Towards ubiquitous OWL computing: Simplifying programmatic authoring of and querying with OWL axioms

Hilmar Lapp, Jim Balhoff
National Evolutionary Synthesis Center (NESCent)
BOSC 2014, Boston
Did all the work:

Jim Balhoff, Programmer extraordinaire

http://github.com/balhoff
RDF is a powerful data integrator

Ontologies allow integrating across descriptive biology

Descriptive biology
- Phenotypes
  - Traits
  - Function
  - Behavior
- Habitat
- Life Cycle
- Reproduction
- Conservation Threats

Biodiversity (Specimens, Occurrence records)
Taxonomy, Species ID
Conservation Biology
Ecology
Genetics
Physiology
Genetic variation
Genomics, Gene expression
Translational bioinformatics

Model organism
-> Human

Fig. 1, Washington et al (2009)

Fig. 3, Washington et al (2009)
Model organism genes -> Evolutionary diversity by semantic similarity of phenotypes
Integrate across studies & fields by virtue of ontologies

Comparative studies + Model organism datasets = Phenoscape Knowledgebase
Computable via shared ontologies, rich semantics, OWL reasoning
- **kb-owl-tools**
  - **OWL conversion**
    - Includes translation of EQ to OWL expressions
  - **Identifier cleanup**
    - URIs for standard properties across ontologies are a mess
  - **Axiom generation**
    - "Absence" classes for OWL EL negation classification workaround
    - SPARQL facilitation (e.g. materialized existential hierarchies such as part_of)
  - **Assertion of absence hierarchy**
    - Based on inverse of hierarchy of negated classes computed by ELK
  - **Materialize inferred subclass axioms**
    - ELK reasoner using extracted tbox axioms only (not feasible with individuals included)

**MOD curators**
- MODs phenotype & gene expression annotations
- gene identifiers

**Phenoscape curators**
- NeXML matrices

**Ontology curators**
- ontologies

**Bigdata triplestore**
- RDF (all)
- tbox axioms

**Owlet SPARQL endpoint**
- OWL

**SPARQL queries**
- applications

**Phenoscape curators**
- Ontology curators
Ontology axiom authoring at scale is cumbersome

- If a developmental precursor is absent, the structure is absent.

“For every anatomical structure, assert that the class of things not having the structure as a part is a subclass of the class of things not having the structure’s developmental precursor structure as a part.”

Scowl: marrying programming to ontology language

“For every anatomical structure, assert that the class of things not having the structure as part is a subclass of the class of things not having the structure’s developmental precursor as part.”

```plaintext
for {
    term <- reasoner.getSubclasses(anatomical_structure)
} yield
(not (hasPart some term)) SubClassOf (not (hasPart some (developsFrom some term)))
```
Scowl

- Allows declarative approach to composing OWL expressions and axioms using the OWL API.
- Implemented as a library in Scala
- Exploits ‘implicit class’ construct in Scala
  - Compiler turns declarative expressions into JVM instructions
- Class, Annotation, Property axioms

```scala
val hasFather = ObjectProperty("http://example.org/hasFather")
val hasBrother = ObjectProperty("http://example.org/hasBrother")
val hasUncle = ObjectProperty("http://example.org/hasUncle")
val axiom = hasUncle SubPropertyChain (hasFather o hasBrother)
```
http://douroucouli.wordpress.com/2014/03/30/the-perils-of-managing-owl-in-a-version-control-system/
object AnatomyOntology extends App {

  val factory = OWLManager.getOWLDataFactory
  val ns = "http://example.org/anatomy.owl#"
  val head = Class(ns + "001")
  val body = Class(ns + "002")
  val hand = Class(ns + "003")
  val arm = Class(ns + "004")
  val anatomical_structure = Class(ns + "005")
  val part_of = ObjectProperty(ns + "006")
  val label = factory.getRDFSLabel

  val ontology = Ontology("http://example.org/anatomy.owl", Set(
    head Annotation (label, "head"),
    head SubClassOf anatomical_structure,
    head SubClassOf (part_of some body),
    head SubClassOf (not(part_of some arm)),

    body Annotation (label, "body"),
    body SubClassOf anatomical_structure,

    arm Annotation (label, "arm"),
    arm SubClassOf anatomical_structure,
    arm SubClassOf (part_of some body),

    hand Annotation (label, "hand"),
    hand SubClassOf anatomical_structure,
    hand SubClassOf (part_of some arm)))

  ontology.getOWLOntologyManager.saveOntology(
    ontology,
    IRI.create(new File(args(0))))
}
Similarly motivated effort: Tawny-OWL

- By Phil Lord
- Allows construction of OWL ontologies in Clojure.
- “the ontology engineering equivalent of R”
- [https://github.com/phillord/tawny-owl](https://github.com/phillord/tawny-owl)
Scowl

- Home: http://github.com/phenoscape/scowl
- MIT-licensed
Ontology-driven querying in SPARQL is not pretty

- Awkward, lengthy, complicated
- Error-prone
- Slow

SPARQL: genes expressed in head muscles

```sparql
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX ao: <http://purl.obolibrary.org/obo/my-anatomy-ontology/>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
SELECT DISTINCT ?gene
WHERE {
  ?structure rdf:type ?structure_class .
  # Triple pattern selecting structure:
  ?structure_class rdfs:subClassOf "ao:muscle" .
  ?structure_class rdfs:subClassOf ?restriction
  ?restriction owl:someValuesFrom "ao:head" .
}
```
Want to allow arbitrary selection of structures of interest, using rich semantics:
(part_of some (limb/fin or girdle skeleton)) or (connected_to some girdle skeleton)

RDF triplestores provide very limited reasoning expressivity, and scale poorly with large ontologies.

However, ELK can answer class expression queries within seconds.
owlet: A little OWL in SPARQL

- **owlet** interprets OWL class expressions embedded within SPARQL queries
- Uses any OWL API-based reasoner to preprocess query.
  - We use ELK that holds terminology in memory.
- Replaces OWL expression with FILTER statement listing matching terms
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX ao: <http://purl.obolibrary.org/obo/my-anatomy-ontology/>
PREFIX ow: <http://purl.org/phenoscape/owlet/syntax#>

SELECT DISTINCT ?gene
WHERE {
  ?structure rdf:type ?structure_class .
  # Triple pattern containing an OWL expression:
  ?structure_class rdfs:subClassOf "ao:muscle and (ao:part_of some ao:head)"^^ow:omn .
}

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX ao: <http://purl.obolibrary.org/obo/my-anatomy-ontology/>
PREFIX ow: <http://purl.org/phenoscape/owlet/syntax#>

SELECT DISTINCT ?gene
WHERE {
  ?structure rdf:type ?structure_class .
  # Filter constraining ?structure_class to the terms returned by the OWL query:
  FILTER(?structure_class IN (ao:adductor_mandibulae, ao:constrictor_dorsalis, ...))
}
owlet: A little OWL in SPARQL

- Implemented in Scala, can be used in Java
- Home: http://github.com/phenoscape/owlet
- MIT-licensed
- Unique to owlet, compared to some related efforts (e.g., SPARQL-DL, Terp in Pellet):
  - Any SPARQL endpoint / OWL API-based reasoner combination
  - No non-standard syntax
owlery: REST web services for owlet

- Allows federation via SERVICE keyword in SPARQL
- http://github.com/phenoscape/owlery

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX ao: <http://purl.obolibrary.org/obo/my-anatomy-ontology/>
PREFIX ow: <http://purl.org/phenoscape/owlet/syntax#>

SELECT DISTINCT ?gene
WHERE
{
  ?structure rdf:type ?structure_class .
  # Triple pattern containing an OWL expression, handled by federated query:
  SERVICE <http://owlery.example.org/sparql> {
    ?structure_class rdfs:subClassOf
    "ao:muscle and (ao:part_of some ao:head)"^^ow:omn .
  }
}
Summary

• Programming and computing with OWL is more complicated than need be

• Need a thriving ecosystem of small tools that fill common gaps

• Scowl, Owlet, and Owlery are small steps in that direction

• Vision: programming with ontologies as easy and ubiquitous as with alignments
Acknowledgements

- Jim Balhoff (NESCent, Phenoscape)
- Chris Mungall (LBL)
- Ontology engineering community (incl. Phil Lord, OWL API)
- Phenoscape personnel, PIs, and curators
- National Evolutionary Synthesis Center (NESCent)
- NSF (DBI-1062404, DBI-1062542)