A T-Box Generator for testing scalability of OWL mereotopological patterns

Martin Boeker[1⋆], Janna Hastings[2], Daniel Schober[1], and Stefan Schulz[3]

[1] Institute of Medical Biometry and Medical Informatics,
University Medical Center Freiburg, Freiburg, Germany
[2] Chemoinformatics and Metabolism,
European Bioinformatics Institute, Hinxton, UK
[3] Department of Medical Informatics, Medical University Graz, Graz, Austria

Abstract. The representation of biomedical structure - from cellular components to organisms - in biomedical ontologies is of pivotal importance, as the internal structure of complex structured objects needs to be referenced in the definition of processes, disorders, phenotypes and many other entities. Yet, most of the existing biomedical ontologies do not contain logical axiomatizations for accurately representing the internal structure.

We have identified the high importance of mereotopology (parthood, connectedness) for accurate representation in this domain, but the representation of mereotopological structure can provide challenges for reasoners. To evaluate the scalability of accurate representation of biomedical structure, we have identified design patterns for (i) parthood, both one-sided, two-sided and cardinality restricted, (ii) class disjointness, and (iii) spatial disconnectedness. In order to evaluate the DL reasoning performance for these patterns, we have created a T-Box Generator to programmatically generate small and large experimental T-Boxes with different reasoning complexities resulting from the relative proportions of the patterns (i) to (iii).

Classification times have been measured for different reasoners in their most common application settings. We found that, as expected, reasoning times increased dramatically with the size and complexity of the generated ontology, and furthermore, even small numbers of cardinality restrictions were a major performance killer.

1 Introduction

It has been repeatedly emphasized that the representation of biomedical structure - encompassing a broad range from the material constitution of cells and cell organelles to anatomies of animals and plants, even hospital buildings and their departments, is of fundamental importance in biomedical ontology. There is hardly any biomedical ontology or terminology which does not refer to structural material entities, as they are the location of physiological, pathological,

⋆ To whom correspondence should be addressed: martin.boeker@uniklinik-freiburg.de

and clinical processes, therapeutic or experimental interventions, as well as the bearers of functions, dispositions, and qualities.

Most OWL-based ontologies currently represent structural entities in a double hierarchy, viz. a taxonomic order paralleled by a partonomy. We find examples for this in the Foundational Model of Anatomy (FMA) [7], as well as in other OBO anatomy ontologies such as the Adult Mouse Anatomy, and the Cellular Component hierarchy of the Gene Ontology (GO) [10].

There are basically two kinds of assertions found in these parallel hierarchies:

- **Taxonomic parents**: A `subclassOf` B, e.g. *Heart* `subclassOf` *Cavitated-Organ*
- **Part implies whole**: All members of the class P are parts of some W. P `subclassOf` *properPartOf* `some` W, e.g. *Heart properPartOf* `some` *CardiovascularSystem*

Such simple representations are not sufficient for properly expressing axioms such as the following:

- **Mutual disjointness**: There are no entities which are both members of class A and members of class B; e.g. there are no entities which are both *nerve cells* and *blood cells*.
- **Spatial disjointness**: No member of class A spatially overlaps with any member of class B, or stricter; no member of A has a part which is also part of any B. Examples: Nothing is totally located in any *liver* and any *kidney*; nothing which is part of some *right arm* can be part of some *left arm*.
- **Whole implies part**: All members of the class W have some member of P as part, e.g. all *cell membranes* have some *lipids* as part. Note that P being a mandatory part of W does not mean that W is a mandatory whole for P.
- **Counts of parts**: Class W has exactly n distinct members of class P as parts, e.g. all *hands* have exactly five *fingers* as parts.

The contributions of this paper are two-fold. Firstly, we define patterns for the representation of these aspects of biomedical structure in OWL. Secondly, we provide an ontology performance evaluation tool, in the form of a TBox generator for different sizes and complexities of ontologies accommodating these patterns. We use this tool to investigate the scalability of reasoning over these patterns through generating ontologies of different sizes and complexities. The advantage of this approach is that ontology developers can *pre-test* an ontology in the design phase, before implementation, based on the expected complexity and size.

2 Patterns for the representation of biomedical structure

The following real world examples from the GO, Mouse Anatomy and FMA illustrate ontology content patterns for partonomy representation:

- The textual definition of *Gastrointestinal tract* in the FMA is as follows:

> Hollinshead's 97:528 - From the foregut will differentiate the esophagus, the stomach, and the proximal half of the duodenum. Two buds appear on the caudal portion: a ventral diverticulum gives rise to the liver, gallbladder, and a portion of the pancreas; a dorsal diverticulum grows into the dorsal mesentery and gives rise to the remaining and major part of the pancreas. From the midgut are derived the second half of the duodenum, the jejunum, ileum, cecum and appendix; ascending colon, and most of the transverse colon. From the hindgut are derived the terminal portion of the transverse colon, the descending and sigmoid colon, and the rectum.

This textual definition shows some good examples of spatial connection, partonomy with cardinality, and spatial disjointness, which are not captured in the formalization of the FMA in OWL.

- The Adult Mouse Gross Anatomy ontology has many examples of partonomies, e.g., *nervous system* has part *central nervous system (CNS)* and other parts, while the *CNS* in turn has part *white matter*, *grey matter* and other parts. It includes no explicit disjoints or spatial disjoints in its representation.
- The GO Cellular Component ontology has divisions at a high level between classes (X) and classes for parts ($X part$), examples of which are *cell/cell part*; *extracellular region/ extracellular region part*; *virion/ virion part*. In other words, the ontology exactly follows the taxonomy/ partonomy distinction but without explicit disjoints asserted. As is the case with the FMA, the textual definitions give much more information than is encoded in the formal axioms. For example, *sperm individualization complex* is defined as follows:

  > A macromolecular complex that cytoskeletal components and part of the cell membrane, forms at the nuclear end of a male germline syncytium, or cyst, and translocates the over the length of the syncytium in the course of sperm individualization. Each complex contains an array of 64 investment cones, one per nucleus, that move synchronously along the spermatogenic cyst.

We express mutual disjointness by the OWL2 DisjointClasses predicate, which corresponds to the set of axioms:

$$\texttt{DisjointClasses}\ (C_1, C_2, \ldots, C_n) =_{def} \{C_1\ \texttt{subclassOf not}\ C_2; \ldots; C_1\ \texttt{subclassOf not}\ C_n; C_2\ \texttt{subclassOf not}\ C_n; \ldots\} \quad (1)$$

Spatial disjointness requires a more complex axiomatization, and there is no OWL predicate for this: It requires the expression of the condition that nothing located in any instance of *Class1* (e.g. *UpperLobe*) is located in any instance of *Class2* (e.g. *LowerLobe*) and vice versa:

$$C_1\ \texttt{subClassOf}\ \mathit{locusOf}\ \texttt{only}\ (\texttt{not}\ (\mathit{hasLocus}\ \texttt{some}\ C_2)) \quad (2)$$

$$C_2\ \texttt{subClassOf}\ \mathit{locusOf}\ \texttt{only}\ (\texttt{not}\ (\mathit{hasLocus}\ \texttt{some}\ C_1)) \quad (3)$$

with *locusOf* being the inverse of the transitive and reflexive relation *hasLocus*, as the most general spatial inclusion relation. Additionally,

$$\texttt{DisjointClasses}(C_1, C_2). \tag{4}$$

In our framework the transitive and irreflexive mereological relations *properPartOf* and *hasProperPart* are subrelations of *hasLocus* and *locusOf*, respectively.

Whole implies part (see example above) shows that partonomies are more complex compared to taxonomies. It is not sufficient to say that C_1 and C_2 are in a partonomic order, as the following cases need to be distinguished:

- *Class1* `subClassOf` *properPartOf* `some` *Class2* ("part implies whole")
- *Class2* `subClassOf` *hasProperPart* `some` *Class1* ("whole implies part")

and the combination of both of the above (two-sided parthood).

3 The T-Box Generator: automatically creating ontologies of different sizes and complexities

We developed a T-Box Generator to generate ontologies resembling real world ontologies differing in a variety of parameters. The generator was programmed in the object functional language Scala[4]. The script can be downloaded from http://www.imbi.uni-freiburg.de/ontology/t-box-generator.zip. The generation of different versions of ontologies can be controlled by command line parameters which enable the batch generation of groups of ontologies.

The T-Box Generator allows the following parameters to be controlled:

- The number of levels in the is_a hierarchy.
- The number of subclasses of each superclass. E.g., with three levels of is_a hierarchy and 10 subclasses a total of $10^3 = 1000$ classes will be generated.
- The number of mutually disjoint classes in each group of subclasses. The mutually disjoint classes are also the target of the *hasPart* relation of the partonomy when generated and are the source of *partOf* relations.
- The number of mutually spatially disjoint classes characterized by the spatially disjoint pattern.
- The creation of a partonomy in the ontology. If the partonomy is created the first class in each subclass group is the source of *hasPart* relations to all mutually disjoint classes in the same group of subclasses as defined above and is the target for *partOf* relations from these classes. The relations of the partonomy can further be controlled.
- The partonomy can either be created as subclasses axiom or equivalent classes axioms for the *hasPart* relation outgoing from the first class in each group of subclasses.

[4] http://www.scala-lang.org/

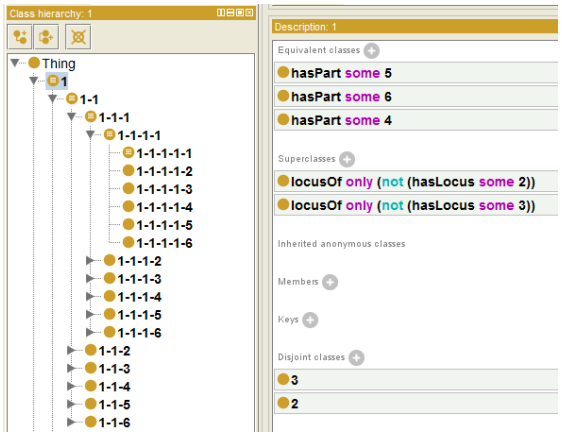

Fig. 1. The figure displays part of a generated ontology with 5 hierarchical levels and 6 subclasses per superclass. From these 6 classes three are mutually disjoint and three are spatially mutually disjoint which includes mutual disjointness. The partonomy has been built as equivalent classes with existential quantification. The target classes of the *hasPart* relation are the mutually disjoint classes from the same group of subclasses.

- The quantification for the *hasPart* relation outgoing from the first class in each group of subclasses can be either set to existential or to an exact cardinality. When it is set to an exact cardinality the transitivity switch (below) is overriden so that the partonomic relations have no transitivity property.
- The pairs of relations *hasLocus* – *locusOf* and *hasPart* – *partOf* can be set as inverse relations.
- The transitivity property for the relations *hasPart*, *partOf*, *hasLocus* and *locusOf* can be set on or off.
- The filename under which an ontology is saved can be set.

This set of parameters allows the generation of a variety of ontologies with typical features of biomedical ontologies to test for performance issues with the given set of ontology tools comprised by the ontology editor Protégé and the DL reasoners Pellet, HermiT and Fact++. The advantage of this approach is that ontology developers are able to *pre-test* an anticipated ontology for performance issues before they actually run into them.

A generated ontology with five levels of is_a hierarchy and six subclasses for each superclass is illustrated in Figure 1.

4 Reasoning scalability: results

We tested three of the most common reasoners that integrate with Protégé and are under an open source license i.e., Fact++ (version 1.5.2) [11], HermiT

hierarchical levels	number of subclasses	total number of classes	number of disjoint classes	n. of spatially disjoint classes	partonomy	subclasses or equivalent classes	object relations invers/ transitiv	quantification	complexity level	Fact++ 1.5.2 [s]			HermiT 1.3.3 [s]			Pellet 2.2.2 [s]		
5	6	7776	0	0	no				1	0.5	0.3	**0.3**	2.6	1.5	**0.7**	1.5	**0.6**	0.3
			3	3	yes				2	0.9	0.7	**0.7**	6.0	2.9	**1.8**	4.0	2.1	**1.1**
						sub	yes		3	1.2	1.1	**1.1**	13.1	**7.6**	8.2	7.3	3.4	**2.2**
						eq	no	existential	4	14.9	14.8	**14.5**	10.8	7.7	**6.0**	**> 10 min**		
							yes		5	24.9	25.0	**24.9**	25.5	**18.3**	18.8	**> 10 min**		
								cardinality 1	6	17.1	17.1	**17.1**	13.0	7.4	**5.8**	**out of memory**		
								cardinality 2	7	**> 10 min**			289.4	**276.4**	277.7	**out of memory**		
3	20	8000	0	0	no				1	0.5	0.3	**0.3**	2.4	1.4	**0.5**	1.4	0.5	**0.4**
			10	10	yes				2	2.1	1.7	**1.6**	10.6	5.5	**3.8**	6.7	3.6	**2.5**
						sub	yes		3	2.6	2.2	**2.0**	16.5	9.3	**7.6**	9.3	5.4	**4.9**
						eq	no	existential	4	15.4	15.4	**15.4**	15.1	**8.5**	10.2	**> 10 min**		
							yes		5	22.6	22.4	**22.3**	23.3	17.1	**15.9**	**> 10 min**		
								cardinality 1	6	**> 10 min**			28.2	20.5	**18.0**	**> 10 min**		
								cardinality 5	7	**> 10 min**			**out of memory**			**> 10 min**		
2	90	8100	0	0	no				1	0.5	0.2	**0.3**	2.4	1.2	**0.8**	1.2	0.5	**0.4**
			45	45	yes				2	6.6	6.7	**6.7**	39.4	**26.1**	28.3	20.5	16.3	**14.1**
						sub	yes		3	8.9	8.7	**8.4**	38.5	33.2	**30.4**	35.3	26.7	**25.6**
						eq	no	existential	4	34.1	33.5	**33.5**	44.1	39.6	**29.8**	**> 10 min**		
							yes		5	60.0	59.6	**59.3**	57.3	**42.5**	44.5	**> 10 min**		
								cardinality 1	6	**> 10 min**			418.2	388.4	**387.4**	**out of memory**		
								cardinality 22	7	**out of memory**			**out of memory**			**out of memory**		

Table 1. The classification times of three sequential measurements for three groups of seven ontologies are shown. The groups of ontologies differ in the depth of is_a hierarchies and number of subclasses per superclass. The number of mutually disjoint and spatially disjoint classes is kept to 50% of the number of subclasses. The ontologies in each group differ by various parameters in increasing complexity.

(version 1.3.3) [8] and Pellet (version 2.2.2) [9]. CB[5], a fast reasoner which does not currently integrate with Protégé 4, and JCel [6], an EL^+ compliant reasoner which was not applicable to all test ontologies, were not included in the test set.

We measured reasoning speed with the default settings for the Protégé 4 reasoner configuration tab, as we assume that many ontology developers in the biomedical domain will use these defaults. These are: all switches for class inferences, all switches for object property inferences, no switches for datatype property inferences and no switches for individual inferences set to active. The measurement was performed on a computer with a 1.6GHz Intel(R) Core(TM) i7 CPU Q720 with 4GB RAM under Windows 7 64-Bit and Java 64-Bit (Version 1.6.25). The Protégé installation was version 4.1 RC2 build 228.

Protégé was started with the Java command line switch -Xmx3000m resulting in an effective memory allocation of 2796MB. For each reasoner and each ontology three consecutive measurements were performed. After each series of measurements Protégé was closed and Java unloaded from memory to avoid any unwanted effects of Java's automatic memory management on the reasoning performance. The results are illustrated in Table 1. For a graphical representation of the results see Fig. 2.

[5] http://code.google.com/p/cb-reasoner/
[6] http://lat.inf.tu-dresden.de/systems/cel/

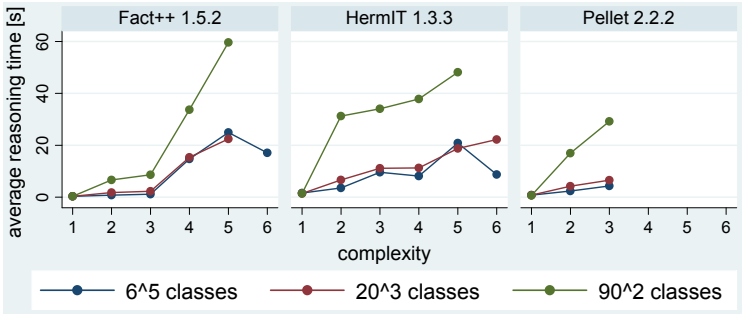

Fig. 2. Reasoning performance grouped by reasoner as well as level of hierarchies and number of subclasses respectively. For the description of complexity levels refer to Tab. 1. Outliers in the complexity level 7 are omitted.

5 Discussion

Many of the current biomedical ontologies modeling aspects of biological structure provide underconstrained logical axioms for the parthood and connectedness of the structured objects they contain. This is reflected by a disconnection between the information content of the textual definitions, which can be quite detailed, and the logical axioms, which tend to reflect a simplistic partonomy and taxonomy, without further axiomatization. Our first contribution is the identification of patterns for representing the missing structural complexity in bio-ontologies.

The patterns we have identified fall short of what is needed to fully represent the domain of mereotopology, since they do not allow the representation of complex structural interrelationships between parts of a whole, in particular where such interrelationships can form *cycles*. An anatomical example of such a structured object is the human heart, which can be simplistically described in terms of a left and right atrium and a left and right ventricle, each of which is connected to each other. For reasons described in [2], OWL models of this type of structure are underconstrained. Another source of complexity that we do not capture in the presently proposed set of patterns is that complex wholes can be partitioned into parts in differing ways and at different levels of granularity [3]. While this does not present a problem for a simplified representation of the overall partonomy such as is used in current bio-ontologies such as GO, it starts to become a problem when, for example, cardinality constraints are used to constrain the overall number of parts (as this can differ in different divisions of the whole into parts).

Certain recent OWL extensions allow the representation of arbitrarily structured objects, such as rules [1] and description graphs [5]. However, these formalisms require strong conditions on how the underlying ontology is to be modeled to remain within what is decidable in the underlying logic, for example, enforcing strict property separation between description graphs and OWL axioms,

and restriction only to known individuals in the body of the rules [2]. Our approach focuses on evaluating the pragmatic tractability of certain patterns while remaining well within the standard, decidable, DL fragment of OWL 2.

Furthermore, we have hypothesized that large-scale migration to more axiomatic representation of biomedical structure might be accompanied by a prohibitively large performance decrease on the side of the DL-based reasoners commonly used in ontology development. Should this be the case, it stands as an obstacle for the migration of existing large-scale ontologies towards greater semantic complexity. To evaluate the performance consequences on reasoners for the different complexity patterns we implemented a T-Box generator that automatically creates test ontologies according to the patterns.

Although the idea of ontology generators in general is not new (see, for example, [6] or the OWL DL Generator[7]), many of those are restricted to a certain semantics and do not allow expressivity parameter adjustment necessary for mereotopology pattern testing. Our T-Box generator allows for control over many parameters that other generators miss and which furthermore allows specifically for mereotopology performance testing. With the above described functionality it could extend current Ontology Editor Tools and could be easily provided as e.g. a Protégé 4 plugin. Another possibility is the inclusion in existing OWL DL benchmarking frameworks [4].

The reasoning time acceptable for a particular ontology engineering effort is dependent on the time available during the ontology life cycle and the phase in which reasoning is applied. Examples of reasoning tasks might be to apply a consistency check before ontology release, or after any change at development time, or even a consistency check used as part of the ontology public user interface in which reasoning forms part of an application use case.

At development time, the time allowed for reasoning is dependent on the frequency at which reasoning is to be carried out, which is a function of the number of people performing changes on the ontology over time and on their expertise level. Having only one knowledgeable developer might require fewer reasoning checks as when three novices are actively changing the ontology. The acceptance of reasoning time is proportional to the ratio of the development session time to reasoning time, e.g. for a one hour editing session 30 minutes reasoning time is certainly not acceptable, whereas for an 8 hour session it might seem reasonable.

As regards our performance results, we found an expected decrease in reasoner performance with increase in ontology complexity, and in particular a dramatic decrease in performance with the use of cardinality constraints. In order to still be able to make the best out of the available resources, we suggest structural simplifications of the desired semantic complexity i.e. the transformation from a complex representation to a less complex and more performant representation.

[7] http://knowledgeweb.semanticweb.org/benchmarking_interoperability/-OWLDLGenerator/

6 Conclusion

For the modeling in the biomedical domain mereotopology (i.e. parthood and connectedness) is of high importance for accurate representation. Together with the considerable size of ontologies in the biomedical domain this can provide challenges for reasoners.

To evaluate and predict the reasoning performance of different reasoners we developed a T-Box generator which allows for the parameterized creation of ontologies. As expected, reasoning time increases with growing complexity and size of the ontology. Further research will include other reasoners and a larger set of ontology patterns.

Acknowledgments

This work was partly supported by the Deutsche Forschungsgemeinschaft (DFG) grant JA 1904/2-1, SCHU 2515/1-1 GoodOD (Good Ontology Design).

References

1. Glimm, B., Horridge, M., Parsia, B., Patel-Schneider, P.F.: A syntax for rules in OWL 2. In: Proc. of OWL Experiences and Directions 2009 (OWLED 2009) (2009)
2. Hastings, J., Dumontier, M., Hull, D., Horridge, M., Steinbeck, C., Sattler, U., Stevens, R., Hörne, T., Britz, K.: Representing chemicals using OWL, description graphs and rules. In: Proc. of OWL: Experiences and Directions (OWLED 2010) (2010)
3. Jansen, L., Schulz, S.: Grains, components and mixtures in biomedical ontologies. Journal of Biomedical Semantics to appear (2011)
4. Ma, L., Yang, Y., Qiu, Z., Xie, G., Pan, Y., Liu, S.: Towards a Complete OWL Ontology Benchmark. In: The Semantic Web: Research and Applications, chap. 12, pp. 125–139. Lecture Notes in Computer Science
5. Motik, B., Cuenca Grau, B., Sattler, U.: Structured objects in OWL: representation and reasoning. In: Proc. of the 17th International World Wide Web Conference (WWW 2008). ACM, Beijing, China (2008)
6. Ongenae, F., Verstichel, S., De Turck, F., Dhaene, T., Dhoedt, B., Demeester, P.: OTAGen: A tunable ontology generator for benchmarking ontology-based agent collaboration. Computer Software and Applications COMPSAC 08 (2008)
7. Rosse, C., Jr, M.J.: The foundational model of anatomy ontology. In: Burger, A., Davidson, D., Baldock, R. (eds.) Anatomy Ontologies for Bioinformatics: Principles and Practice, pp. 59–117. Springer, London (2007)
8. Shearer, R., Motik, B., Horrocks, I.: HermiT: A highly-efficient OWL reasoner. In: Dolbear, C., Ruttenberg, A., Sattler, U. (eds.) Proceedings of the 5th Workshop on OWL: Experiences and Directions. Karlsruhe, Germany (2008)
9. Sirin, E., Parsia, B., Cuenca Grau, B., Kalyanpur, A., Katz, Y.: Pellet: A practical OWL-DL reasoner. Journal of Web Semantics 5, 51–53 (2007)
10. The Gene Ontology Consortium: Gene ontology: tool for the unification of biology. Nat. Genet. 25, 25–9 (2000)
11. Tsarkov, D., Horrocks, I.: FaCT++ description logic reasoner: System description. In: Proc. of the Int. Joint Conf. on Automated Reasoning (IJCAR 2006). pp. 292–297. Springer (2006)