

Complex Queries for Hypothesis Validation on the Web

Pablo N. Mendes, Amit P. Sheth

Knoesis Center, Computer Science & Engineering, Wright State University

<http://knoesis.wright.edu>

Introduction

Biomedical datasets:

- Numerous, distributed
- Heterogeneous, independently managed
- Different levels: genomic, phenomic, pharmaceutical, clinical, etc.

An integrative approach requires the ability to ask **complex questions** across the entire spectrum. We use ontologies to facilitate data integration and enable richer querying capabilities.

We are applying our knowledge-driven query formulation system to support research on *Trypanosoma cruzi*, causative agent of Chagas disease, one of the most globally significant tropical diseases.

Hypothesis Validation

"Is the gene XYZ a potential vaccine candidate to Chagas disease?"

- 1) Find all genes annotated as potential vaccine candidates
- 2) Find all genes with proteomic expression evidence in the mammalian host life-cycle stages with glycosylphosphatidylinositol (GPI) anchor or signal peptide predictions for *T.cruzi*.

Complex Queries

Researchers can define and execute queries at different levels of complexity, including different types of relationships:

Atomic relationships are trivially found in the data through the querying mechanism.

e.g. gene has *protein expression* prot_exp

Computable relationships can be automatically established by the invocation of a Web service.

e.g. gene1 has *best blast hit* gene2
gene has *protein domain* domain

Complex relationships can be defined in terms of other atomic, computable and complex relationships. The definition of complex relationships can be performed through our interface in the same way queries are built.

e.g. gene is *vaccine candidate* to disease (see blue box)

Path relationships can include any number of atomic, computable or complex relationships. They differ from complex relationships in that they are not predefined. Any sequence of connected entity-relationship pairs between a given start and end are valid path relationships.

e.g. "How are geneA and geneB related?"

Spatiotemporal relationships are complex relationships that connect entities to their locations and time-related contexts.

e.g. gene *sequence_overlap* gene
gene_expression1 *overlaps_in_time* gene_exp_2

RESEARCHER



- 1 Start typing a **term** to include in your query. Select desired term from the list or choose "any term."

geneXYZ
gene
geneABC
geneXYZ <input type="checkbox"/>
genome
genotype

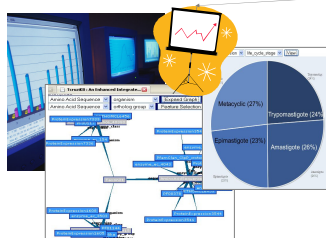
- 3 Choose **relationship** of interest. Alternatively, say "any relationship."

gene
geneXYZ
vaccine_candidate
?any_relationship
expression
interacts_with
organism
product
vaccine_candidate <input type="checkbox"/>

- 5 Repeat process until the desired query is achieved. Choose to execute query when done. System encodes SPARQL and submits query.

gene	disease
geneXYZ	ChagasDisease

- 8 Display results in all visualization components.



DATA SERVER

- 1 (simultaneously while typing) Search ontology for possible **terms** that match the typed text.
- 2 Search ontology for possible **relationships** for the selected term.
- 4 Search ontology for possible **terms** for the selected relationship.
- 6 Execute SPARQL query over dataset.
- 7 Check policies for:
 - translating complex relationships;
 - triggering service executions for computable relationships.

Complex relationship definition:

```
?gene vaccine_candidate ?disease IF
?gene organism ?organism .
?organism causes ?disease .
?gene expression ?exp .
?exp life_cycle_stage ?stage .
?stage hosted_by ?host .
?host a txn:Mammal .
?gene product ?protein .
?protein pfam_domain pfam:GPI_anchor
```

Computable relationship trigger:

```
?protein pfam_domain ?domain
Property function: pfam_domain
Class: HmmerWebServiceInvoker
Input: seq=?protein, db=Pfam
Output: ?domain
```



Implementation

The key enabler of the query builder is the ontology schema. The schema defines what are the possible classes and relationships between those classes in the ontology. We currently support schema definitions in RDFS. In cases where the schema is absent (pure RDF), our system is able to traverse the RDF graph and infer the possible domains and ranges of the relationships for that dataset.

This prototype is based on:

- Ajax to provide the dynamic look-and-feel, performing data exchange in the background;
- Resource Description Framework (RDF/S) to represent the ontology-based description of data;
- SPARQL as the language to encode the queries built through the interface;
- SPARQL protocol to communicate with servers (SPARQL Endpoints) for query execution.

Dataset

Trypanosoma cruzi genome database (**TcruziDB**) provides access to a relational database containing, among others:

- Genome;
- Proteomic expression;
- Ortholog groups;
- Protein domain predictions.

We mapped the database to ontologies through the D2RQ software (FU-Berlin). We also reuse several vocabularies and ontologies, including the Gene Ontology, Sequence Ontology and NCBI Taxonomy.

Since typical genome databases do not contain information that is referred to as common knowledge (such as descriptions of life-cycle stages), we augmented the data by creating ontologies to hold such knowledge.

Acknowledgements

Thanks to Jessica Kissinger, the TcruziDB and the GUS teams for facilitating the access to data and for useful insight. Special thanks also to Bobby McKnight for contributing with the results visualization interfaces.