Criteria-Based Partitioning of Large Ontologies

Anne Schlicht, Heiner Stuckenschmidt
KR and KM Research Group, University of Mannheim
A5, 6 68159 Mannheim, Germany
{anne, heiner}@informatik.uni-mannheim.de

Categories and Subject Descriptors: I.2.11 Distributed Artificial Intelligence, D.2.7 Distribution, Maintenance, and Enhancement

General Terms: Ontologies, Modularization

1. INTRODUCTION

With the increasing use of ontologies in many branches of science and industry not only the number of available ontologies has increased considerably but also many widely used ontologies have reached a size that overburdens development and quality control procedures. It has been argued that the maintenance of large ontologies would be greatly facilitated by decomposing large ontologies into smaller modules that cover certain subtopics of the ontology. Another argument in favor of ontology modularization is the fact that very large ontologies sometimes cannot adequately be handled by the available ontology tools, some cannot even be loaded into an standard editor.

In our work, we focus on the task of splitting up an existing ontology into a set of modules according to some criteria that define the notion of a good modularization. Intuitively, we can say that a module should contain information about a coherent subtopic that can stand for itself. This requires that the concepts within a module are semantically connected to each other and do not have strong dependencies with information outside the module. These considerations imply the need for a notion of *dependency* between concepts that needs to be taken into account. The resulting model is the one of a weighted graph $O = \langle C, D, w \rangle$ where nodes C represent concepts and links D between concepts represent different kinds of dependencies that can be weighted according to the strength of the dependency. This leads us to a first central assumption underlying our approach:

> **Assumption 1:** Dependencies between concepts can be derived from the structure of the ontology.

Depending on the representation language, different structures can be used as indicators of dependencies. In previous work, we have shown that this assumption is valid in many cases [5]. A second basic assumption of our approach that directly follows from the first assumption and will be the focus of this paper is the following:

> **Assumption 2:** The Quality of a modularization can be determined on the basis of the structure of the individual modules and the connections between them.

This assumption does not only provide a justification for structure-based ontology partitioning, it also allows us to adapt the partitioning algorithm originally proposed in [6] by explicitly taking structural criteria for measuring the quality of the resulting modular ontology into account.

In the following, we investigate the partitioning of large ontologies into a set of interconnected modules based on structural quality criteria using the PATO tool. We revise and extend the original algorithm reported in [6] in several directions: 1) We generate an actual OWL ontology based on the assignment of concepts to modules and optimize the resulting model by allowing for overlap between modules. 2) We present a method for automatically selecting optimal parameters for the algorithm that maximize the quality of the result. 3) We show that the criteria can be adapted according to application needs or user preferences leading to different partitioning results that reflect the modified parameters.

All methods described in this paper have been implemented in the PATO system, a tool for partitioning OWL ontologies. PATO is available online[1].

K-CAP'07, October 28–31, 2007, Whistler, British Columbia, Canada.
ACM 978-1-59593-643-1/07/0010.

2. PARTITIONING ALGORITHM

In the following, we review the structure-based partitioning algorithm proposed in [6, 5] and propose a number of optimizations developed in the meantime. The resulting enhanced partitioning algorithm provides the basis for the work on criteria-driven ontology partitioning described in section 3.

Step 1.1: Create Dependency Graph. In the first step, a graph structure is created that represents the dependencies between elements in the ontologies. For RDF and OWL ontologies, nodes in the graph are values of "rdf:label" or "rdf:ID", weighted edges are created e.g. for subclass relations or classes that use the same property.

Step 1.2: Determine Strength of Dependencies. In the second step the proportional strength network (compare [1], page 54ff) for the dependency graph is computed, i.e. the edge weights are updated. In particular, the proportional strength for the edge connecting nodes c_i and c_j is $w(c_i, c_j) = (a_{ij} + a_{ji})/(\sum_k a_{ik} + a_{ki})$ where a_{ij} is the preassigned weight.

Step 2: Determine Modules. In the third step, the notion of a line island (a set of nodes for which the strength of the connection between the nodes inside the set is higher than the strength of any connection to nodes outside the set) is used to determine sets of ontology elements that should be in one module.

Step 3.1: Assign Isolated Concepts. After partitioning, in some cases there will be some leftover nodes which are not assigned to any cluster. These leftover nodes are assigned to the cluster to which they have the strongest connection.

Step 3.2: Merging In the next step it is decided whether some neighbored modules should be merged again because they are rather

[1] http://webrum.uni-mannheim.de/math/lski/Modularization/

strongly connected.

Step 3.3: Axiom Duplication. In some applications it is beneficial to duplicate axioms and include them in more than one module. Axioms are copied to neighboring modules if duplication decreases the connectedness [4] of the distributed ontology.

2.1 Criteria-Based Optimization

Investigation of applications for ontology modularization reveals that the criteria for determining a "good" partitioning depend heavily on the concrete application. For enabling adjustment to different application requirements, the parameters that influence the final partitioning are customizable in our partitioning tool. This means that the user may adjust the default setting to produce various resulting partitionings. For facilitating this adjustment we extended pato to automatically respond to given criteria defined in [4].

Based on a set C of criteria and their weights w_c Pato chooses the configuration p that maximizes the weighted sum of the criteria values $v_{c,p}$, i.e. it computes $\max_{p \in Config} \sum_{c \in C} w_c \cdot v_{c,p}$.

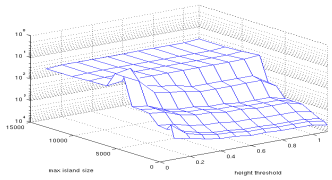
3. APPLICATION

We demonstrate partitioning and visualization using Pato on the NCI ontology, other ontologies are processed with analog results. The partitioning tool and additional examples are online available[1]. We assume two different applications with different requirements. First, we want to provide an overlook over the ontology. Second, for usage in an interactive semantic browser, modules are extracted that are small enough for visualization and further processing.

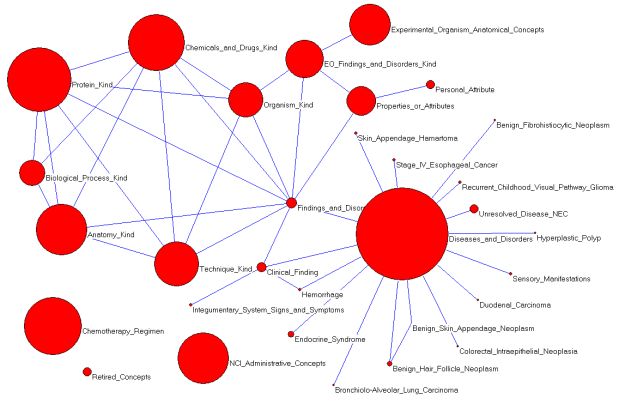
3.1 Visualization

Apart from the resulting OWL-modules, Pato generates networks that can be visualized using Pajek, a tool for large network analysis. The network shown in the figure below displays each module as a vertex, the size corresponding to the number of terms in the module. In addition to visualization, we used Pajek for determining the module labels. In particular, a module is labeled by the vertex with the highest *betweenness* (defined by [3] for social networks).

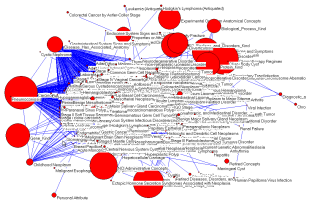
For succesful visualization of the whole ontology, the number of modules should be about 30 to provide as much information as can be displayed. Furthermore very large modules should be avoided. Therefore the criteria weights are set to 1 for *connectedness* and 2 for *abs(numberOfModules-30)*.



The weighted sum of the criteria values is maximal for the configuration *height threshold=0.2, max island size=7000*.



The module graphs displays the connections between modules.



Using the configuration that is optimal for the extraction application for creating a module graph is not recommended.

3.2 Extraction

We used Pato to extract modules from ontologies for the Semantic Web browser Magpie [2]. Magpie is a browser plugin, in which instances of ontology classes are identified in the current Web page and highlighted with the color associated to each class. In current work Magpie is extended towards open semantic browsing in which the employed ontologies are automatically selected and combined from online ontologies. Such an extension relies on mechanisms that not only dynamically select appropriate ontologies from the Web, but also extract from these ontologies the relevant and useful parts to describe classes in the current Web page.

For visualization of modules and online combination it is important that the modules are small. In addition, the created modules should be self contained as far as possible. Therefore the criteria weights are set to 1 for *connectedness* and 5 for *bulkyness*, resulting in appropriate modules.

4. CONCLUSION

We presented a partitioning method that automatically adapts to given requirements. The method is implemented in a flexible partitioning tool that produces different partitionings depending on the application it is used for.

Currently the optimal configuration is chosen from the set of configurations that were performed before. This is the basis for implementing an optimization mechanism that start with the default configuration and improves it step by step according to the given criteria performing a greedy strategy. Another possibility is starting at different random configurations like the walkSAT algorithm. For facilitating the specification of weights for the criteria we are planneing to extend the algorithm to support weight ranges. With this relaxation there is not a single optimal solution but a set of solutions, each corresponding to a possible weight assignment.

Acknowledgement

This work was partially supported by the German Science Foundation in the Emmy-Noether Program under contract Stu 266/3-1.

5. REFERENCES

[1] R.S. Burt. *Structural Holes. The Social Structure of Competition.* Harvard University Press, 1992.

[2] M. Dzbor, J. Domingue, and E. Motta. Magpie - towards a semantic web browser. In *International Semantic Web Conference (ISWC)*, 2003.

[3] Linton C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, 40(1):35–41, 1977.

[4] Anne Schlicht and Heiner Stuckenschmidt. Towards Structural Criteria for Ontology Modularization. In *Workshop on Modular Ontologies ISWC*, 2006.

[5] Heiner Stuckenschmidt. Network Analysis as a Basis for Partitioning Class Hierarchies. In *Workshop on Semantic Network Analysis ISWC*, 2006.

[6] Heiner Stuckenschmidt and Michel Klein. Structure-based partitioning of large concept hierarchies. In Sheila A. McIlraith, Dimitris Plexousakis, and Frank van Harmelen, editors, *Third International Semantic Web Conference (ISWC 2004)*, pages 289–303, Hiroshima, Japan, nov 2004.