology similarity can be employed for exploring an ontology collection via cluster analysis. We inspected the LOV Feb. 2016 snapshot using hierarchical clustering with Ward's minimum variance method using *the R language.*[6] We separately analyzed ontology clusters according to entity-based, axiom-based and class-expression-based ontology metrics, and usually could single out several apparent clusters. There were also some outliers, typically corresponding to very large ontologies. Although the clusters could be interesting per se, it appears more practical to let the user explicitly select the

SEMANTICS 2016: Posters and Demos Track
September 13-14, 2016, Leipzig, Germany

[1] http://owl.vse.cz:8080/OOSP/; the webpage also links to commented screencasts describing both the general functionality and the new features described in this paper.
[2] http://bioportal.bioontology.org/
[3] http://lov.okfn.org/dataset/lov/
[4] http://ws.nju.edu.cn/njvr/
[5] http://owl.vse.cz:8080/ontofarm/
[6] https://www.r-project.org/

ontology metrics upon which the similarity computation is to be carried out, as well as the similarity threshold. We support this in the following scenario.

Similarity-Based Ontology Search Scenario.

While the construction of homogeneous ontology sets is supported by the OOSP core, in order to construct heterogeneous sets we need to apply similarity-based search as a new functionality.[7] There is a three-step workflow.

First, the user provides either the *URI of an online ontology* or a *storage code* of an ontology stored in OOSP.

Second, s/he obtains an *ontology metrics overview* for the given ontology; while for an online ontology they are computed on the fly, for ontologies stored in OOSP they are merely retrieved. S/he may select an *ontology pool* restricting the scope of ontologies to be considered. Next, s/he selects the ontology metrics to be considered for similarity computation out of six ontology metrics groups: entity metrics (4) including the number of entities; axiom metrics (25) including the number of different axiom types (e.g., subsumption or equivalence); class expression type metrics (11) including expression types used for construction of anonymous classes (e.g., existential quantification); taxonomy metrics (10) describing the taxonomy (e.g., the number of top classes, leaf classes, branching degree or maximum taxonomy depth); annotation metrics (6) including counts of selected annotation types (e.g., labels or comments) and of different languages involved in label annotations; finally, detail metrics (13) including some newly designed metrics related to domain/range (e.g., the number of anonymous classes as domain definition). In all the user can select any combination of ontology metrics out of the 69 metrics available in OOSP.As further parameters s/he can specify the maximum number of ontologies on output, the minimum and/or maximum of the similarity threshold, and the preference of the upper or lower end of this interval.

Third, the user obtains the *ontology set* meeting the provided restrictions, including a downloadable table with all metrics values for all selected ontologies.

Implementation.

Ontology metrics are computed using our own implementation in Java or the OWL-API library.[8] For similarity computation we use the R language. Based on ontology metrics computed for the given ontology and the selected ontology pool, an $n \times m$ data matrix is generated, where n corresponds to the number of ontologies in the selected ontology pool plus one ontology given by the user and m to the number of selected ontology metrics. On input data we first apply the *scale* function in R (centering and scaling the attributes of a numeric matrix). Next, we compute the *euclidean distance* between the given ontology and all ontologies from selected ontology pool in a pair-wise manner, where the two vectors v_1, v_2 represent ontologies O_1 and O_2 with values of the computed ontology metrics:

$$d(v_1, v_2) = \sqrt{\sum_{j=1}^{m} (v_{1j} - v_{2j})^2} \quad | \quad d(v_1, v_2) \in [0, \infty]. \qquad (1)$$

We then transform the computed distance function to similarity using the following formula:

$$s(v_1, v_2) = \frac{1}{1 + d(v_1, v_2)} \quad | \quad s(v_1, v_2) \in [0, 1]. \qquad (2)$$

Finally, the ontologies are selected according to the similarity search parameters, i.e. the maximum number of similar ontologies, the similarity interval and its preferred end.

4. FCA IN ONTOLOGICAL BENCHMARK CONSTRUCTION

Formal Concept Analysis.

Formal Concept Analysis (FCA), introduced by Ganter and Wille [3], enables to extract groups of similar objects from a set of objects O, where the objects are described by a set of attributes T. The starting point for the FCA method is a data structure called *formal context*, consisting of the sets T, O and I; I is a binary relation such that $I \subseteq O \times T$. (A, B) is a *formal concept*, or *concept* in short, of the formal context (O, T, I) iff $A \subseteq O$ and $B \subseteq T$. A is the *extent* and B is the *intent* of the formal concept (A, B). A formal context can be transformed into a mathematical structure called *concept lattice*. A complete concept lattice contains all formal concepts of (O, T, I) with the order $(A_1, B_1) \leq (A_2, B_2) \Leftrightarrow A_1 \subseteq A_2$.[9]

For constructing concept lattices we use *Lattice Constructor*, (LeCo) which is one of the modules of the *Coron* platform.[10] LeCo is theoretically grounded in the work of Szathmary et al. [11]. On input it takes the frequent closed itemsets (FCIs); O corresponds to an *itemset*. The *support* of an itemset X, denoted as $supp(X)$, is the number of items in this itemset, $supp(x) = |X|$. Further, an itemset is *frequent* if its support is not lower than a given *minimum support*. Next, an itemset is *closed* if it has no *proper superset* with the same support with regard to the order defined above. FCIs are reduced representations for all possible frequent itemsets and they are also used for non-redundant bases of valid association rules [6]. Since we are not interested in the complete concept lattice,[11] we only generate an *iceberg lattice*, also called the *intent lattice of a context* [3].

Using FCA for Ontology Collection Exploration.

In our case objects (items) in an itemset are ontologies and attributes correspond to ontology metrics. In our exploration we employed six groups of ontology metrics available in OOSP [14], see Section 3. Our motivation of using FCA for exploration of ontology collections is to consider its utility for ontological benchmark construction. We suppose that it would enable the user to discover an ontology set rich in many different characteristics and large enough at the same time. In practice, the user could come up with a certain ontology metric (and possibly its value), and the FCA-based service could provide different options of ontology sets differing in size and characteristics. In terms of FCA it means that we focus on concepts with "large" intent as well as "large" extent.

[7] Under 'Go to Ontology Search Based on Similarity'.

[8] http://owlapi.sourceforge.net/

[9] For a more precise description we refer to [3].

[10] http://coron.loria.fr/

[11] Generating a complete concept lattice for large data is also computationally demanding.

In our exploratory workflow we employ a four-step preprocessing phase, for which we use the R language:

1. Exporting ontologies satisfying the selected numeric and categorical metrics,[12] from the chosen pool.
2. Discretizing the numerical metrics into at most n intervals of equal frequency.
3. Generating binary attributes from categorical and discretized numerical values, ignoring those having zero or one unique value for the original metric.
4. Creating a binary table for ontologies × attributes.

Based on this binary table LeCo generates the iceberg concept lattice according to the given minimal support.

Preliminary Insights.

For the three major collections, LOV (2016 snapshot), BioPortal and Nanjing, we applied different settings for the pre-processing step and for the iceberg concept lattice[13] generation. We briefly discuss selected results.

LOV. The setting with at most 10 attributes per discretized metrics in combination with minimal support by 50 ontologies yielded 319 attributes in the binary table and generated a rather flat (3-level) lattice.[14] The level number reflects both the position of the concept in the lattice and the number of attributes occurring in the concept intent.

To obtain a richer lattice we decreased the max number of attributes per metric to 5, leading to 167 attributes and a five-level lattice. An example from level three is {*labels [7,20), range class [2,7), object property range [2,7)*} *(51)*, i.e. a concept with 51 ontologies having a rather small value for the three respective metrics: number of labels, named classes in range and defined range for object properties. The lattice is however typically dominated by attributes corresponding to *higher* occurrences of respective metrics, such as an example from level four: {*labels [104, 16878], range class[26, 2329], axiom [802, 44101], object property range [27,2329]*} *(55)*. We see that there are the same metrics but with highest occurrence values, plus a reference to the high number of axioms. While the high number of labels might be surprising, the correlation of the other three metrics seems obvious. The two examples indicate that typical concepts from the lattice refer to mutually dependent metrics. It also shows there are more often concepts with a higher occurrences of respective metrics. This can be explained by the fact that attributes with high occurrence go together, but also, to a certain extent, by the equal-frequency discretization of numerical metrics. Since most ontology metrics for LOV exhibit a lognormal distribution, discretization using equal width would lead to domination of the lattice by lower-interval values of metrics; in contrast, in the case of equal frequency the long tail includes a large range of values for a given metric, which increases the chance that more ontologies occur together in the long tails of multiple metrics.

In order to explore even more complex lattices, we decreased the min support to 30. This generated a lattice with ten levels. An example from level ten, having 30 ontologies in its extent, is: {*classes[46,2864], object properties [36,2529], leaf classes [36,2123], annotation assertion [411, 34491], labels [104, 16878], range class [26,2329], axiom count [802,44101], logical axioms [210,22834], object property domain [23,1892], object property range [27,2329]*} *(30)*. This concept, again, refers to *high* values of metrics: compared to the previous ones, it also refers to the number of classes, annotations, etc.

BioPortal. We used a setup close to that above: max 5 binary attributes per metric, and min support 30. The generated lattice had eight levels, and the concepts were dominated by attributes corresponding to high metric values, which is more-or-less typical for the BioPortal ontologies, e.g., {*classes [3425,505039], leaf classes [2483,504358], subClassOf [3991,638199], labels [2220,505053], axiom [30090, 8202428]*} *(36)*, i.e. besides the structural attributes (classes, leaf classes, subClassOf and axioms) there is again a high number of labels in these 36 BioPortal ontologies.

NanJing. We used a setup with max 5 binary attributes per metric and min support 130, because of the higher number of ontologies (1403). The lattice had eight levels, and the concepts had similar intents as in the case of LOV. This can be explained by the fact that NanJing contains a similar kind of ontologies as LOV and they even shares 212 of them (computed using a method based on similarity of ontology signatures suggested by Matentzoglu et al. [8]). A concept from level eight of the lattice is: {*classes [35,1717], object properties [18,305], leaf classes [27,1421], domain class [16,275], range class [12,266], classes more than once in domain or range [6,71], object property domain [12,268], object property range [12,266]*} *(130)*, i.e. there are 130 ontologies from NanJing having those eight attributes. Besides those on classes and object properties, there is one referring to the "classes more than once in domain or range" metric, which measures the average usage of the same class across different domain or range axioms. Further, there is "domain class", referring to named classes appearing in domain axioms.

Based on our preliminary experimentation we can conclude that a reasonable setup for discovering interesting concepts via FCA might be a minimal support around 10% and an application of equal frequency discretization method to a maximum number of five intervals. This assumption is to be verified by a further study.

5. RELATED WORK

Several advanced ontology similarity measures have been discussed in the literature. For example Wu et al. [13] suggest a new learning algorithm for ontology similarity measuring in a high-dimensional space based on regularization and first-order representation. In our approach the Euclidean distance seems sufficient with respect to our dimensional space size (up to a few hundred of binary attributes). Ontology similarity is also applied in various areas, e.g., in ontology matching, as described by Euzenat and Valtchev [2].

Although there are many different repositories providing access to their ontologies, to the best of our knowledge there is no other approach that would directly apply ontology similarity computation as means of ontology access and ontological benchmark construction. The most related work has been presented by Allocca et al. [1], who introduced an approach for finding equivalent ontologies in the Watson engine. For each ontology a canonical form was

[12] Using the MySQL database underlying OOSP.

[13] Only 'lattice' notion from now on, for brevity.

[14] The lattices, in textual form, are at http://owl.vse.cz:8080/SEMANTiCS-2016/.

computed and indexed, and these canonical forms could be compared. Equivalence detection is however a different task than similarity computation. Generally the Watson search engine[15] allows to search ontologies using keywords. It is possible to specify keyword-based entity search scope, e.g., in labels of properties. Via its Java API, Watson also provides a SPARQL endpoint along with some precomputed metrics: concept coverage, DL expressivity, representation language (e.g., RDFS), numbers of classes, properties, individuals and statements. BioPortal provides a term-based search for classes and properties in ontologies, where one can further restrict the ontology category (e.g., anatomy). BioPortal RESTful services offer several count-based metrics per ontology, e.g., the number of classes or properties. LOV also provides a RESTful service for term-based search over ontologies or terms, and a SPARQL endpoint. Although some repositories present values of metrics, these are not searchable. Other ontology repositories solely provide collections of ontologies without rich metadata (e.g., the Oxford Ontology Library, Protege Ontology Library, or Ontohub).

Rouane-Hacene et al. [10] used FCA for designing compact concept lattices transforming to ontology design patterns. This approach further applies Relational Concept Analysis adding relational attributes to classical binary attributes in FCA. Zhao [15] investigates using FCA for ontology building, mapping and visualization. Moreover, he also points out that semantic web techniques can be used for FCA applications. However, these approaches mainly analyze individual entities of ontologies we focus on ontologies as wholes, in the concept extent.

6. CONCLUSIONS AND FUTURE WORK

While similarity-based ontology search can contribute to the construction of either homogeneous or heterogeneous ontological benchmarks, FCA may rather help with recommendation of ontology metric value combinations shared by a number of ontologies. In OOSP we newly provide support for similarity-based ontology search. Regarding the applicability of FCA for ontological benchmark construction, we performed a preliminary investigation in terms of a priori ontology collection analysis. In all, FCA could provide important insights into potential ontology sets sharing attributes corresponding to values of ontology metrics. Although we experienced that attributes in concept intents are usually dependent on each other, this information could be potentially useful for construction of benchmarks of sufficient size also being richly described by different characteristics.

Since both cluster and FCA analyses enable us to categorize ontologies, it could also be interesting to compare them, considering that FCA applies discretization while clustering does not.[16] Further, since it is always difficult to select proper metrics for reasonable clustering, we could employ FCA in order to discover ontology metrics that are significantly related to each other. These metrics serve as input for cluster analysis, and more meaningful comparison of the clusters with concepts from FCA could be done. Finally, we plan to consider other discretization methods than equal-frequency and equal-width in the future, possibly accounting for the specific character of ontology collections on input.

7. ACKNOWLEDGEMENT

This work has been supported by the CSF grant no. 14-14076P, “COSOL – Categorization of Ontologies in Support of Ontology Life Cycle”.

8. REFERENCES

[1] Allocca C., d’Aquin M., Motta E. Finding equivalent ontologies in watson. In: ISWC 2008 Posters and Demonstrations. CEUR-WS. Vol. 401. 2008.

[2] Euzenat J., Valtchev P. Similarity-based ontology alignment in OWL-lite. In: ECAI. Vol. 16. 2004.

[3] Ganter B., and Wille R. Formal concept analysis: mathematical foundations. In: Springer Science & Business Media, 2012.

[4] Kalyanpur A., Parsia B., Sirin E., Cuenca Grau B. (2006). Repairing unsatisfiable concepts in OWL ontologies. In: 3rd European Semantic Web Conference 2006.

[5] Katifori A. et al.: Ontology visualization methods – a survey. In: *ACM Computing Surveys (CSUR).* 39(4), 10 pages, 2007, ACM.

[6] Kryszkiewicz M. Concise representations of association rules. In: Pattern Detection and Discovery. Springer Berlin Heidelberg, 2002. 92-109.

[7] Lehmann J., Bühmann L. (2010). ORE - a Tool for Repairing and Enriching Knowledge Bases. In: 9th International Semantic Web Conference 2010.

[8] Matentzoglu, N., Bail, S., Parsia, B.: A Snapshot of the OWL Web. In: 12th International Semantic Web Conference 2013.

[9] Rice M. D., Siff M. Clusters, concepts, and pseudometrics. In: *Electronic Notes in Theoretical Computer Science* 40. 323-346. 2001.

[10] Rouane-Hacene M., Huchard M., Napoli A., Valtchev P. Using formal concept analysis for discovering knowledge patterns. In: 7th Int’l Conf. on Concept Lattices and Their Applications. No. 672. 2010.

[11] Szathmary L., Valtchev P., Napoli A., Godin R. Constructing Iceberg Lattices from Frequent Closures Using Generators. In Proc. of the 11th Intl. Conf. on Discovery Science (DS ’08), 136–147.

[12] Valtchev P., Missaoui R., Lebrun P. A Fast Algorithm for Building the Hasse Diagram of a Galois Lattice. In Proc. of Colloque LaCIM 2000, pages 293-306, Montreal, Canada, 2000.

[13] Wu J. Z., Yu X., and Gao W. Distance computation of ontology vector for ontology similarity measuring and ontology mapping. *Journal of Difference Equations and Applications* (2016). 1-12.

[14] Zamazal O., Svátek V.: OOSP: Ontological Benchmarks Made on the Fly. In: Workshop SumPre’15 at ESWC 2015.

[15] Zhao Y. Using formal concept analysis for Semantic Web applications. Contributions to Ubiquitous Computing. Springer. 157-176. 2007.

[15] http://watson.kmi.open.ac.uk/

[16] While Rice and Siff did a general comparison about this [9], we plan to compare the methods specifically on ontology collections.