yOWL: An ontology-driven knowledge base for yeast biologists

Natalia Villanueva-Rosales\textsuperscript{a}, Michel Dumontier\textsuperscript{a,b,*}

\textsuperscript{a}School of Computer Science, Carleton University, Ottawa, Ont., Canada K1S 5B6
\textsuperscript{b}Department of Biology, Carleton University, Ottawa, Ont., Canada K1S 5B6

\textbf{A R T I C L E   I N F O}

\textbf{Article history:}
Received 1 September 2007
Available online 11 May 2008

\textbf{Keywords:}
Semantic web
Data integration
Semantic query answering
Knowledge management
OWL
Ontology
Biological data
Yeast
Saccharomyces cerevisiae
Mashup

\textbf{A B S T R A C T}

Knowledge management is an ongoing challenge for the biological community such that large, diverse and continuously growing information requires more sophisticated methods to store, integrate and query their knowledge. The semantic web initiative provides a new knowledge engineering framework to represent, share and discover information. In this paper, we describe our efforts towards the development of an ontology-based knowledge base, including aspects from ontology design and population using “semantic” data mashup, to automated reasoning and semantic query answering. Based on yeast data obtained from the Saccharomyces Genome Database and UniProt, we discuss the challenges encountered during the building of the knowledge base and how they were overcome.

© 2008 Elsevier Inc. All rights reserved.

\textbf{1. Introduction}

Online biological information is available via web pages, stored in ftp sites or relational databases, and textually described in publications. However, web search engines are unable to answer questions about this massive knowledge base other than identifying resources that contain some subset of the specified attributes. The main reason for this limitation is that the \textit{representation} of biological information on the web is \textit{not} \textit{machine understandable}, in the sense that computers cannot interpret words, sentences or diagrams so as to correctly \textit{reason} about the objects and the relations between them that are implicitly stated in those documents [1]. The primary goal of the \textit{semantic web} is to add \textit{semantics} to the current Web, by designing \textit{ontologies} which explicitly describe and relate objects using formal, logic-based representations that a machine can understand and process [2,3]. This ongoing effort is expected to facilitate data representation, integration and question answering, of critical importance in the life sciences.

Ontologies already play an important role in managing medical terminology [4–6], and more recently in the discovery and execution of grid [7] and semantic web services [8]. The Open Biomedical Ontologies (OBO) is a shared portal of biological/medical ontologies that includes the popular Gene Ontology (GO) [9]. By providing a standardized vocabulary, OBO controlled vocabularies and taxonomies are used in the annotation of biological information, which helps make \textit{information} more accessible for computer interpretation. Through the OBO Foundry effort, OBO ontologies are being redesigned and mapped to the Basic Formal Ontology (BFO) [10], a domain independent ontology that provides distinction between objects and processes and can be linked using basic relations [11]. Together, they should provide a powerful platform to describe and annotate domain specific knowledge, and open the possibility of making queries at various levels of ontological granularity and potentially from diverse domains. This in turn, will enable knowledge discovery by tapping into expert knowledge. For instance, a biochemical ontology might help one discover enzymes as the set of proteins that catalyze reactions, even though these have not been asserted as enzymes. While the first generation of OBO ontologies cannot be used in this way because they do not contain explicit logical descriptions to define class membership in terms of their properties, work towards a second generation of OBO ontologies using formal, logic-based knowledge representation languages will provide enhanced functionality toward this goal [12–14].

\textit{OWL}, the Web Ontology Language [15], is a knowledge representation language for designing ontologies on the Semantic Web. One variant, OWL-DL, is based on description logics (DL), a subset of First Order Logic that allows description of complex concepts from simpler ones with an emphasis on decidability of reasoning tasks [16] (i.e. the results will be returned in a finite
amount of time). Reasoning tasks like checking ontology consistencies, computing inferences, and realization (classifying real world objects into their most specific category) can be executed by a computer program called a reasoner (e.g. Pellet [17], FACT++ [18] and Racer Pro [19]) over DL ontologies [20]. The design of OWL-DL bio-ontologies favorable to reasoning may be achieved by the application of semantic web best practices [21], relation formalisms [11], normalization [22], design patterns and workflows [23]. Sophisticated biomedical ontologies such as the Foundational Model of Anatomy are being converted to OWL and this has proven useful in simplifying the ontology and identifying inconsistencies [24,25]. The FungalWeb project involved the design of an ontology to reason about enzymes of importance to the yeast biotechnology industry [26]. The BioPax OWL ontology [27] enables data sharing via an ontology for pathways, interactions and molecular participants, for data exchange between pathway data providers such as HumanCyc [28] and Reactome [29].

A considerable challenge in bioinformatics is the resolution of multiple identifiers for the same entities. The proliferation of identifiers stems from (1) direct user submissions to a general database (i.e. NCBI), (2) the construction of "boutique" databases (i.e. SGD) and (3) value added annotations fuels a need for each provider to issue new identifiers so as to keep track of their contributions [30–32]. However, keeping track of these identifiers is such a problem that it becomes necessary to create databases of database identifiers [33,34]. In fact, identifiers have such a pervasive influence in the life sciences that people talk about identifiers instead of the entities they are meant to identify. While LinkHub [35] offers a first step at navigating this confusing network of identifiers, YeastHub [36] provides an RDF-based data warehouse which lets one add data and create queries between resources. While flexible, the lack of ontology requires users to indicate which user-contributed resources are equivalent, a difficult task, and this precludes the automatic knowledge discovery.

In this paper, we create a semantic knowledge by designing and populating expressive ontologies primarily using the information from the Saccharomyces Genomes Database (SGD) database2 such that one could effectively search this knowledge base using expected relations rather than with a collection of keywords. One example would be to identify a gene involved in a genetic interaction experiment and whose gene product is involved in transcription (Query 8). SGD is a free and widely used resource for yeast genomic and proteomic information, from which structural and functional chromosome features (telomeres, genes, etc), database cross references, molecular function, cellular component, biological process, interactions, pathways, phenotypes and literature references as listed in Table 1.

2.1.2. Step 2. Determine essential concepts, reusing existing ontologies when possible

An initial set of concepts was manually collected to represent the type and attributes of the data located in the data files. Concept definitions were obtained from the SGD glossary, WordNet and Wikipedia and were manually added to the class using the "rdfs:comment" annotation property for human consumption.

To illustrate the conceptual process, we will illustrate the methodology for the "interactions" file. This file describes results from physical or genetic interaction experiments and each row contains columns for the pair of interacting genes/proteins, the type of interaction experiment, the impact on viability and a reference to a publication in which it was reported. To represent this information in the ontology, the following concepts were created: (a) the class of InteractionExperiment and more specific classes from an enumeration of types of interaction experiments (e.g. SyntheticLealthyExperiment, AffinityCapture-MS, etc). Values captured in the "viability" column were captured as viability states (viable, non-viable) or growth states (slow growth, etc). References to journal articles identified by PubMed identifier are captured by a Publication class.

Data found in the "go.slim_mapping" file corresponds to information defined by the Gene Ontology. However, no OWL document was available for this subset of GO. Thus, we constructed an OWL ontology3 with concepts that are equivalent (owl:EquivalentClass)

---

1 http://www.co-ode.org/downloads/protege-x/.
2 http://www.yeastgenome.org/.
to those in the complete OWL\(^4\) GO ontology, but with English class names like “Nucleus” rather than alphanumeric class names favored by OBO ontologies (i.e. GO\(_{0005634}\)).

2.1.3. Step 3. Construct a primitive hierarchy of concepts

A hierarchy of concepts for the yOWL primitive ontology\(^5\) was developed by iteratively categorizing the set of necessary classes. For instance, a SyntheticLethalityExperiment is a type of SyntheticGeneticExperiment which is a type of GeneticInteractionExperiment which is a type of Experiment. The result is the design of an taxonomic branch that is homogeneous and increasingly specialized in that each child term can be easily differentiated from its parent. In line with general normalization techniques, all ontological terms are asserted to have but a single parent.

2.1.4. Step 4. Map to an upper level ontology

An upper level ontology should be carefully selected according to the purpose and scope of the ontology designed. The Basic Formal Ontology (BFO) provides a simplified framework to distinguish from qualities, objects and processes. We mapped classes defined in the yOWL ontology to the BFO ontology. For example, the validation status of open reading frame is a type of quality, a Publication is a type of object and an Experiment is a type of Process. In doing so, we anticipate that our classes (and the restrictions placed on them) will be compatible with second generation OBO ontologies that also adopt the BFO framework.

2.1.5. Step 5. Assign relations between concepts and attributes

An essential part of the modeling process is to establish which relations exist between the set of concepts. To do this, we need two things: (1) a set of basic relations to draw from and (2) generate a kind of topics map. We have designed the Basic Relation Ontology (BRO)\(^6\) to hierarchically organize basic object relations for use in OWL ontologies. A root relation isRelatedTo provides a generic relationship between any entity and also provides object-process, object-quality, mereological, spatial and temporal relations. NULO,\(^7\) an integrated upper level ontology, maps the domain and ranges of BRO relations with BFO concepts so as to semantically constrain the assignment of relations and provide enhanced reasoning and inference capabilities. Taken together, NULO consists of 36 classes, 50 object properties and 2 datatype properties, 17 annotation properties. Similarly to the use of upper level ontologies, the use of NULO in several ontologies imposes extra restrictions that generating further inferences may facilitate semantic integration in an abstract level.

New object properties were added to describe the more specific relations required (but not restricted to) in this domain. The first pair hasReference/isReferencedIn provides a relation between a publication and the entity it references. The second pair hasSource/isSourceOf, links an entity with its source of origin, and provides a means to assign data provenance. A pair of quality relations hasSta-

---

\(^4\) http://www.co-ode.org/downloads/protege-x/.


\(^6\) http://ontology.dumontierlab.com/bro.

\(^7\) http://ontology.dumontierlab.com/nulo.
**Fig. 2.** Selected classes and relations from yOWL ontology.

2.2. Ontology Population and Mashup

In this work, the ontology population involved: (1) assigning names, (2) asserting class membership, and (3) determining proper relations between entities. In the assignment of unique names, we constructed Universal Resource Identifiers (URI) from an assigned base namespace (Table 2) plus the identifier. In the assignment of types, the yOWL ontology sufficiently covers the genomic and proteomic data obtained from SGD tab files. However, given that there was no relational database schema available, the specific relations between the data had to be manually determined based on the concept and relations extraction results of steps 2 and 4.

We designed PHP-based parsers to automatically populate (instantiate classes and relations) the yOWL ontology from SGD tab files, obtaining an OWL document in RDF/XML format. A necessary first step was to normalize the content such that multi-valued column entries delineated by “/” or “|” delimiters were separated and treated as distinct entities. Another problem was that these files lacked references to the primary key, the SGD identifier (SGID: a numeric identifier prefixed with an ‘S’—chromosomal features file). Instead some contained references to gene names or ORF names, and we found additional references to resources outside of the SGD. In particular, information about genes and their open reading frames could be obtained from UniProt records in RDF⁹ RDF documents were fetched for each entity for which information was available. Local identifiers for blank nodes in these RDF documents were replaced with globally identifiers by appending the name of the entity, thereby avoiding the incorrect integration of triples. GO URIs were also mapped to the GOSLIM ontology. The curated RDF document¹⁰ can be seamlessly imported and integrated with the yOWL populated ontology,¹¹ providing a mashup with additional information such as type (UniProt schema), biological source, and GO terms, among others.

Using an Intel Pentium 4 computer with 3GB RAM, we were able to load the entire data-instantiated ontology using Protégé 4 alpha (build 53) and RacerPorter 1.9.1, but unable to execute realization for query answering. Thus, due to limited resources available at the time of writing this paper, the query examples described in this paper were selected from a representative subset of the complete yOWL data. This subset was obtained by performing a search for genes with links across the ontology and contains approximately 2% of the whole dataset (5770 instances). While

---

*Figures and tables are represented as described in the image, with appropriate labels and references to ontologies and namespaces.*

---

⁸ [http://ontology.dumontierlab.com/yowl-complex](http://ontology.dumontierlab.com/yowl-complex)

⁹ [http://yeastgenome.org/](http://yeastgenome.org/)

¹⁰ [http://ontology.dumontierlab.com/yowl-jbi-unitprot.owl](http://ontology.dumontierlab.com/yowl-jbi-unitprot.owl)

¹¹ [http://ontology.dumontierlab.com/yowl-jbi-owl](http://ontology.dumontierlab.com/yowl-jbi-owl)
smaller, this approach still enabled a demonstrated of proof of principle on how OWL-DL ontologies can be used to enable semantic query answering over data, rather than undertake performance testing on available tools.

2.3. Question answering

The design of reasoning capable applications can facilitate information retrieval as well as aid in the discovery of new knowledge about a subject of interest. We demonstrate, by means of examples, how a scientist could retrieve information from yOWL such that they can query at various levels of ontological granularity, query across data having multiple synonymous identifiers, and make semantically constrained queries. We focus on two main categories of queries: class queries and graph pattern based queries.

Class queries are constrained to the concepts, properties, and individuals contained in the ontology. While class queries have been around for a long time, the Manchester OWL syntax [42] directly maps to the OWL knowledge representation language and can be easily formulated using the Protégé 4 DL Query plug-in and the Pellet/Fact++ reasoners that are embedded in this application.

Graph pattern queries are more powerful than class queries in that they allow multiple variables to be specified and restricted, and hence the query returns a sub-graph of the underlying knowledge base. For example, we might ask to retrieve genes/gene products, pathway and source where the gene is involved in a protein modification pathway. These type of queries can be formulated using the nRQL query language, a lisp based query language [19] supported in Racer Pro 1.9.1. Intuitively, these queries are based on graph patterns composed by nodes and edges. Each node represents a class, an individual or a variable to be bound to an individual. Edges represent relations through properties or restrictions in the ontology. For the answering of nRQL queries we used Racer Pro (via the Racer Porter interface) with the instance indexing feature enabled for better performance and unique name assumption.

3. Results

3.1. Heterogeneous biological data integration

3.1.1. Resource integration and provenance

SGD assigns a unique identifier (SGCID) for every chromosomal feature it provides. All other identifiers including gene names, gene aliases, ORF names, and all database cross-references were assigned one of the namespaces in Table 2 and made an instance of owl:-

<table>
<thead>
<tr>
<th>Source</th>
<th>URI</th>
</tr>
</thead>
<tbody>
<tr>
<td>yOWL ontology</td>
<td><a href="http://ontology.dumontierlab.com/yowlprimitive">http://ontology.dumontierlab.com/yowlprimitive</a></td>
</tr>
<tr>
<td>GOslim ontology</td>
<td><a href="http://ontology.dumontierlab.com/goslim">http://ontology.dumontierlab.com/goslim</a></td>
</tr>
<tr>
<td>GO</td>
<td><a href="http://purl.org/obo/owl">http://purl.org/obo/owl</a> GO</td>
</tr>
<tr>
<td>SGD*</td>
<td>urn:lsid:yeastgenome.org:SGD</td>
</tr>
<tr>
<td>Genbank*</td>
<td>urn:lsid:ncbi.nlm.nih.gov:genbank:</td>
</tr>
<tr>
<td>EBI*</td>
<td>urn:lsid:ebi.ac.uk:</td>
</tr>
<tr>
<td>DIP*</td>
<td>urn:lsid:dip.doe-mbi.ucla.edu:</td>
</tr>
<tr>
<td>SGD*</td>
<td>urn:lsid:candidagénome:SGD</td>
</tr>
<tr>
<td>CandidaDB*</td>
<td>urn:lsid:candidadb:</td>
</tr>
<tr>
<td>URMB*</td>
<td>urn:lsid:uimb:reaction:</td>
</tr>
<tr>
<td>EUROSCARF*</td>
<td>urn:lsid:euroscarf:</td>
</tr>
<tr>
<td>BioGrid*</td>
<td>urn:lsid:thebiogrid:org:</td>
</tr>
<tr>
<td>MetaCyc*</td>
<td>urn:lsid:metacyc:</td>
</tr>
<tr>
<td>GermOnline*</td>
<td>urn:lsid:germonline:org:</td>
</tr>
<tr>
<td>UniProt</td>
<td><a href="http://purl.uniprot.org/uniprot/">http://purl.uniprot.org/uniprot/</a></td>
</tr>
<tr>
<td>UniParc</td>
<td><a href="http://purl.uniprot.org/uniprot/">http://purl.uniprot.org/uniprot/</a></td>
</tr>
</tbody>
</table>

* LSID assigned in the absence of known authority.

3.2. Question answering

3.2.1. Types of queries posed to yOWL

Table 3 lists example queries (class and graph pattern based) and the OWL features they use in order to demonstrate the basic functionality and advantages of OWL ontology-driven queries.

3.3. Query results

3.3.1. Query 1. Find all individuals that have a molecular function

Class Expression: hasFunction some Molecular_function

This query returns the set of individuals that have some (at least one) known molecular function. While molecular functions assigned to genes/proteins are generally more specific, these are resolve to this parent term defined in the GO slim ontology. Among the results, we find that the Elongin A, F-box protein encoded by the open reading frame identified by the SGD identifier S000005174 has been associated with an instance of transcription regulator activity.

Additional functional annotation can be obtained from the UniProt RDF knowledge base. We imported yeast specific UniProt RDF records into our yOWL knowledge base. In doing so, we discover new information beyond the original SGD data, such that YDR363w (ESC2) and YGL127c (SOH1) exhibit DNA processing function, but these are described in free text entries. Since UniProt assigns its own GO URIs, it was necessary to create a mapping [13] from those URIs to the PURL URIs used by GO. By additionally importing the GO-slim ontology, we can query the enhanced knowledge base with classifiedWith some Molecular_function to retrieve genes/proteins that have GO functional annotation. Among the results, we discover that the pre-mRNA-splicing factor CWC21 (Q03375/S000002890) is annotated with protein binding function.

3.1.2. Instantiation of the Gene Ontology

By adopting the BFO in this work, we subscribe to the idea that molecular functions, cellular components and biological processes really do exist and as such each gene/protein instantiates its own GO instance. For simplicity of reasoning about GO slim terms from SGD data file, we designed an OWL-DL version of the yeast GO slim ontology [12] comprises of 79 classes spanning the three hierarchies (molecular function, cellular component and biological process). The use of this GO slim ontology substantially reduces the complexity of reasoning over the full GO ontology (with over 19,000 terms), but it is still possible to import the full ontology to reason with as each class is equivalent to a GO class. We created an instance of the correspondent GO term for each GO annotated gene/protein, which opens the door to making future statements about those particular functions, processes and components.

x


### 3.3.2. Query 2. Find all uncharacterized open reading frames that have a known molecular function

**Class Expression:** OpenReadingFrame that (hasFunction some Molecular_function) and (hasStatus some (ORFStatus and not (Verified or Dubious)))

This conjunctive query illustrates how (i) multiple restrictions (OpenReadingFrame, hasFunction, hasStatus) on an individual must be satisfied and (ii) requires background knowledge of status types (Verified and Dubious are known and different ORF status types from other types—i.e. uncharacterized). The result to this query contains among others, a putative F-box protein encoded by the ORF with the SGD identifier S000005255 and whose molecular function is protein binding. Since uncharacterized open reading frames are those that likely encode a protein, but for which there are no specific experimental data demonstrating that a gene product is produced in *Saccharomyces cerevisiae*, such queries open new avenues for experimental investigation and validation.

### 3.3.3. Query 3. Find pathway participants that are also physical interaction participants

**Class Expression:** PathwayParticipant and PhysicalInteractionParticipant

This query illustrates the use of conjunctive queries with defined classes for knowledge discovery. A defined class relies on necessary and sufficient conditions to logically describe its membership requirements. An OWL-DL reasoner (e.g. Pellet, Racer, FACT++) will discover which individuals satisfy the class restrictions and will classify such individuals as instances of that defined class. The yOWL ontology contains the class PathwayParticipant for which the necessary and sufficient conditions for membership are: (i) be an instance of an independent continuant and (ii) is a participant in some pathway. The class InteractionParticipant is defined to be (i) an instance of an independent continuant and (ii) participant in some physical interaction experiment. The full query can be posed in terms of primitive (not defined) classes: *Find all continuants that participate in a pathway and participate in a physical interaction experiment*. Both queries return the same set of individuals as an answer, which includes among others: the BET4 gene (YJL031C; SGDID S000003568) whose protein product has a known role in a protein modification pathway and has been shown to interact in five physical interaction experiments (affinityCapture-MS, FRET, reconstituted complex, dosage rescue and co-purification). Thus, knowledge that spans different curated information can be easily queried.

### 3.3.4. Query 4. Find all open reading frames (ORF) on chromosome 5

**Class Expression:** OpenReadingFrame that isPartOf value chromosome5

An important aspect of biological modeling is the use of transitive parthood relations. Within the BFO framework, both objects and processes can have parts, although they are restricted to the same type (i.e. an object can have another object as its part). In the yOWL ontology, open reading frames are parts of chromosome strands which themselves are part of a chromosome. The result of this query includes the YEL059C-A ORF (SGDID S000002954) which is asserted to be part of the Crick strand that is part of chromosome 5. Given that the part of relation is transitive, the reasoner can infer that this ORF is also part of the chromosome 5.

### 3.3.5. Query 5. Find all the interaction experiments that are referred in at least 4 publications

**Class Expression:** InteractionExperiment that (hasReference min 4 Publication)

This query imposes cardinality restrictions over the property hasReference. The result contains the AffinityCapture-MS interaction experiment with SGD identifier interaction_173, which is referred in the publications identified by PMID 1805837, 12374754, 16429126 and 16554755. In Racer, queries including cardinality restrictions cannot contain transitive relations [43]. When the cardinality restrictions involve operators like “at most” or “exactly”, it is necessary to state that we have all the knowledge required to answer the question (i.e. close the world at query time), and the semantics of the query would change from at most four references to at most four known references. Such queries typically require the use of the “unique name assumption” in order to distinguish between individuals that have not been explicitly asserted to be different. This configuration option is offered by both, Racer and Pellet reasoners. Otherwise, axioms containing the OWL differentFrom property axioms would need to be added to the ontology.

### 3.3.6. Query 6. Find all DNA regions that are not physically mapped

**Class Expression:** DNARegion and not (hasStartCoordinate some int) and not (hasEndCoordinate some int)

All physically mapped DNA regions have a known start coordinate and end coordinate along the chromosome. The result of this query contains, among others, negative regulator gene in general amino acid biosynthetic pathway (SGDID S00000929174), which is an instance of “NotPhysicallyMappedFeature”. In the yOWL ontology, “NotPhysicallyMappedFeature” is equivalent to the query, and can be answered either under normal open world assumption or with negation as failure, as implemented by Racer Pro.

The following queries are considered under the category of graph pattern based query, and therefore, the result will contain...
the set of values (bindings) for each variable (node) in the pattern graph queried that satisfy the conditions described in such a graph.

3.3.7. Query 7. Find genes that play a role in transcription and are participants in some genetic interaction experiment. Return both the genes and their associated publications

The graph pattern for this query is illustrated in Fig. 3. The result of this query is a set of tuples, each one containing the values for publications and the genes they reference that satisfy both conditions: (i) play a role in transcription and (ii) are participant in a genetic interaction experiment. This result contains among others the tuple: publication (PMID:16431986) with the gene (S000005705) whose gene product is a part of the APT subcomplex of cleavage and polyadenylation factor.

3.3.8. Query 8. Give the set of identifiers and their database sources for genes involved in a protein modification pathway

The graph pattern for this query is illustrated in Fig. 4. This query illustrates the use of OWL sameAs property to assert that cross references are identifiers to the same object, thereby facilitating the aggregation of information from multiple resources. This query returns, among others, the individual with the identifiers (id from source): S000003568, YLJ031c and BET4 from SGD, PWY30-11 from MetaCyc, YLJ031c (also found as YLJ031c) from EUROSCARF, 4594 from DIP, orf10.1039 from CGD, CA1034 from CandidaDB, CAA89323.1 and AAA21386.1 from GenBank/EMBL/DBJ, 33728 from BioGRID, UP10000345FSE and Q00618 from EBI, NP_012503.2 and 853421 from NCBI. Thus, even when sources assign multiple identifiers, or there exists multiple spellings, yOWL integrates this knowledge and facilitates query answering.

3.3.9. Query 9. Find all information related to Gene NSA3

The graph pattern for this query is illustrated in Fig. 5. As mentioned in the Ontology design and ontology population section, yOWL contains a property hierarchy, whose top property is isRelatedTo. Therefore, every relation (asserted or inferred) between NSA3 and any other individual, will imply that NSA3 is related to that individual. This query retrieves all the individuals NSA3 is related to at the most general level of granularity: (sources) EBI, CGD, NCBI, GenBank/EMBL/DBJ, BioGRID, DIP and CandidaDB, (ORFStatus) verified, chromosome8_Watson, proteasome_complex, (GO identifiers) GO_S000001094_Ribosome_biogenesis_and_assembly, GO_S000001094_Nucleolus, GO_S000001094_Protein_catabolic_process and GO_S000001094_Protein_binding. It is also related to a large set of interactions, and a set of publications including PMID:16922378. This exploratory query can later be refined searching for more specific types of relations between NSA3 and other individuals (e.g. find all the molecular functions related with NSA3 or the location of NSA3).

3.3.10. Query 10. Find genes/proteins with transferase activity that are part of a complex, have rescued a non-viable phenotype by overexpression and have a known role in some pathway. Retrieve the genes, sources, pathways, experiments, chromosome and complex that satisfy these requirements

The graph pattern for this query is illustrated in Fig. 6. This query represents a more sophisticated query that spans a greater portion of the yOWL knowledge base. It requires the integration of all the information including ontology-based inferences (e.g. be part of a chromosome) and ontology integration (e.g. the transferase activity GO term). The answer to this query is the set of tuples containing a gene, source, pathway, experiment, chromosome and complex that together with the relations satisfy the restrictions defined in the query. An answer to this query is the gene BET4, with source SGD that is a participant in the pathway_1 and also in the experiment_30. This experiment has as outcome the phenotype_30 that has the quality of being non-viable. BET4 also participates in the interaction_871 obtained from a Dosage Rescue Genetic Interaction Experiment. This gene is annotated with the S000003568.GO_Transferase_activity, an instance of the GO transferase activity molecular function and is also part of the chromosome10 and the Rab-protein_geranylgeranyltransferase_complex, which is an instance of complex. Each unbound variable of the query (unfilled circle) is bounded to an individual that satisfies the restrictions imposed in the query.

4. Discussion

In previous sections we presented the prototype of an ontology driven knowledge management system in which the semantics of RDF and OWL are used to integrate and query over heterogeneous biological knowledge. We will now discuss in greater detail some features, lessons learned and remaining challenges.

4.1. Ontology design and ontology population

While the approach used to design the ontology (generate bag of concepts, place in hierarchy, enhance with additional domain knowledge, establish relations, add necessary conditions) follows general practices, there are two distinguishing aspects in this work. The first is that we used the conceptual framework provided by the BFO upper level ontology to guide the design of
classes and the use of the BRO basic relationship ontology to establish the basic relationships between the classes. The use of an upper level ontology will also permit semantic mapping of classes with other ontologies by establishing, in the very least, whether the class is a type of quality, object or process. A second point is that our methodology was specifically applied to remodel information that is presumably stored in a relational database without having access to the schema, which required a high human intervention. However, our manual modeling process forms the basis by which classes and relations may be automatically extracted from a database model (e.g., database schema, data files, etc.). An important aspect of this process is to clearly identify when a relation is of the type is a or that of partwhole, both of which are widely used in the biology domain in taxonomies and mereologies. Following an automatic extraction would be the more challenging aspect of identifying existing ontologies with compatible semantics. In this paper, we trivially generated a GO slim ontology that references classes defined in a much larger GO document, thereby specializing an existing ontology for practical considerations, and supporting the idea that a mapping to existing ontologies could be done after the automatic generation of classes in one’s own namespace.

The task of ontology population is labor intensive and requires intervention by a technical and domain expert. However, the development of RDF data converters for a variety of heterogeneous data sources, as being done with the Bio2RDF project, is a clear first step towards the population of OWL ontologies. In this work, we integrated SGD and UniProt RDF data into our OWL ontology so as to execute queries across these information sources.

4.2. Knowledge management

Although a large amount of information about yeast genes and proteins is available at SGD, new experimental information is being generated everyday in laboratories across the world. Clearly, facile methods to publish information and rapid access to new knowledge would be incredible assets in scientific knowledge discovery. Semantic web languages such as RDF and OWL facilitate publication of machine understandable information (i.e., RSS) and allow automated integration and querying from distributed web sources, and hence these are suitable in moving forward towards this enhanced vision of biological data management. The use of OWL ontologies and easy to use interfaces facilitates the management of a semantic knowledge base to keep track of information and execute novel queries. For instance, let us add the recent discovery of 11 genes involved in meiotic DNA processing based on genetic interaction experiments [44]. We create a new ontology using Protege4, we import the yOWL ontology, add functional annotation to each of the genes using the GO terminology, and publish the document at some web accessible location. Thus, scientists are now empowered to create their own knowledge bases on literature, experiments, etc., and can easily share them, thus creating a new network of distributed, machine understandable, scientific knowledge bases.

4.3. Gene–protein resolution

Despite SGD’s recent thrust to improve its annotation of proteins [45], this resource does not clearly differentiate between genes and their gene products (i.e., proteins). This is enormously problematic in reasoning about processes that involve proteins and not genes, as these should be considered disjoint. In particular, experiments that deal exclusively with genes (microarray), or proteins (two hybrid), and making strong statements via necessary conditions will lead to inconsistencies. While it will ultimately become necessary to differentiate between genes and proteins, this will require active curation or integration of knowledge which links these for yeast. However, since yeast biologists routinely use gene/protein names interchangeably, they might expect to see all relevant information when asking about either one, thus making the distinction unnecessary and inconvenient.

4.4. Semantic knowledge integration in the biological sciences

Given the broad scope of this ontology, we recognize that there may be subsets that overlap with other ontologies, particularly community-driven ontologies that are part of the Open Biomedical Ontologies (OBO). Two major issues that precluded the use of the first generation of OBO ontologies were: (i) OBO ontologies did not adopt OWL semantics, (ii) few of them adhered to logical subsumption. Efforts via the OBO Foundry aim to redesign OBO ontologies by adopting the formal ontology of the BFO and by providing

---


15. http://ontology.dumontierlab.com/yowl-jbi-example.owl
clearer guidelines for ontology design. Recent work has also established formal semantics to transform OBO ontologies into OWL [12–14]. Thus, we anticipate that we will be able to map our concepts to OBO ontologies in the near future.

A controversial aspect of our work relies on the use of a self-declared base URI for all third part identifiers of sources without resolvable URLs, in order to facilitate future data integration. This problem is solely rooted in the fact that currently there is no global directory for base URIs of public biological data providers. Since OWL inherits the semantics of RDF, instances may be assigned to different namespaces. While we can state that the resource identified by UP10000052DF0 is an instance of http://ontology.dumontierlab.com/yowl-1.0#OpenReadingFrame, we would expect that the proper namespace of that individual belongs to the original data provider, SGD in this case. Some data providers, such as UniProt will provide RDF or OWL documents with these identifiers (e.g. http://beta.uniprot.org/uniparc/UP10000036C3F.rdf). Unfortunately, many data providers, including the SGD, do not make their information available in RDF, and do not provide a base URI from which their resources may be identified. Thus, it falls on a third party to assign a unique URI. One way to do this is assign an HTTP URI (URL) in the data provider's namespace i.e. http://yeastgenome.org/UP10000052DF0. But this choice is wholly arbitrary, and it falls on the data provider to resolve that URI, even if there is no resolver that exists today or in the future. Without a proper URL resolver, semantic web clients would query the host server and waste network bandwidth in trying every single identifier and never learning more about the resource. Another possibility is to assign a URI in the third party's namespace, i.e. http://dumontierlab.com/PI0000052DF0, and potentially offer a service to resolve such a URI. Aside from whether the third party actually wants to serve the data, a more fundamental problem with this approach is that different URIs might be assigned for each third party data provider for the very same resource, thereby effectively dismantling the potential of RDF semantics to trivially integrate what should be an identical resource. Another possibility (the one we adopted) is the use of the Life Science Identifier (LSID), a location-independent encoding of resource (URN) [46]. The advantage of this approach is that the URL resolution of the entity is done via another protocol, therefore allowing changes in URL end-points. Unfortunately, there are two issues with the LSID approach: data providers must (i) register with an LSID authority directory and (ii) implement a resolver that will convert the URN into a URL internet resource. Problematically, many data providers have not subscribed to the LSID resolution mechanism and therefore there may never be resolution for these entities. Compounding this problem, the LSID authority directory was not available at the time of this study, and we were forced to assign LSIDs based on the data providers root DNS. Should the data providers register with the LSID authority some future date with a different LSID, we can add another sameAs statement to our knowledgebase to enable data integration. Alternatively, a case might be made for new OWL semantics to make namespaces equivalent. In any case, yOWL is ready to integrate data containing LSID identifiers or data providers URLs when available.

Realizing the vision of data integration requires that statements made in different sources about a single resource be considered equivalent. As outlined in the introduction, bioinformatics databases are particularly keen on maintaining their own identifiers to maintain provenance about value added contributions. This approach results in a number of equivalent identifiers for the same resources. Using the OWL property sameAs, database cross references are made equivalent to the SGD resource. Thus, a user may query the knowledge base using any of the equivalent identifiers and return the union of statements about that resource. Some responders, such as Racer, provide the means to query only asserted knowledge, thereby retrieving knowledge of some subset of data providers. Such behavior is particularly well suited for users wishing to filter the knowledge base depending on the data or data providers they trust.

4.5. Knowledge discovery

The ability to define classes in OWL (e.g. PathwayParticipant and PhysicalInteractionParticipant) given a set of (logically described) necessary and sufficient conditions, allows a reasoner to infer the individuals that belong to that defined class. This is a significant difference between RDF and OWL, where the former has no such vocabulary to express this concept. A reasoner can also determine that a user's query corresponds to a class already defined in an ontology. Thus, individuals that belong to defined classes will be classified by the reasoner in the realization process and will not require on the fly query evaluation, which will play a role in improving query performance over greater amount of data.

4.6. Granular semantic search

The yOWL ontology supports semantic query formulation across various levels of granularity in both class and property hierarchies. In the case of class hierarchies, a scientist may generally ask about DNA regions or query specialized DNA regions (e.g. Open Reading Frame). Most of the available biological data models contain information in a specific granularity level, the population of classes using their definitions, as explained in previous paragraph, will allow a reasoner to infer the membership of individuals to classes that otherwise would be gaps precluding the integration of data at these levels of granularity (e.g. Query 3 that includes a PathwayParticipant, a class that does not contain any asserted individual). In the case of property hierarchies that are rooted on a non-transitive relation, one can ask whether there is any relation between two or more objects with multiple unknown concepts between them. This provides a general data mining approach to discover relations between two or more resources.

4.7. Transitive relations

OWL ontologies provide the ability to define transitive relations (e.g. if an Open Reading Frame is part of a Chromosome Strand, and the Chromosome Strand is part of the Chromosome, then the Open Reading Frame is part of the Chromosome). These relations are very useful in knowledge discovery. Transitive relations in relational databases are not straightforward as they require recursive SQL queries that extend relational algebra. This is hard to maintain given the information needed a priori (e.g. database schema, data-types) that may limit the scope of the application, making it domain dependent. Also, the user will need to have a previous training on SQL queries, which is uncommon among biological scientists. The use of transitive and composed properties among other is a clear advantage of the use of ontologies over relational databases.

4.8. Closed world reasoning and unique name assumption

Life sciences terminology often requires cardinality restrictions over properties (e.g. a carbon atom has exactly 6 protons) and negation (e.g. individuals that are DNA Regions but are not physically mapped). Life sciences ontologies are populated from databases where different names represent different entities. For these reasons, the ability of using negation as failure (implemented by RacerPro and its query language nRQL) and the ability to apply Unique Name Assumption (RacerPro and Pellet) played an important role in the query answering, especially for queries for knowl-
edge discovery. These features are also important in the population of defined classes containing cardinality restrictions containing the “at least”, “at most” and “exactly” operators. Given that new hypothesis are typically generated using the available evidence, a scientist would like to pose queries with the assumption that all the knowledge is available at certain point, and therefore “closed-world” reasoning is necessary and applicable in the biology domain.

4.9. Query answering interfaces

The construction of semantically correct queries is facilitated by user-friendly interfaces. Protégé 4 offers to the users the ability to construct class queries using English phrase-like phrases (the Manchester OWL Syntax [42]). The Protégé 4.0 DL query plug-in aids in the construction of the query by dynamically suggesting the phrase grammar and available entities, relations and individuals. This kind of interface helped in the construction of sophisticated queries with no training required. Unfortunately, class queries do not return the individuals that bound the graph pattern based queries, which is essential when users want to identify multiple data that they are interested in. While RacerPro returns the set of individuals that related satisfy the graph pattern based query, and is generally quite powerful, we found it difficult to construct the nRQL queries using the RacerPorter interface. We are aware that some efforts have been made to implement more intuitive interfaces to nRQL [26] including a graphical query language [47]. We expect this trend to continue and will facilitate the use of semantic web applications by the scientific community.

4.10. Scalable data management

A major challenge remains with the efficient storage and retrieval of ontological data. Current applications necessarily store all data in memory to execute reasoning tasks. It will be difficult, if not impossible to store large databases with all their inferences in memory without sophisticated hardware. In our study, due to limited resources, we necessarily restricted both the ontology size (GO slim instead of the full GO) and amount of data to reason about (2% subset of about 320,000 instances). Databases such as Instance Store [48] provide a first step towards reasoning databases, but to our knowledge this is currently limited to role-free queries. More sophisticated solutions are clearly required, like the efforts towards ontology modularization described in [49,50]. Our contribution towards this goal is the generation of decidable ontologies with large amounts of data (individuals) as use cases to represent the needs for scientist for the testing and development of new and data management (including reasoning) approaches.

5. Conclusions

In this work, we described a first approach to logically describe, integrate and query yeast biological data from the SGD database using the OWL-DL ontology language. Several features make this work unique. First, we designed a domain specific ontology that incorporates concepts drawn from raw data and expert knowledge. We maintained upper level semantics by differentiating between objects and processes, using basic relations and instantiating functions, components and processes for each gene/protein based on the Gene Ontology. We described the methodology used for semantically augmenting biological databases in absence of domain specific ontologies and discuss its scope and characteristics. We also made use of OWL-DL semantics to integrate identical resources from different data providers and overcome the problem of heterogeneous identifiers integration. We illustrated the process of adding additional knowledge from UniProt (automatically) and a relevant publication (manually) in a new ontology for modularity purposes. Finally, we illustrated the use of diverse queries at various levels of ontological granularity using DL reasoners, highlighting the difference of using open and closed world semantics and their importance in the biology domain. This work marks a beginning for using the semantic web framework in yeast knowledge discovery. However, significant challenges remain in realizing the potential of the semantic web, such as the automated creation and population of ontologies (via ontology lifting), the efficient storage and reasoning with large amounts of ontological data for reasoning and the development of intuitive interfaces among others.

Acknowledgments

This paper is an extension of the paper [51] published in WWW2007/HCLSDI workshop proceedings. We thank the anonymous reviewers for providing constructive feedback that has improved the quality of this paper. Funding for N.V.R. and M.D. was with CONACYT scholarship #150581 and NSERC, respectively.

Appendix A. Supplementary data

Supplementary data associated with this article can be found, in the online version, at doi:10.1016/j.jbi.2008.05.001.

References


