Use of OWL within the Gene Ontology

Christopher J Mungall*[1] Heiko Dietze[1] David Osumi-Sutherland[2]

Abstract. The Gene Ontology (GO) is one of the most successful and widely used ontologies in the life sciences and indeed in the history of knowledge representation. Commonly conceived of as a simple terminology structured as a directed acyclic graph, the GO is actually well-axiomatized in OWL and makes use of a large stack of OWL tools. Here we outline some of the lesser known features of the GO, describe the GO development process, and our prognosis for future development in terms of the OWL representation.

1 Introduction

The Gene Ontology (GO) is a bioinformatics resource for describing the roles genes play in the life of an organism, covering a variety of species from humans to bacteria and viruses.

The way the GO is most commonly presented in publications elides much of the underlying axiomatization and formal semantics. The most common conception is a Directed Acyclic Graph (DAG) $G =< V, E >$, where each vertex in V is a particular gene "descriptor", and E is a set of labeled edges connecting two vertices in V. The GO is not of much use to biologists in itself - the value comes in how databases use the GO to "annotate" genes and molecular entities. A database D is a pair $< A, M >$ where M is a set of molecular entities (e.g. genes or the products of genes) and A is a set of associations where each association connects an element of V with an element of M. Currently GO has some 40k vertices, nnn edges, and the combined set of databases using GO have xxx associations covering yyy genes in zzz different species[1].

The users of the GO apply it in a number of ways. The simplest way is to interrogate a database, for example to find the set of descriptors for a gene, or to find the set of genes for a descriptor (making use of the edges in E). One of the most common uses is to find a functional interpretation of a set of genes, a so-called enrichment test. For example, given a set of genes that become active as a result of an organism being exposed to an environmental toxin[REF]. Another use is as a component of a diagnostic tool for finding causative genes in rare diseases[**?**]. As of today, the GO has been cited NNN times (and this is an under-representation of the true use), with XXX tools dedicated to using the GO, integrated into MMM databases.

This simple formulation of the GO is popular, but outdated, as the GO has incorporated an ever increasing number of OWL constructs over the years. The core ontology graph G maps to *SubClassOf* axioms – either between two named classes (so called is_a links) or between a class an expression of the form R some Y. Behind the scenes there are additional axioms, some of which we will describe in this manuscript.

2 The Axiomatic Structure and Logic of the GO

The GO consists of 40k classes, but also includes an import chain that brings in an additional nnn classes from 5? additional ontologies. The majority of the axioms in this import chain are within the EL++ profile, allowing for the use of faster reasoners (axioms outset this subset are described later).

The breakdown of axiom types and expression types is as follows [TABLE]. As can be seen, the most common expression types in GO are existential restrictions and intersections, with the latter used almost entirely within equivalence axioms.

The entire ontology reasons in nn s in Elk[**?**], and nn minutes in Hermit.

3 Equivalence axioms in GO

The meat and potatoes for reasoning in GO are the equivalence axioms, most typically of a genus-differentia form, i.e. X EquivalentTo G and R1 some Y1 and Rn some Yn. These are known colloquially in GO as cross products, since the set of such defined classes X can be drawn from the cross-product of the set G and the sets Y1..Yn.

The existence of these axioms allow us to use reasoners to automatically classify the GO, something that is vitally important in an ontology with so many classes. This was previously done entirely by hand, which was time-consuming and error-prone. We aim to follow the Rector Normalization pattern[REF] as closely as possible but in practice this is hard, as we are still in the midst of refactoring a largely hardcoded structure. As this is a labor-intensive procedure, we sometimes break this down into chunks depending on the external ontology used for normalization. See for example [4] which describes refacoring and reasoning using the CHEBI ontology of chemical entities. Ongoing work involves the cell type ontology, Uberon and classifications of proteins.

These axioms can be broken down into intra-ontology and inter-ontology equivalence axioms. The inter-ontology axioms have the added benefit of connecting different ontologies used for classification of different types of data.

We use a number of different Object Properties in these axioms, largely taken from the OBO Relations Ontology [link to RO website].

4 Constraints in the GO

As well as automatic classification, there is a need for automated quality control processes. The GO has a large QC pipeline, a subset of which is implemented via standard OWL reasoning operating over the axioms in GO.

The majority of constraints in GO are encoded via disjointness axioms; many of the relations used in GO are quite general and thus domain and range constraints are less useful. We achieve more powerful contextual domain-range type assertions using disjointness axioms. For example, the 'part of' relation is flexible regarding whether is it used between processes (for example, a GO biological

process such as flower development) or between static material entities (for example, a GO subcellular component, such as synapse). This generality limits the utility of domain and range. However, the RO includes axioms of the form: 'part of' process DisjointWith 'part of' some continuant Which prohibits category-crossing uses of 'part of' which would be invalid.

Disjointness axioms are also used in the traditional way, between siblings in a taxonomic classification.

We frequently have need to encode spatial and temporospatial constraints. For example, most of the cells in a complex organism such as yourself consist of a number of compartments, including the nucleus (the central HQ, where most of your genes live) and the cytosol (a kind of soup full of molecular machines doing their business). Its not enough to simply state that the cell and cytosol are disjoint classes. We also want to encode what in RCC8 terms is the disconnected relation, the fact that no parts are shared in common (made impossible by the existence of a membrane barrier between the two). We do this using General Class Inclusion axioms (GCI axioms), e.g. ('part of' cytosol) DisjointWith ('part of' nucleus)

Constraints are also used to check the contents of databases. For example..

Another common type of contsraint in the GO are so-called 'taxon constraints' [3]. The basic idea here is that the GO covers biology for all domains of life, from single-celled organisms to humans. However, many of the classes are applicable to specific lineages. For example, in describing the function of genes in Dictyostelium discoideum (slime mold), it would be a mistake to use the GO class brain development, as these ameoboid organisms lack a nervous system of any type. Whilst a human curator would be unlikely to make such a gross error, the same cannot be said for algorithmic prediction methods that make use of the 'ortholog conjecture' to infer the function of a gene in one species based on the function of the equivalent gene in another species. Here it is useful to have a knowledge-based approach to validation.

5 Smuggling OWL expressions into Databases

[5] Logic is not the only fruit: Annotations in the GO

Say something about how its not just logic thats important to us in GO. Highlight axiom annotations. Mention the terminological confusion over annotation.

6 The GO Development environment

Protege and OBO-Edit[2] dual editing Protege plugins[http://wiki.geneontology.org/index.php/Ontology_editor TermGenie[**?**] OWLTools and Oort Continuous Integration[6] VCSs and continued use of OBO

Discussion of obstacles, what we need, when do we want it. Inferring subclass of 'part of' some Y. More OE like experience in Protege. More tooling. Something like Maven for ontologies, especially with dependency/version management.

Currently we have a requirement that ontology developers can switch between use of obo-edit and Protege. Unfortunately, the reasoning support in obo-edit is poor (not integrated with Elk or any standard OWL reasoner). This means that there is effectively no inferred class view, so we cannot rely on dynamic classification if developers wish to see a rich and complete hierarchy. As a workaround, we have a batch process that mass asserts all direct inferred subclass axioms (and tags these as being asserted via an axiom annotation). In theory these can be deleted and recreated en-masse. In practice this is frankly a rube goldbergesque system that has evolved ad-hoc over the years, and we are planning to overhaul this.

Mention that GO development is closely coordinated with others like CL, UBERON, both in terms of ontology integration and shared infrastructure

Limitations of MIREOT. We have to mention OBO Foundry efforts here even though in many ways its ceased to be of relevance...

Downstream use of the GO. More OWLishness required!

The detailed OWL axiomatization of GO is primarily used within the GO consortium, as a tool for automation and QC during the ontology development lifecycle, and as a database QC tool. However, these more advanced axioms are still not widely used by the many groups developing software the embed or leverage the GO.

Discussion on why this is and whats to be done. better integration of statistical/probablistic views and logical.

7 What does the future hold?

GO was designed to be used with a simple database structure, pairwise associations between nodes in the GO graph and molecular entities.

Where were going now: describing complex interactions. How should this be done? ABox or TBox or some mix?

FIGURE: Noctua

8 Conclusions

OWL is great but we want better support, easier tools, .

A small amount of constructs go a long way: EquivalentClasses, SomeValuesFrom, DisjointWith, SubPropertyChainOf Elk was a game-changer In many aspects GO axiomatization is of a similar expressivity to earlier DL ontologies like GRAIL. Lessons? Takes a long time for tools to harden, specs to mature and technology to percolate Foundations of OWL2 are solid (specifications, core APIs) but we need more tools that can be used as part of the ontology development lifecycle. Examples are things like owlet. How can we get OWL axioms to be used more downstream of the ontology development lifecycle? Should they be? Is the future in integration between DLs and the kinds of statistics and probabilistic reasoning more common in bioinformatics

9 Conclusions

YADDA

References

1. JA Blake, M Dolan, H Drabkin, DP Hill, L Ni, D Sitnikov, S Bridges, S Burgess, T Buza, F McCarthy, D Peddinti, L Pillai, S Carbon, H Dietze, A Ireland, SE Lewis, **Mungall, C.J.**, et al. Gene ontology annotations and resources. *Nucleic Acids Research*, 41(D1):D530–D535, 2013.
2. John Day-Richter, Midori A Harris, Melissa Haendel, Gene Ontology OBO-Edit Working Group, and Suzanna Lewis. OBO-Edit–an ontology editor for biologists. *Bioinformatics*, 23(16):2198–2200, August 2007.
3. J Deegan, E Dimmer, and **Mungall, C. J.** Formalization of taxon-based constraints to detect inconsistencies in annotation and ontology development. *BMC bioinformatics*, 11(1):530, 2010.
4. David P Hill, Nico Adams, Mike Bada, Colin Batchelor, Tanya Z Berardini, Heiko Dietze, Harold J Drabkin, Marcus Ennis, Rebecca E Foulger, Midori A Harris, Janna Hastings, Namrata S Kale, Paula de Matos, **Mungall, C. J.**, Gareth Owen, Paola Roncaglia, Christoph Steinbeck, Steve Turner, and Jane Lomax. Dovetailing biology and chemistry: integrating the Gene Ontology with the ChEBI chemical ontology. *BMC genomics*, 14(1):513, January 2013.
5. Rachael P Huntley, Midori A Harris, Yasmin Alam-Faruque, Judith A Blake, Seth Carbon, Heiko Dietze, Emily C Dimmer, Rebecca E Foulger, David P Hill, Varsha K Khodiyar, Antonia Lock, Jane Lomax, Ruth C Lovering, Prudence Mutowo-Meullenet, Tony Sawford, Kimberly Van Auken, Valerie Wood, and **Mungall, C. J.** A method for increasing expressivity of Gene Ontology annotations using a compositional approach. *BMC Bioinformatics*, 15(1):155, 2014.
6. **Mungall, C. J.**, Heiko Dietze, Seth J Carbon, Amelia Ireland, Sebastian Bauer, and Suzanna Lewis. Continuous Integration of Open Biological Ontology Libraries. 2012.