Ontology Access Kit (OAK)

Ontology Summit
2023

https://incatools.github.io/ontology-access-kit
Why OAK? Different computational use cases for ontologies

I need to reason over and build ontologies

I need to analyze data using ontologies

<table>
<thead>
<tr>
<th>Name</th>
<th>Syntax</th>
<th>Semantics</th>
</tr>
</thead>
<tbody>
<tr>
<td>inverse role</td>
<td>$R$</td>
<td>${(x, y) \in A^I \times A^I \mid (y, x) \in R^I}$</td>
</tr>
<tr>
<td>universal role</td>
<td>$U$</td>
<td>$A^I \times A^I$</td>
</tr>
<tr>
<td>top</td>
<td>$\top$</td>
<td>$A^I$</td>
</tr>
<tr>
<td>bottom</td>
<td>$\bot$</td>
<td>$0$</td>
</tr>
<tr>
<td>negation</td>
<td>$\neg C$</td>
<td>$A^I \setminus C^I$</td>
</tr>
<tr>
<td>conjunction</td>
<td>$C \sqcap D$</td>
<td>$C^I \cap D^I$</td>
</tr>
<tr>
<td>disjunction</td>
<td>$C \sqcup D$</td>
<td>$C^I \cup D^I$</td>
</tr>
<tr>
<td>nominals</td>
<td>$[a]$</td>
<td>$[a^I]$</td>
</tr>
<tr>
<td>univ. restriction</td>
<td>$VR.C$</td>
<td>${x \in A^I \mid (x, y) \in R^I$ implies $y \in C^I}$</td>
</tr>
<tr>
<td>exist. restriction</td>
<td>$JR.C$</td>
<td>${x \in A^I \mid$ for some $y \in A^I$, $(x, y) \in R^I$ and $y \in C^I}$</td>
</tr>
<tr>
<td>Self concept</td>
<td>$\text{Self}$</td>
<td>${x \in A^I \mid (x, x) \in S^I}$</td>
</tr>
<tr>
<td>qualified number restriction</td>
<td>$\leq n.S.C$</td>
<td>${x \in A^I \mid #y \in A^I \mid (x, y) \in S^I$ and $y \in C^I} \leq n}$</td>
</tr>
<tr>
<td></td>
<td>$\geq n.S.C$</td>
<td>${x \in A^I \mid #y \in A^I \mid (x, y) \in S^I$ and $y \in C^I} \geq n}$</td>
</tr>
</tbody>
</table>
Why OAK? Different computational use cases for ontologies

- I need to reason over and build ontologies
- I need to analyze data using ontologies

ontodev/robot
ROBOT is an OBO Tool

OAK
Ontology Access Kit
OAK: A Python library for ontology access

Modular python packages: For developers and data scientists

```python
relationships[
    subjects: Optional[Iterable[str]] = None,
    predicates: Optional[Iterable[str]] = None,
    objects: Optional[Iterable[str]] = None,
    include_tbox: bool = True,
    include_aabox: bool = True,
    include_entailed: bool = True
] -> Iterator[Tuple[str, str, str]]
```

Returns all matching relationships

**Parameters:**
- **subjects** – constrain search to these subjects (i.e. outgoing edges)
- **predicates** – constrain search to these predicates
- **objects** – constrain search to these objects (i.e. incoming edges)
- **include_tbox** – if true, include class-class relationships (default True)
- **include_aabox** – if true, include instance-instance/class relationships (default True)
- **include_entailed** –

**Returns:**

```python
roots[
    predicates: Optional[List[str]] = None,
    ignore_owl_things: True,
    filter_obsoletes: True,
    id_prefixes: Optional[List[str]] = None
] -> Iterable[str]
```

All root nodes, where root is defined as any node that is not the subject of a relationship with one of the specified predicates

**Parameters:**
- **predicates** –
OAK: A Python library for ontology access

Modular python packages: For developers and data scientists

Command Line Interface: For everyone!!

50 multi-option commands

```python
relationships
   [subjects: Optional[Iterable[str]] = None, predicates: Optional[Iterable[str]] = None,
    objects: Optional[Iterable[str]] = None, include_tbox: bool = True, include_aabox: bool = True,
    include_entailed: bool = True] -> Iterator[Tuple[str, str, str]]
```

Returns all matching relationships

Parameters:
- `subjects` - constrain search to these subjects (i.e. outgoing edges)
- `predicates` - constrain search to these predicates
- `objects` - constrain search to these objects (i.e. incoming edges)
- `include_tbox` - if true, include class-class relationships (default True)
- `include_aabox` - if true, include instance-instance/class relationships (default True)
- `include_entailed` -

Returns:

```python
roots
   [predicates: Optional[List[str]] = None, ignore_owl_thing=True, filter_obsoletes=True, id_prefixes:
    Optional[List[str]] = None] -> Iterable[str]
```

All root nodes, where root is defined as any node that is not the subject of a relationship with one of the specified predicates

Parameters:
- `predicates` -

Example command:

```bash
runoak -i prontolib:fbbt.obo tree FBBt:00004751 -p i
```

- [i] FBBt:10000000 ! anatomical entity
- [i] FBBt:00007016 ! material anatomical entity
- [i] FBBt:00007001 ! anatomical structure
- [i] FBBt:00007013 ! acellular anatomical structure
- [i] FBBt:00007245 ! cuticular specialization
- [i] **FBBt:00004751 ! wing vein**
### What can you do with OAK?

<table>
<thead>
<tr>
<th>Basic Ontology Lookup</th>
<th>Graph-oriented Operations</th>
<th>Ontology Associations</th>
<th>OWL-oriented Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>lookup/search</td>
<td>Ancestors, descendants</td>
<td>E.g gene to terms</td>
<td>axioms</td>
</tr>
<tr>
<td>definitions, aliases</td>
<td>Visualization</td>
<td></td>
<td></td>
</tr>
<tr>
<td>relationships</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Text and NLP</th>
<th>Validation</th>
<th>Change</th>
<th>Semantic similarity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Annotate text</td>
<td>Structural validation</td>
<td>Diffs</td>
<td>Jaccard</td>
</tr>
<tr>
<td>Ontology Matching</td>
<td></td>
<td>Apply changes</td>
<td>IC-based</td>
</tr>
</tbody>
</table>

**Plugins for:**
- Fast graph operation
- OWL reasoning
- Embedding
Ontology Subgraph Visualization

runoak -i cl.db viz -p i,p 'memory T-cell'

Customizable JSON stylesheet

Compact representation of OWL TBox axioms (e.g. existential restrictions)

https://github.com/INCATools/obographviz
Computing and applying changes: Knowledge Graph Change Language

Changes.kgcl
- Add exact synonym “hand” to UBERON:0002398
- Move ‘astrocyte’ from ‘neuron’ to ‘glial cell’

Curation UI
Synonymizer
Diff Engine

OAK

Apply
Obo or owl file
Make PRs
Oneto Crawler

OAK

https://github.com/INCATools/kgcl
OAK: Common Interfaces to Multiple Backends

https://incatools.github.io/ontology-access-kit/interfaces
OAK: A Common Interface to Multiple Backends

SQL Alchemy

Pronto + fastobo

FunOWL

rdflib

SQL Adapter

OBO Format Adapter

OWL Adapter

SPARQL Adapter

Local Files on Filesystem

Common Interface

method1(...) method2(...) ...

.methodN(...)
OAK: A Common Interface to Multiple Backends

SQL Adapter

SPARQL Adapter

Common Interface

method1(...)
method2(...)
...
methodN(...)

Postgres

UberGraph

OntoBee

Rdflib + sparqlwrapper

OntoPortal Client

Ontoportal Adapter

OLS Adapter

Remote access
Examples of swapping backends

runoak -i cl.db relationships CL:0000540  ## Sqlite is fastest and recommended route
runoak -i cl.owl relationships CL:0000540
runoak -i cl.ofn relationships CL:0000540
runoak -i cl.obo relationships CL:0000540  ## CHEBI syntax errors
runoak -i ontobee:cl relationships CL:0000540
runoak -i ubergraph:cl relationships CL:0000540
runoak -i bioportal:cl relationships CL:0000540  ## Not yet implemented
runoak -i ols:cl relationships CL:0000540  ## Not yet implemented
# OAK wraps and reuses core ontology libraries

<table>
<thead>
<tr>
<th>Task</th>
<th>Library Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prefix &lt;-&gt; URI Mapping</td>
<td>github.com/cthoyt/curies</td>
</tr>
<tr>
<td>SQL Databases</td>
<td>github.com/INCATools/semantic-sql</td>
</tr>
<tr>
<td>OLS Access</td>
<td>github.com/cthoyt/ols-client</td>
</tr>
<tr>
<td>Bioportal/Ontoportal Access</td>
<td>github.com/cthoyt/ontoportal-client</td>
</tr>
<tr>
<td>OBO and OBO Json</td>
<td>github.com/althonos/pronto</td>
</tr>
<tr>
<td></td>
<td>github.com/fastobo/fastobo-py</td>
</tr>
<tr>
<td>Graph inference</td>
<td>github.com/balhoff/relation-graph</td>
</tr>
<tr>
<td>Mappings</td>
<td>github.com/mapping-commons/sssom/</td>
</tr>
<tr>
<td>Ontology Changes</td>
<td>github.com/INCATools/kgcl</td>
</tr>
<tr>
<td>SemSim and embedding</td>
<td>github.com/AnacletoLAB/ [via plugin]</td>
</tr>
<tr>
<td>Visualization</td>
<td>github.com/INCATools/obographviz</td>
</tr>
</tbody>
</table>
OAK Reasoning Strategy

Classification reasoning: prior to release

- Not a priority area for OAK (handled by robot)
  - Is possible with OAK robot plugin
- Future
  - Whelk.rs + py-horned-owl?
  - owlery?

“Graph walking” reasoning

- Use relation-graph OR
- Iteratively walk graphs
Acknowledgments

Charles Tapley-Hoyt
Patrick Kalita
Harshad Hegde
Nico Matentzoglu

Harry Caufield
Justin Reese
Sierra Moxon
Chris Kindermann (KGCL)
James Overton (KGCL)

+ All developers of PyPI packages we use

https://contrib.rocks/preview?repo=INCATools%2Fontology-access-kit