Exploring Gene Ontology Annotations with OWL

Simon Jupp[1], Robert Stevens[1] and Robert Hoehndorf[2]*

[1]School of Computer Science, University of Manchester, UK. [2] Department of Genetics, University of Cambridge, Cambridge, UK

ABSTRACT

Motivation: Ontologies such as the Gene Ontology (GO) and their use in annotations make cross species comparisons of genes possible, along with a wide range of other activities. Tools, such as AmiGO, allow exploration of genes based on their GO annotations. This human driven exploration and querying of GO is obviously useful, but by taking advantage of the ontological representation we can use these annotations to create a rich polyhierarchy of proteins for enhanced querying. This also opens up possibilities for exploring GOA for redundancies and defects in annotations.

To do this we have created a set of OWL classes for mouse GOA genes. Each gene is represented as a class, with the appropriate relationships to the GO aspects with which it has been annotated. We then use defined classes to query these protein classes and to build a complex hierarchy. This standard use of OWL affords a rich interaction with GO annotations to give a fine partitioning of the proteins in the ontology.

1 INTRODUCTION

The creation of the Gene Ontology (GO) (Harris 2004) has had a major impact on the description and communication of the major functionalities of gene products for many species. GO has some 24,000 terms for annotating gene products and is used in around 40 species databases and in cross species databases such as Uniprot and Interpro (Camon 2004). It is widely used for querying such databases, making cross species comparison or in data analyses, such as over-expression analysis in microarray data (Baehrecke 2004).

The GO is mainly used as a controlled vocabulary to ensure genes are consistently annotated using standard terminology across many data resources; this alone offers many benefits for data integration and analysis. GO is, however, much more than a vocabulary; it also provides additional information about how these GO terms are related to each other. These relationships have a well-defined semantics that bring added value to the GO. For example, the hierarchical relationships allow for all kinds of a particular term to be retrieved, as well as those with an annotation of the term itself. These and other relationships provide support for navigation, as well as making explicit the relationship between the entities being described.

The AmiGO browser (Carbon 2009) (see also DynGO (Liu 2005), QuickGO (Binns 2009)) provides such an interface and exploits the hierarchical structure of the gene ontology to support query expansion. For example, when searching AmiGO for receptor activity genes, the results returned also include genes involved in GPCR activity because GPCR activity is a subclass of receptor activity. This hierarchical structure is also useful for data mining tasks (Pavlidis 2004). Enrichment analysis is a common technique used in the analysis of high-throughput gene expression data; sets of interesting genes can be grouped or clustered based on common GO annotations

(See http://www.geneontology.org/GO.tools.shtml for more GO tools).

Whilst highly useful, many of these tools fail to exploit the full potential of the GO's representation for reasoning and querying over gene annotations. Most of the tools that were investigated do not facilitate rich querying that takes into account the semantics of the GO. For example, it was difficult to ask for all proteins that are located in a membrane or part of a membrane, that are receptor proteins involved in a metabolic process. To answer such a query correctly some form of reasoning over the ontology is required. The ability to perform such rich queries would enable more precise and flexible exploration of the GO annotations.

The Web Ontology Language (OWL)[1] and the Open Biomedical Ontology (OBO)[2] format have a strict semantics that makes it possible to use automated reasoners to help build and use knowledge captured in an ontology. In order to explore the potential of reasoning over the GO annotations we need to describe the relationships between the genes and their annotation within a framework that can also exploit the semantics encoded into the GO. Our approach uses the Web Ontology Language, for which a mapping from OBO exists, to represent both the GO annotations alongside the GO to exploit the GO and its annotation for querying and exploration.

As an ontology of attributes of gene products, GO itself does not explicitly contain gene products; GO annotations

* To whom correspondence should be addressed.

[1] http://www.w3.org/TR/owl-ref/

[2] http://obofoundry.org/

are attached to gene products in databases or flat-files (See http://www.geneontology.org/GO.annotation.shtml). Using the compositional approach to ontology building we can create an ontology from these annotations that explicitly relates gene products to GO and then add defined classes to impose a hierarchy on the gene products. For example, we can create a defined class (in Manchester OWL syntax) such as:

```
Class: NuclearMembraneReceptorGeneProduct
EquivalentTo:
GeneProduct
that has_molecular_function some ReceptorActivity
and located_in some NuclearMembrane
```

This defined class will recognize any class of gene product that has both of these attributes, or children of these attributes, and subsume it within the hierarchy of gene products. In this standard use of OWL and automated reasoning, we can add more of such defined classes to build an arbitrarily complex polyhierarchy for querying and navigation of entities annotated with the GO. Figure 1 shows such an inferred polyhierarchy centered on annotations for the GRM1[MGI:1351338] gene product.

2 METHOD

An initial set of GO annotations for mouse genes were downloaded from the Mouse Genome Informatics (MGI) site[3]. In order to reduce the size of the dataset to ease development we only selected annotations that had evidence codes of EXP, IDA, TAS, RCA, IC (See http://www.geneontology.org/GO.evidence.shtml for definitions). We also further filtered these genes to exclude the RIKEN cDNA genes. In order to express these annotation ontologically we created an OWL class for each of the genes. We then describe each gene according to its annotation using existential OWL restrictions. From this a simple pattern emerges where each gene class is restricted by the corresponding GO term from the annotation.

```
Class: GeneProduct
SubclassOf:
GeneProduct that participates_in only biological_process
and located_in only cellular_component
and has_molecular_function only molecular_function
```

[3] http://www.informatics.jax.org - accessed Nov 20, 2011.

Rather than generate the axioms by hand we use the OPPL language to specify and instantiate the pattern (Iannone 2009). OPPL allows us to express patterns for each of the three branches of GO. A Java program is then used to parse the go annotations file downloaded from MGI and instantiate the OPPL and generate the OWL ontology.

The generated GO association ontology is then manually edited using Protégé 4.1 (beta, build 220). We initially created classes to represent subsets of the top level GO terms by defining OWL classes for genes found in a particular cellular compartment. For example, we can create the class of mitochondrial gene products as follows:

```
Class: MitochondrialGeneProduct
EquivalentTo:
GeneProduct
that located_in some (GO:'mitochiondria' or
                      (part_of some GO:'mitochondria'))
```

We repeat this basic pattern for the top level cellular compartments, and then continue for the biological processes and molecular function classes. From these base level class descriptions we can then begin to create more complex class descriptions composed of classes previously created. We can now create a class to query for the mitochondrial receptor gene products with the following class definition:

```
Class: MitochondrialReceptorGeneProduct
EquivalentTo:
GeneProduct
and MitochondrialGeneProduct
and has_molecular_function some GO:'receptor activity'
```

This pattern continues until we begin to create classes that are composed of terms from all three branches of the gene ontology. For example, to get the mitochondrial proteins that are receptor proteins and participate in cell killing we can generate the following OWL class:

```
Class: CellKillingMitochondrialReceptorGeneProduct
EquivalentTo:
GeneProduct
and MitochondrialReceptorGeneProduct
and participates_in some GO:'cell killing'
```

We can continue in this vein creating an arbitrary number of defined classes, each of which will subsume and be subsumbed by other classes fitting the definition in the growing ontology. At the leaves of this polyhierarchy we have the classes representing the gene products themselves.

3 RESULTS

We extracted all mouse genes from the MGI database and applied our filtering, producing a total of 29,559 gene-annotation pairs. On conversion to OWL classes this represents 10,104 individual genes. After importing GO, the final ontology of primitive protein classes and the GO contains 39,332 OWL classes.

We created a further 120 defined classes describing various gene categories. As an exemplar, we concentrated on genes with receptor activity, located in some membrane and with processes involved in cell growth, metabolism and signal transduction.

In order to classify the ontology we used several DL reasoners. Classification was performed on a 2.2ghz i7 Mac Book Pro requiring around 3GB of memory. Table 1 shows the performance times for each reasoner.

Reasoner	Version	Average Timing (Seconds)
Fact++	1.52	~ 400
Pellet	2.1.2	~ 300
HermiT	1.3.3	~ 500

To illustrate the querying capabilities of the generated ontology we show a query to get the genes that are located in the nuclear membrane of the cell, that participate in some metabolic process and have the function of some receptor activity. Figure 1 shows a screen shot from Protégé of a define class named *MetabolicNuclearMembraneReceptorGeneProduct*. This class is composed of the intersection of three other defined classes named *NuclearMembraneProtein*, *ReceptorActivityProtein*, and *MetabolicProcessProtein*. These classes are defined in OWL as the following:

Class: ReceptorActivity
EquivalentTo: Gene
that *has_molecular_function* some GO:' receptor activity'

Class: MetabolicProcess
EquivalentTo: Gene
that *participates_in* some GO:'metabolic process'

Class: NuclearMembraneGeneProduct
EquivalentTo: Gene
that (*located_in* some (GO:'nucleus' or (*part_of* some GO:'nucleus')) and (*located_in* some GO:'membrane' or (*part_of* some GO:'membrane'))

Class: MetabolicNuclearMembraneReceptorGeneProduct
MetabolicProccess and ReceptorActivity and
NuclearMembraneGeneProduct

After reasoning over the ontology we infer that only the Grm1 gene is a subclass of our MetabolicNuclearMembraneReceptorGeneProduct class. Although this is a relatively simple query, in order for it to answer some reasoning is required, which is made possible by this approach of using OWL. Our attempts to replicate such a query in the popular online tools for querying GOA using a simple conjunction of these terms yielded no results, showing a clear advantage to the OWL approach over existing tools.

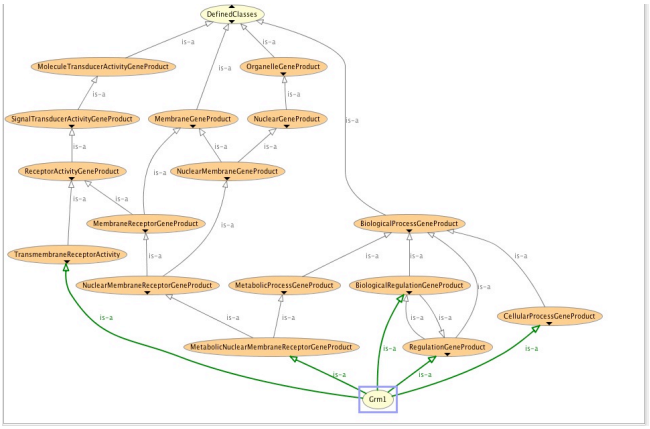

Figure 1. Showing the classification of the GRM1 gene according to generated defined classes for gene products

4 DISCUSSION

Although the queries demonstrated here are relatively simple, they serve to illustrate the potential of a pure OWL approach to querying GOA. Using similar patterns we can begin to imagine more complex class description that utilise additional expressivity in OWL, such as the use of complement classes to query for genes that '*has_molecular_function some not (ReceptorActivity) and participates_in some SignalTransduction'*, which would find those genes that have a function other than receptor activity and are involved in signal transduction. (Note that the semantics mean that such genes can have a receptor activity, but must have an activity other than receptor activity. GO annotations are not closed, so we cannot say *'not (has_molecular_fucntion some ReceptorActivity)'* and expect to recognize any genes.)

The announcement of the GO cross products extension to the GO[4] will provide logical definitions for the GO classes. These definitions will enable richer OWL queries over the GO annotations and the potential to infer more annotations on existing GOA genes (Fernández-Breis 2010).

The next stage of development will be to incorporate more defined classes and different ontologies such as the phenotype annotations for mouse genes and descriptions of cells in which they are known to function. This will enable queries such as those genes that are known to participate in processes that are involved in a particular phenotype.

Our current exploratory implementation performs reasonably well, but the number of defined classes is currently small. Adding further semantics into the ontology will afford further opportunities; adding disjointness axioms to GO may help us uncover mis-annotations and we have yet to fully exploit property characteristics such as transitivity and functionality. We can also explore ways of flexibly incorporating annotations with differing degrees of confidence through use of the GO evidence codes and programmatically generating the defined classes that form the polyheirarchy of genes. Finally, we need to present the ontology via tools such as the OWLBrowser[5] .

In this work we have made a straight-forward use of OWL and automated reasoning to deliver a flexible way to query all aspects of GO annotations. The polyhierarchy formed also provides similarly rich navigation in a gene product orientated setting. Finally, we provide a flexible framework for exploring and manipulating GO and other valuable annotations developed by the community.

AVAILABILITY

The ontologies and associated files are available to download from http://owl.cs.manchester.ac.uk/mouse_goa/index.html. We recommend Protégé 4.1 beta for viewing the generated ontology.

ACKNOWLEDGEMENTS

This work was funded by the e-LICO project---
EU/FP7/ICT-2007.4.4.

REFERENCES

Harris MA et al (2004). *The Gene Ontology (GO) database and informatics resource*. Nucleic Acids Res. Jan 1;**32**(DATABASE):D258–D261

Evelyn Camon and Rolf Apweiler et al. *The Gene Ontology Annotation (GOA) Database: sharing knowledge in Uniprot with Gene Ontology* Nucl. Acids Res. (2004) 32(suppl 1): D262-D266 doi:10.1093/nar/gkh021

Eric H Baehrecke, Niem Dang, Ketan Babaria and Ben Shneiderman. *Visualization and analysis of microarray and gene ontology data with treemaps* BMC Bioinformatics 2004, 5:84doi:10.1186/1471-2105-5-84

Seth Carbon, Amelia Ireland, Christopher J. Mungall, ShengQiang Shu, Brad Marshall, Suzanna Lewis, the AmiGO Hub, and the Web Presence Working Group. *AmiGO: online access to ontology and annotation data.* Bioinformatics. 2009 January 15; 25(2): 288–289.

Liu H, Hu ZZ, Wu CH. *DynGO: a tool for visualizing and mining of Gene Ontology and its associations*. BMC Bioinformatics. 2005 Aug 9;6:201.

Binns D, Dimmer E, Huntley R, Barrell D, O'Donovan C, Apweiler R. *QuickGO: a web-based tool for Gene Ontology searching*. Bioinformatics. 2009 Nov 15;25(22):3045-6.

Pavlidis P, Qin J, Arango V, Mann JJ, Sibille E. *Using the gene ontology for microarray data mining: a comparison of methods and application to age effects in human prefrontal cortex.* Neurochem Res. 2004 Jun;29(6):1213-22.

Luigi Iannone, Alan L. Rector, Robert Stevens: *Embedding Knowledge Patterns into OWL*. ESWC 2009: 218-232

Jesualdo Tomás Fernández-Breis, Luigi Iannone, Ignazio Palmisano, Alan L. Rector, Robert Stevens. Enriching the Gene Ontology via the Dissection of Labels Using the Ontology Preprocessor Language. In Proceedings of EKAW'2010. pp.59~73

[4] http://wiki.geneontology.org/index.php/Category:Cross_Products

[5] http://code.google.com/p/ontology-browser/