Improving the Usability of Large Ontologies by Modularization

Anne Schlicht

Universität Mannheim, Germany
anne@informatik.uni-mannheim.de

1 Problem Definition

Recently, the benefits of modular representations of ontologies have been recognized by the semantic web community. With the growing utilization of ontologies in almost all 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 cannot be handled by the available reasoners (like the ontology for Diagnoses for Intensive Care Evaluation (DICE)), some cannot even be loaded into an standard editor e.g. the Foundational Model of Anatomy ontology (FMA). When system memory and computation time cannot be extended anymore the only feasibility to use existing tools is partitioning of the large ontology into smaller parts. Realization of this divide-and-conquer approach requires an infrastructure that supports reasoning over a distributed ontology and partitioning of a large ontology into a distributed ontology.

2 Related Work

The state of the art in ontology engineering is the usage of monolithic ontologies. Large ontologies are engineered by teams using editors like Protégé [1] with no support for modularization apart from the “owl:imports” construct that copies all axioms of one ontology into another. Maintenance and usage of ontologies is completely centralized in opposition to the idea of knowledge sharing that initiated the utilization of ontologies. There are tools for ontology development and a couple of reasoners (RACER [2], PELLET [3]) available but approaches to modular ontologies are still in the fledgling stages.

A modular ontology formalism is defined by every approach of mapping or importing ontologies because the modules of an ontology can be connected by links or import declarations.
The only tool that implements a distributed setting for DL i.e. performs reasoning tasks using different instances of a reasoner for different modules is the tableaux-reasoner DRAGO [4]. It uses the DL-reasoner pellet [3] and the modular ontology formalism Distributed Description Logic (DDL) [5]. This reasoner scales reasonably well but the underlying formalism is designed for the integration of existing ontologies. Consequently global completeness is not a desired feature of DDL as it is for representing an existing monolithic ontology in a modular form. Nevertheless, this approach provides a good basis for distributed reasoning in a decompositional setting.

$\mathcal{E}$-connections [6] were designed for composition, they can be used with the centralized DL-reasoner PELLET [3]. Using $\mathcal{E}$-connections for decomposition is implemented based on conservative extensions [7, 8] in the ontology editor SWOOP [9], but frequently fails partitioning i.e. the result is very often similar to the original ontology. Using $\mathcal{E}$-connections with another partitioning algorithm would suffer from the required semantical disjointness of local domains and impossibility to model subsumption links.
Package based description logic [10] uses modularization to enable collaborative development of ontologies. This approach is application focussed and provides an editor. It is structured as hierarchical owl import, extended by visibility management for concepts. Drawbacks of this approach are its limitation of module relations to owl imports and restriction to the $\mathcal{ALC}$ subset of DL.
Furthermore, some development tools for large ontologies implement ideas of modularization (e.g. "microtheories" of the CYC ontology [11]) but only with hierarchical i.e. transitive reuse of complete modules and customized for one specific ontology.
Every approach to modular ontologies imposes restrictions on the connections between modules like using a single link property or not being able to represent concept subsumption. Depending on the application these restrictions are often to strong for information preserving modularization of existing ontologies.
A promising reasoning algorithm was proposed independent of research on description logics [12]. This method improves performance of common first order resolution by partitioning the clauses. Obviously, the approach can be adapted for ontologies, the performance on description logic will be evaluated. A modification of partition based first order reasoning for propositional logic in a peer-to-peer setting was implemented in the SOMEWHERE system. It scales well and provides anytime reasoning but the poor expressivity limits applicability for DL.

3 Expected Contribution

The contribution of the thesis is an infrastructure for modular ontologies that improves usability of large[1] OWL-DL ontologies, solving the problems mentioned above. In particular the work will provide

- Representation
- Distributed Reasoning
- Partitioning Algorithms

Representation of modular ontologies includes definition of the connections between modules and communication protocol. The ontology modules and the links between them will be formulated in OWL-DL.
Reusing a large ontology usually requires first extracting the symbols and axioms that are relevant for the reusing application. On a modularized ontology this expensive and sophisticated extraction process is substituted by the much simpler task of selecting relevant modules.

[1] An ontology is considered "large" if the number of concepts and properties or the complexity of the structure impedes utilization with state of the art editors or reasoners.

4 Methodology and Status

Problem Definition. The problem that has to be solved is enabling usage and maintenance of large ontologies which is hardly possible at the moment.
Identification of Criteria for the Solution. Prior to analysing and evaluating the existing approaches we defined and classified the criteria for a good solution of the problem i.e. the criteria for a good modular ontology infrastructure [13].
Analysis of other Approaches. Based on the criteria and their importance we are now analysing the existing approaches guided by a translation of the modular ontology formalisms to DL. Now the main weaknesses and difficulties of approaches to modularity are detected. The results are related to properties of ontologies because performance of approaches generally depends e.g. on the ontology language or depth of the hierarchy.
Goal Definition. Guided by the analysis of existing approaches and available tools we defined the part of the problem that will be addressed in the thesis. This goal is an infrastructure for modular ontologies that supports distributed reasoning and partitioning.
Evaluation Framework. In order to focus research it is helpful to set up the evaluation at a early stage so it can guide the work. Evaluating the results of this work on modularization consists of verifying the correctness of the implemented algorithms and comparing the performance to state of the art centralized and distributed reasoners.
Evaluate other Approaches. After the evaluation is set up existing approaches can be evaluated to further reveal virtues and drawbacks of design alternatives.
Design. The preceding analysis and clear definition of the goal shows on which formalism the work will be based and which tools are to be reused and extended. The different parts of the work are listed and related to conclude a plan, that is then carried out.
Evaluation. The last step is evaluation of correctness and performance of the implementation using ontologies of different size, language, expressivity and density.
Developing complex algorithms is usually an iterative process. Evaluation reveals weaknesses and bugs leading one step back to redesigning the algorithm accordingly. Sometimes it may even be necessary to readjust the goals if they turn out to be to hard or are fully achieved by other implementations.

Problem definition and identification of criteria where carried out within the past nine month since I started my PhD. Currently I am analysing other approaches and defining the goal of the thesis.

5 Approach

For the development of an infrastructure for modular ontologies the crucial decision is on the linking language to be used. To guarantee the right choice we analyse the existing approaches with respect to their expressivity, tractability and extendability. This analysis is based on a translation of the mapping languages DDL and $\mathcal{E}$-connections as well as the import approaches P-DL and Semantic Import to common description logic. The mapping approaches are already translated by their developers [14, 5], we are working on the translation of import approaches.
The most promising approach though still incomplete is Semantic Import [15], this

work will be continued in cooperation with the inventors to develop a formalism for modular ontologies. Representation in DL will presumably reveal that import and mapping are semantical equivalent i.e. they can be defined in terms of each other. On the other hand the differences in expressivity of the different approaches and the corresponding computational properties are much more clear when formulated in DL. Every approach imposes certain syntactic restrictions on an ontology that is to be converted to that type of modular ontology. Mainly these are restrictions on the ontology language or the mapping/import language. For example local domains must be definable such that there are no properties connecting elements of different local domains[2], or (in case of $\mathcal{E}$-connections) these properties cannot be transitive. Modular ontologies are not formulated in unrestricted DL because it is very hard (maybe impossible) to find a tractable reasoning algorithm for unrestricted DL in a distributed setting. Especially the combination of intersecting local domains with the above type of property raises difficulties. Thus developing a modular ontology formalism inevitably means trading completeness for tractability. There is not a single formalism that is the right trade-off for all situations but a set of different formalisms that reflect the relative importance of completeness vs. tractability. The existing approaches can be viewed as part of this set, but there are many other restrictions to evaluate on different reasoning tasks for a better trade-off and to fill the gap for less restricted modular DL.
The second part of the work is the development of a distributed reasoner that can handle large distributed ontologies. For unrestricted distributed DL we cannot expect to find an efficient reasoning algorithm, but in some applications for which time is not a sparse resource even an unefficient satisfyability test would be very helpful.
There are two starting points for developing a global complete distributed reasoner. One is the reasoner DRAGO [4], a distributed tableaux-reasoner based on the DL-reasoner PELLET [3]. This reasoner scales reasonably well and is currently the only approach to actually distributed reasoning. Centralized reasoner inevitably fail on large ontologies because they cannot even load them. The other starting point to distributed reasoning are results from distributed first order reasoning [12], the tractability of these methods on DL will be investigated.
Distributed reasoning trades decreased memory requirements for increased communication time and some optimization methods that are used in centralized reasoning cannot be applied to a distributed knowledge base. Hence critical for the performance of distributed reasoning is the partitioning of the knowledge base because it determines communication time and the amount of time a module spends waiting for information from other modules. Effects of the partitioning were not considered for Drago because it was designed for the composition of existing ontologies. Thus, optimizing the partitioning for reasoning with Drago may greatly reduce computation time. Furthermore the time problem will be addressed by implementing an anytime reasoning algorithm that considers increasing parts of the modular ontology. In the first step only the current module is checked for inconsistencies, the next step includes all direct predecessors.
To facilitate application of the developed distributed reasoner we aim at providing it as plug-in interface for the ontology editor Protégé [1].

[2] i.e. property assertions $p(x, y)$, $p(x, z)$ are not permitted if y and z are from different local domains.

The first step of the evaluation is verification of the implemented reasoning algorithms using ontologies that are small enough for existing DL-reasoners to provide reference for comparision of reasoning tasks. After verification the performance of developed reasoners will be compared to centralized reasoning and related to the applied restrictions. Experiments will be carried out using very large real-life ontologies like the DICE ontology and the FMA ontology. The time requirements will be compared to those of Drago and other distributed reasoners that are available in the meantime. Based on this evaluation it will be decided which of the implemented algorithms provide a reasonable trade-off between completeness and tractability for what type of ontology.

References

1. Noy, N., Sintek, M., Decker, S., Crubezy, M., Fergerson, R., Musen, M.: Creating semantic web contents with protege-2000. Intelligent Systems **16**(2) (2001) 60– 71
2. Wessel, M., Möller, R.: A high performance semantic web query answering engine. In Horrocks, I., Sattler, U., Wolter, F., eds.: Proc. International Workshop on Description Logics. (2005)
3. Sirin, E., Parsia, B., Grau, B.C., Kalyanpur, A., Katz, Y.: Pellet: A practical owl-dl reasoner. Jounal of Web Semantics (2006) to appear.
4. Serafini, L., Tamilin, A.: DRAGO: Distributed reasoning architecture for the semantic web. In: Proc. of the Second European Semantic Web Conference (ESWC'05). (2005)
5. Borgida, A., Serafini, L.: Distributed description logics: Assimilating information from peer sources. Journal of Data Semantics **1** (2003) 153–184
6. Grau, B.C., Parsia, B., Sirin, E.: Combining owl ontologies using e-connections. Journal Of Web Semantics **4**(1) (2005)
7. Lutz, C., Walther, D., Wolter, F.: Conservative extensions in expressive description logics. In: Proceedings of the Twentieth International Joint Conference on Artificial Intelligence IJCAI-07, AAAI Press (2007)
8. Grau, B.C., Horrocks, I., Kazakov, Y., Sattler, U.: A Logical Framework for Modularity of Ontologies. In: Twentieth International Joint Conference on Artificail Intelligence (IJCAI). (2007)
9. Grau, B., Parsia, B., Sirin, E., Kalyanpur, A.: Automatic Partitioning of OWL Ontologies Using E-Connections. In: Proc. of Description Logic Workshop (DL). (2005)
10. Bao, J., Caragea, D., Honavar, V.: Towards collaborative environments for ontology construction and sharing. In: International Symposium on Collaborative Technologies and Systems. (2006)
11. Curtis, J., Matthews, G., Baxter, D.: On the effective use of cyc in a question answering system. In: IJCAI Workshop on Knowledge and Reasoning for Answering Questions. (2005)
12. Amir, E., McIlraith, S.: Partition-based logical reasoning for first-order and propositional theories. Artificial Intelligence **162**(1-2) (2005) 49–88
13. Schlicht, A., Stuckenschmidt, H.: Towards Structural Criteria for Ontology Modularization. In: Proc. of the ISWC 2006 Workshop on Modular Ontologies. (2006)
14. Grau, B.C., Kutz, O.: Modular ontology languages revisited. In: IJCAI Workshop on Semantic Web for Collaborative Knowledge Acquisition. (2007) to appear.
15. Pan, J.Z., Serafini, L., Zhao, Y.: Semantic import: An approach for partial ontology reuse. In: Proc. of the ISWC 2006 Workshop on Modular Ontologies. (2006)