OWLDEF: Integrating OBO and OWL

Robert Hoehndorf[1], Anika Oellrich[1], Michel Dumontier[3], Janet Kelso[4], Heinrich Herre[2], Dietrich Rebholz-Schuhmann[1]

[1]European Bioinformatics Institute
[2]Institute for Medical Informatics, Statistics and Epidemiology, University of Leipzig
[3]Department of Biology, Institute of Biochemistry and School of Computer Science, Carleton University
[4]Department for Evolutionary Genetics, Max Planck Institute for Evolutionary Anthropology

ABSTRACT

An integration of the OBO Flatfile Format and the Web Ontology Language (OWL) would enable automated reasoning, inferences and consistency checking of biomedical ontologies and support the development and maintenance of ontologies developed in the OBO Flatfile Format. So far, the translation of relations in the OBO language to OWL is performed according to a single rigid pattern and in violation of the relation definitions of the OBO Relationship Ontology. We extend both the OBO Flatfile Format and the Manchester OWL Syntax to accommodate relation definitions. Based on these extensions, we implemented and evaluated two software applications. The first converts the OBO Flatfile Format to an OWL representation. The second uses automated inferences to convert OWL ontologies back to a representation in the OBO Flatfile Format. The OWLDEF method is generally applicaple whenever ontologies are developed primarily using patterns and not a detailled knowledge representation language. The tools and libraries we developed for the OWLDEF method are available from `http://bioonto.de/obo2owl`.

1 INTRODUCTION

Directed acyclic graphs (DAG) have been a popular representation format for biomedical ontologies. The Gene Ontology (GO) originally has been developed in the form of a DAG (Ashburner et al. 2000). The *OBO Flatfile Format* (OBOF) (Horrocks 2007, Mungall and Day-Richter 2008), a graph-based knowledge representation language, was derived from the representation format of the GO. Currently, many ontologies in the biomedical domain are developed in the OBOF, in particular the OBO and OBO Foundry ontologies (Smith et al. 2007). Furthermore, due to its simplicity and the availability of tools, the OBOF is being used in other domains where ontologies are primarily developed by domain experts instead of logicians.

In the OBOF, nodes represent ontological categories and edges represent relations between these categories. The OBO Relationship Ontology (RO) provides formal definitions for commonly used relations between ontological categories (Smith et al. 2005). Currently, no explicit semantics for the OBOF is available that can accommodate the relation definitions from the RO.

We developed an extension to the OBOF and to the Manchester OWL Syntax (Horridge et al. 2006) based on the assumption that any statement in OWL in which two variables for classes occur, determines a relation between these two classes. Based on this assumption, we provide a novel implementation of the RO in OWL and a software application to convert OBO ontologies to OWL. Furthermore, we provide another software application which uses OWL reasoning to infer new binary relations between classes. Our method and software applications lead to an integration of the OBOF with OWL while maintaining the semantics for relations provided by the RO.

2 METHODS

We developed the OWLDEF method which combines the OBOF, the RO and OWL reasoning to provide a flexible and extensible semantics for biomedical ontologies. OWLDEF consists of three parts. First, we provide an extension to the syntax and semantics of the Web Ontology Language (OWL). Second, we extend the syntax of the OBOF to incorporate complex definition patterns for relations between classes. Third, we provide an implementation of the RO based on our previous extensions.

In the first part we focus only on one human readable syntax of OWL and extend it with the means to define binary relations between classes. We extended the Manchester OWL Syntax (Horridge et al. 2006) by adding two symbols, `?X` and `?Y`. Both are intended to be variable symbols that represent OWL *classes*. We have extended the OWL semantics to include an interpretation of these variable symbols (Hoehndorf 2009). Intuitively, the variables range only over the *named* classes in the signature of the OWL ontology. Therefore, using these symbols and the semantics we provide does not lead to a proper extension of OWL expressivity.

Based on this extension of the Manchester OWL Syntax, we assume that any OWL class axiom that can be formulated using this extended form of the Manchester OWL Syntax defines a relation between two classes (between `?X` and `?Y`). For example, the statement `?X SubClassOf: ?Y` defines the **is-a** relation, and the statement `?X SubClassOf: part-of some ?Y` may define the **part-of** relation between two classes.

The OBOF provides a means to express relations between classes, yet it does not enable the definition of the relations themselves. In the OWLDEF method, we use OWL axioms in our extended syntax to define relations between classes in the OBOF. For this purpose, we extend the `Typedef` environment in the OBOF to include the definition of relations. For example, to define the relation **has-part**, we use the following `Typedef` statement in the OBOF:

```
[Typedef]
id: has-part
name: has-part
owldef: ?X SubClassOf: has-part some ?Y
```

According to our semantics, every *use* of the relation **has-part** in the OBOF is expanded to an OWL axiom in which the variables are filled by the classes between which the relation was asserted. For example, the statement that every mouse body has some tail as part in the OBOF is:

```
[Term]
id: Mouse_body
relationship: has-part Tail
```

Using the OWLDEF method, *Mouse_body* and *Tail* fill `?X` and `?Y`, respectively. The resulting OWL axiom would be

```
Mouse_body SubClassOf: has-part some Tail
```

Based on this approach, we provide definition patterns for relations in the RO. The patterns are shown in table 1. Because OWL only used binary relations, we ignore the temporal argument from the relations. Future research is required to identify a standard method for representing temporal arguments of these relations in OWL.

Although most relations follow an existential all-some pattern, some relations must be formalized differently. In particular the relation **integral-part-of** cannot be formalized using a standard existential pattern. A class *C* is an **integral-part-of** a class *D* if and only if *C* is a **part-of** *D* and *D* **has-part** *C*. These two statements do not directly translate into a single OWL axiom. Therefore, we performed a transformation into a single axiom which is equivalent to both axiom's holding:

```
(?X and not (part-of some ?Y)) or
(?Y and not (has-part some ?X))
             subclassOf Nothing
```

This axiom states that it is not possible (`subclassOf Nothing`) that some entity is an instance of `?X` and not the part of some `?Y`, and neither is it possible that some entity is an instance of `?Y` and has no `?X` as part. This is formally equivalent to asserting both axioms necessary for integral parthood.

The patterns we define cannot only be used to expand relations between classes into complex OWL statements, but also to convert a complex OWL ontology into a set of relations between classes. For this purpose, let *L* be the set of named classes in the signature of an OWL ontology. Then, for each pair of classes *x* and *y* in *L*, we replace `?X` with *x* and `?Y` with *y* in the OWLDEF patterns. Then, we use OWL reasoning to verify whether the resulting axiom is true in the OWL ontology. If the resulting axiom is true in the OWL ontology, the relation between the classes *x* and *y* holds and we can add this information to an ontology in the OBOF.

In our application of the OWLDEF method to the OBOF, we currently focus on binary definition patterns. However, the same method can be used to define *n*-ary relations between classes. In this case, we must introduce more variables in the Manchester OWL Syntax, i.e., variables $X_1,...,X_n$.

3 IMPLEMENTATION AND EVALUATION

We implemented the expansion and the contraction of relational patterns in two separate software libraries and applications. The first Java library is designed to convert OBOF ontologies to OWL using the OWLDEF relation patterns and the Manchester OWL API (Horridge et al. 2007). The OBOF parser from the OWL API is extended to read the `owldef` definitions for the relations from the `typedef` statement. Based on these `owldef` definitions, whenever a `term` statement is encountered that contains a relation with an `owldef` definition in the `relationship` field, we replace `?X` and `?Y` with the corresponding term names from the term definition and convert the resulting string into an OWL axiom using the inline parsing mechanism for the Manchester Syntax from the OWL API. The resulting OWL axioms are then added to the OWL ontology in addition to the axioms which are created by the original OBO parser of the OWL API.

Second, we provide an implementation to extract relational patterns from an OWL ontology. For this purpose, an OWL ontology is read using the Manchester OWL API. Based on a list of relational patterns and the list of all class names in the loaded OWL ontology, binary relations between classes are generated as OWL axioms: each class name in the signature of the OWL ontology is used to replace `?X` in the pattern and then combined with all class names to replace `?Y` in the same pattern. Consequently, all combinations of named classes are generated to fill variables in the relation patterns, leading to a list of OWL axioms.

Using the Hermit OWL reasoner (Motik et al. 2009), we attempt to prove each of these OWL axioms and keep track of those that the reasoner could infer from the axioms asserted in the ontology. As a consequence, we obtain a list of theorems that hold in the ontology. We convert these back to the OBOF by asserting the relations in the OBO ontology that were inferred using OWL reasoning.

We provide a set of OWLDEF patterns that are applicable in many OBO ontologies. In particular, we provide an OWLDEF translation of the RO and some further widely used relations. Table 1 shows some of the OWLDEF definitions we currently maintain.

To evaluate our method, we applied it to the Celltype Ontology (CL) (Bard et al. 2005). We chose the CL due to its average size (1062 classes), relative maturity and lack of formal definitions. The CL uses two relations, **is-a** and **develops-from**. The pattern for **is-a** is `?X SubClassOf: ?Y` and the pattern for develops-from is `?X SubClassOf: develops-from some ?Y`. We implement the pattern for **develops-from** using the `owldef` statement in the OBOF:

```
[Typedef]
id: develops_from
name: develops_from
owldef: ?X SubClassOf: develops-from some ?Y
```

The CL contains 1253 **is-a** and 275 **develops-from** statements, i.e., 1528 axioms that restrict CL categories using one of these two relations. We classify the generated OWL ontology using the Hermit OWL reasoner. Based on the classified OWL ontology, we attempt to prove the two patterns for each pair of named classes in the ontology. We use the Hermit reasoner to perform these inferences. Using this approach, we identify 9,497 **is-a** and 124,420 **develops-from** statements that we add to the OBOF representation of the CL.

We further evaluated our method using the Malaria Ontology which uses the **realized-by** relation, and provide the translations at our website.

4 DISCUSSION

4.1 Comparison to other approaches

There are several methods and tools available to convert ontologies in the OBOF to OWL (Moreira and Musen 2007, Mungall 2005). Some tools and methods are capable of converting OWL to OBO (Tirmizi and Miranker 2006). At least one semantics is proposed for the OBOF that uses an interpretation of OBO in OWL (Horrocks 2007). All these conversion tools and methods for OBO to OWL

Relationship	OWLDEF Pattern
part-of	?X subclassOf part-of some ?Y
has-part	?X subclassOf has-part some ?Y
integral-part-of	(?X and not (part-of some ?Y)) or (?Y and not (has-part some ?X)) subclassOf Nothing
has-integral-part	(?X and not (has-part some ?Y)) or (?Y and not (part-of some ?X)) subclassOf Nothing
proper-part-of	?X subclassOf proper-part-of some ?Y
has-proper-part	?X subclassOf has-proper-part some ?Y
located-in	?X subclassOf located-in some ?Y
location-of	?X subclassOf location-of some ?Y
contained-in	?X subclassOf contained-in some ?Y
contains	?X subclassOf contains some ?Y
adjacent-to	?X subclassOf adjacent-to some ?Y
transformation-of	?X subclassOf transformation-of some ?Y
transformed-into	?X subclassOf transformed-into some ?Y
derives-from	?X subclassOf derives-from some ?Y
derived-into	?X subclassOf derived-into some ?Y
preceded-by	?X subclassOf preceded-by some ?Y
precedes	?X subclassOf precedes some ?Y
has-participant	?X subclassOf has-participant some ?Y
participates-in	?X subclassOf participates-in some ?Y
has-agent	?X subclassOf has-agent some ?Y
agent-in	?X subclassOf agent-in some ?Y
realized-by	?X subclassOf realized-by only ?Y
realizes	?X subclassOf realizes some ?Y
lacks-part	?X subclassOf not (has-part some ?Y)
has-function	?X subclassOf has-function some ?Y
lacks-function	?X subclassOf not (has-function some ?Y)
has-function-realized-by	?X subclassOf has-function some (realized-by only ?Y)

Table 1. OWLDEF patterns for the OBO Relationship Ontology. Emphasized relations are not a part of the OBO Relationship Ontology but are included in its extensions.

have in common that they interpret a relation *R* between two classes *C* and *D* as an existential statement:

```
C SubClassOf: R some D
```

Although this pattern is appropriate for a majority of currently used relations in OBO and OBO Foundry ontologies, it fails in several cases. Table 1 lists several such cases. In particular, the **integral-part-of** and **has-integral-part** relations in the RO require a different translation to OWL. Further relations that are used in OBO ontologies include the **realized-by** relation between a function or disposition and a process. The **realized-by** relation must not be formalized using an existential pattern, as this would imply the false assertion that every function or disposition is actually realized by some process (Schulz et al. 2009). Several complex relations such as **has-function-realized-by** (Hoehndorf et al. 2010) require a more expressive translation to OWL.

To the best of our knowledge, there are no conversion tools available that are compatible with the RO in that they apply the definition patterns of the RO in the conversion. Similarily, the OWL implementation of the RO does not coincide with the definitions of the RO relations in first order logic. We are also not aware of an implementation of the RO in OWL that implements or approximates the definition patterns the RO attempts to provide.

4.2 Limitations

The OWLDEF method provides a flexible way to define relations using complex OWL statements. However, it interferes with other parts of the OBOF. In particular disjointness, intersection and union statements do not interoperate well with the OWLDEF method. To illustrate the problem, consider the following definition of a category in the OBOF:

```
[Term]
id: ID:1
intersection_of: ID:2
intersection_of: integral-part-of ID:3
```

The difficulty is that `integral-part-of ID:3` is not a class description when the OWLDEF method is used. Instead, `ID:1 integral-part-of ID:3` would translate into one OWL axiom. Axioms cannot be disjoint from classes (`ID:2`), and therefore, the meaning of these statements is not clear.

However, the current translations of the OBOF to OWL do not provide an adequate semantics for this statement either, because the relation **integral-part-of** is not and cannot be translated appropriately. One possible solution would be to disallow the use of relational statements in intersection, disjointness or union statements, and allow only class names as arguments. It is subject to future research to provide a semantics for these statements in combination with the OWLDEF method.

4.3 Performance

The conversion from OWL to OBO is implemented using a naive approach. We currently use every pair of named classes and attempt to infer the OWL statement that results from replacing `?X` and `?Y` in the definition patterns with the named classes. Consequently, if n is the number of named classes in an OWL ontology, this approach requires n^2 inferences.

In our use-case using the Celltype Ontology (Bard et al. 2005), the conversion of OWL to OBO required 264 minutes on an AMD Opteron processor with 2.3GHz and using 10GB of memory. Converting the Malaria Ontology required several days. In the future, we will attempt to reuse already performed inferences and use heuristics to speed up the process of infering the patterns. Designing a more efficient algorithm is subject to future work.

5 CONCLUSIONS

We provide a method for integrating the OBOF and the Web Ontology Language (OWL). Many ontologies in the biomedical domain are developed using the OBOF. It is derived from a graph-based language in which nodes correspond to ontological categories and edges to relations between these categories. The OBO Relationship Ontology provides definitions for the relations used in many biomedical ontologies. These definitions have so far been neglected in any attempt to integrate the OBOF and OWL.

The OWLDEF method extends the OWL Manchester Syntax to include variables for OWL classes. The assumption is that any OWL axiom in two variables for classes defines a relation between these classes. Based on this assumption and the extension to the OWL syntax, we provide a novel implementation of the OBO Relationship Ontology in OWL.

There are two main directections of application for the OWLDEF method. The first is to convert biomedical ontologies from the OBOF to OWL. This part is implemented in a novel OBO to OWL conversion software that includes the OWLDEF relation patterns. The second direction is from OWL to the OBOF. For this purpose, we implemented a prototype conversion software that makes use of the Hermit OWL reasoner to infer new relations that obtain between two categories in an ontology in the OBOF. Combining both directions of the OWLDEF method leads to an integration between the OBOF and OWL. It permits the use of the methods and software applications that were developed for OWL to infer new knowledge in biomedical ontologies. Using automated inference and expressive relations will support the development, maintainance and correctness of biomedical ontologies.

REFERENCES

Michael Ashburner et al. Gene ontology: tool for the unification of biology. the gene ontology consortium. *Nat Genet*, 25(1):25–29, May 2000.

Jonathan Bard, Seung Y. Rhee, and Michael Ashburner. An ontology for cell types. *Genome Biology*, 6(2), 2005.

Robert Hoehndorf. *Basic considerations for improving interoperability between ontology-based biological information systems*, chapter 4.1, pages 71–80. University of Leipzig, 2009.

Robert Hoehndorf, Axel-Cyrille Ngonga Ngomo, and Janet Kelso. Applying the functional abnormality ontology pattern to anatomical functions. *Journal for Biomedical Semantics*, 2010.

Matthew Horridge, Sean Bechhofer, and Olaf Noppens. Igniting the owl 1.1 touch paper: The owl api. In *Proceedings of OWLEd 2007: Third International Workshop on OWL Experiences and Directions*, 2007.

Matthew Horridge et al. The Manchester OWL Syntax. *Proc. of the 2006 OWL Experiences and Directions Workshop (OWL-ED2006)*, 2006.

Ian Horrocks. OBO Flat File Format Syntax and Semantics and Mapping to OWL Web Ontology Language. Technical report, University of Manchester, March 2007. `http://www.cs.man.ac.uk/~horrocks/obo/`.

Dilvan A. Moreira and Mark A. Musen. OBO to OWL: a protege OWL tab to read/save OBO ontologies. *Bioinformatics*, 23(14): 1868–1870, July 2007.

Boris Motik, Rob Shearer, and Ian Horrocks. Hypertableau Reasoning for Description Logics. *Journal of Artificial Intelligence Research*, 36:165–228, 2009.

Chris Mungall. Mapping obo to owl. `http://www.godatabase.org/dev/doc/mapping-obo-to-owl.html`, 2005.

Chris Mungall and John Day-Richter. The obo flat file format specification, version 1.3. `http://www.geneontology.org/GO.format.obo-1_3.shtml`, 2008.

Stefan Schulz, Holger Stenzhorn, Martin Boeker, and Barry Smith. Strengths and limitations of formal ontologies in the biomedical domain. *RECIIS – Electronic Journal in Communication, Information and Innovation in Health*, 3(1):31–45, 2009.

Barry Smith et al. Relations in biomedical ontologies. *Genome Biol*, 6(5), 2005. ISSN 1465-6914.

Barry Smith et al. The OBO Foundry: coordinated evolution of ontologies to support biomedical data integration. *Nat Biotech*, 25(11):1251–1255, 2007.

Syed H. Tirmizi and Daniel Miranker. Obo2owl: Roundtrip between obo and owl. Technical report, University of Texas, 2006. URL `ftp://ftp.cs.utexas.edu/pub/techreports/tr06-47.pdf`.