POSH : The Prolog OWL Shell

Chris Mungall

Lawrence Berkeley Laboratory

Ontology Hacking

- I'm in a hurry and I want to:
  - translate all "*R some Y*" to "(*R some Y) or Y*"
  - add an equivalence axiom for every class pair with closely matching labels
  - automatically extend an ontology using some tabular data files and/or relational db
  - iteratively refactor my ontology based on complex structural pattern matching
- What are my options?

Ontology Environments

- GUI
  - Protégé 4
  - OBO-Edit
- API
  - Java
    - OWL API
    - Jena
    - Ortiz

Ontology Environments

- GUI
  - Protégé 4
  - OBO-Edit

} Not powerful enough!
need programmatic capabilities

- API
  - Java
    - OWL API
    - Jena
    - Ortiz

} I'm in a hurry!
too verbose;
hard to read and write;

Where is the Perl of OWL?

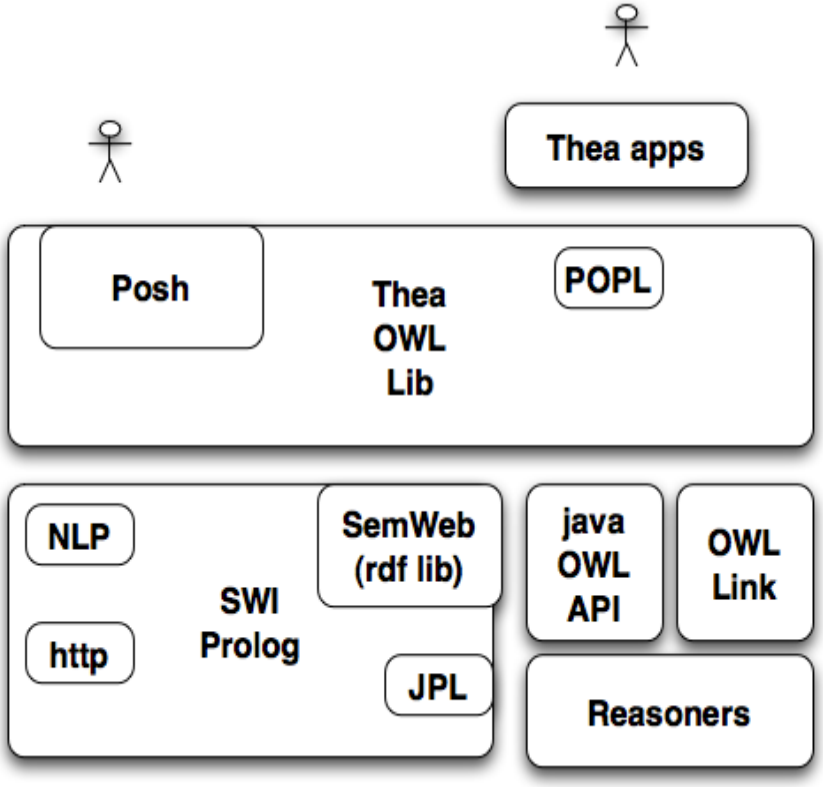
- Perl is remarkably good for **slicing, dicing, twisting, wringing, smoothing, summarizing**, and otherwise **mangling** text…
- Perl programs are **easy to write** and **fast to develop**…
- Perl is a good **prototyping** language…

– Lincoln Stein, **How Perl Saved the Human genome Project**, *Dr Dobbs Journal*, 1996, http://drdobbs.com/184410424

- What's a good way for slicing, dicing, mangling and hacking *axioms* and *expressions*?

Pathologically Obfuscated Semantic Hacking (POSH)

- What is it?
  - Command line interface to Thea (Vassilidas, OWLED2009)
- Features
  - OWL2 Manchester-like syntax
  - Command Line (REPL)
  - Declarative
    - but with full 'impure' programmatic capabilities
    - Turing complete
  - Succinct
  - Configurable and extendable
  - Behind the scenes label<->IRI translation



Posh Lightning summary

- Infix predicates
  - Axiom predicate shortcuts:
    - < (SubClassOf)
    - == (EquivalentClasses)
    - (or use OWL2 syntax)
  - Expression operators
    - and
    - or
    - not
    - some
    - all
    - min(N)
    - max(N)
    - …
- Prolog syntax
  - variable leading upper case
- Prolog queries
  - Predicates dynamically mapped to queries on in-memory RDF db
  - E.g.
    - forebrain < part_of some brain.

Initiation

```
 $ thea-poshj --format rdf_direct http://
purl.obolibrary.org/obo/uberon.owl

%  ::: Welcome to Posh, the Prolog OWL Shell :::
% Parsed uberon.owl
? - writeln('hello world').
hello world
true.
?- subClassOf(X,Y).
X = 'http://purl.obolibrary.org/obo/UBERON_0000002',
Y = 'http://purl.obolibrary.org/obo/UBERON_0001560' ;
X = 'http://purl.obolibrary.org/obo/UBERON_0000002',
Y = 'http://purl.obolibrary.org/obo/UBERON_0005156' ;
X = 'http://purl.obolibrary.org/obo/UBERON_0000002',
```

standard Thea query

underlying ontology uses numeric IRIs

Querying asserted axioms

```
 ?- q X where X < part_of some brain.
forebrain.
'medial forebrain bundle'.
hindbrain.
'cranial dura mater'.
brainstem.
'nucleus of brain'.
'regional part of brain'.
'midbrain-hindbrain boundary'.
'brain blood vessel'.
'brain grey matter'.
'brain white matter'.
'brain meninx'.
'brain pia mater'.
'subventricular zone'.
'ventricular system of brain'.
'brain vasculature'.
…
```

prolog syntax
is highly configurable!

Posh provides configurable
IRI <-> label translation

Querying inferences

axioms in {}s
sent to reasoner

```
?- init hermit.
?- q X where {X < part_of some brain}.
'cortical layer VI'<part_of some brain.
'commissure of inferior colliculus'<part_of some brain.
'cortical layer V'<part_of some brain.
'cingulate cortex'<part_of some brain.
'brain arachnoid mater'<part_of some brain.
'limitans nucleus'<part_of some brain.
'cortical layer II'<part_of some brain.
'brachium of inferior colliculus'<part_of some brain.
'cortical layer I'<part_of some brain.
'pontine tegmentum'<part_of some brain.
'cortical layer IV'<part_of some brain.
'cortical layer III'<part_of some brain.
'brain arachnoid mater'<part_of some brain.
--[SNIP]--
```

Mixed prolog / reasoner queries

```
?- q X where
  {X < neuron and X < not(part_of some brain)}.
```

open-world

```
?- q X where
  {X < neuron}, \+{X < not(part_of some brain)}.
```

mixed open/closed world
(cf SPARQL FILTER)

POPL: Prolog Ontology Processing Language

```
-- rewrites expressions as if R were reflexive:
?- R some Y ===> Y or R some Y.


-- add equivalence axioms where labels closely match
?- assert(
     sameLabel(X,Y) :- label(X,XN), label(Y,YN), X\=Y,
porter_stem(XN,N),porter_stem(YN,N)
  ).

?- add X==Y where sameLabel(X,Y).
```

Similar tools

- Declarative JVM language with REPL + OWL API
  - Groovy
    - El-Vira (Hoehndorf)
  - Armed Bear Common Lisp
    - LSW (Ruttenberg)
- OPPL
- owl.rb (Balhoff)
- SPARQL + various environments

Availability

- http://blipkit.wordpress.com/posh/

- also distributed as part of Thea:
  - http://www.semanticweb.gr/thea/
  - (check out “posh” branch)