UNIVERSITÄT KLAGENFURT

From RDF to Logic Programming: Extracting Facts with rdflib

Palina Dubatouka

Workflow

Turtle file (.ttl)

Load the TTL file

RDF Graph (rdflib)

Use the Python library RDFLib to parse the TTL file to Graph.
It reads the TTL syntax, turns it into triples (Subject, Predicate, Object) and stores them.

Facts_definition() method

Extract all triples from the RDF graph, including RDFS and OWL relations.

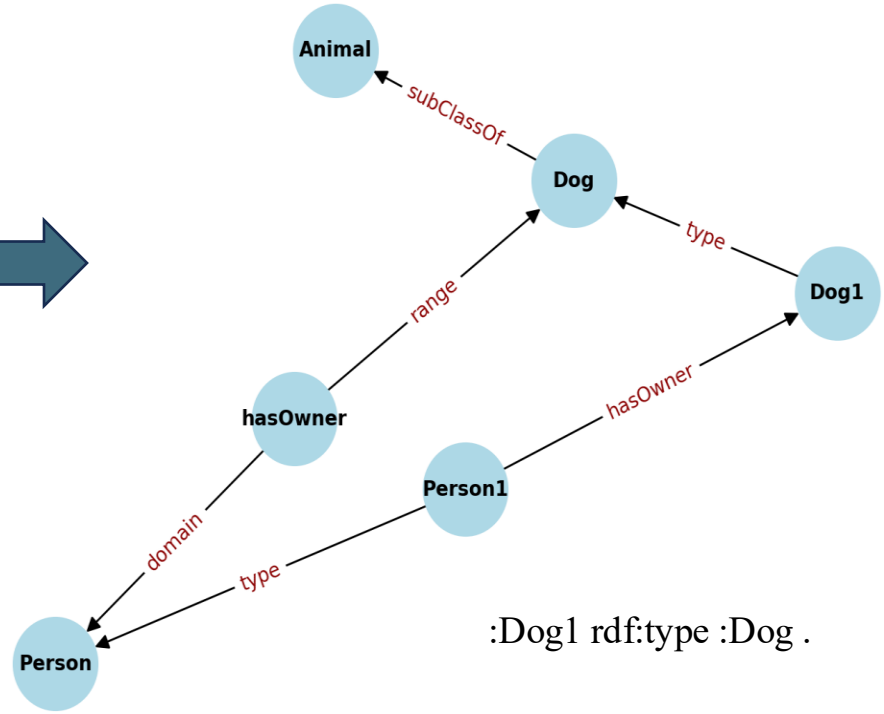
Write to file and Sort

Write each set of triples to the file
Sort according to the Subject

ASP Facts in .lp file

Save the file with ASP facts

Simple RDF Graph: Dog and its Relations



:Animal a rdfs:Class .

:Dog a rdfs:Class ;
rdfs:subClassOf :Animal .

:Person a rdfs:Class .

:Dog1 rdf:type :Dog .

:Person1 rdf:type :Person .

:hasOwner a rdf:Property ;
rdfs:domain :Person ;
rdfs:range :Dog .

:Person1 :hasOwner :Dog1 .

TTL to ASP Example

(Subject) (predicate) (Object)

```
mbco:gives_modelling_value_for a owl:ObjectProperty ;
    rdfs:label "gives modelling value for" ;
    rdfs:domain mbco:computational_modelling_method ;
    rdfs:range mbco:physical_quantity ;
    rdfs:subPropertyOf mbco:gives_value_for .
```

Transform into ASP facts

```
entity("gives_modelling_value_for","ObjectProperty").
label("gives_modelling_value_for","gives modelling value for").
domain("gives_modelling_value_for","computational_modelling_method").
range("gives_modelling_value_for","physical_quantity").
subPropertyOf("gives_modelling_value_for","gives_value_for").
```

Other methods:

- **qname_atom (g, term: URIRef)** method turns a *URIRef* into a short local part (ex:Dog → "Dog").
- **bnode_only_types(g: Graph, b: BNode)** method is used for *Bnode;* if a blank node only has rdf:type triples (no other predicates), return its list of classes (ex:GITT_TEST mbco:has_participant [a mbco:agent] ;)
- **esc(s: str) -> str** method escapes special chars in literal strings (\", \\). Used only for *Literal*

RDF/RDFS/OWL → ASP facts

RDFLib (python library)	Example (Turtle)	ASP fact
RDF.type	:Dog1 **rdf:type/a** :Dog	entity("Dog1", "Dog").
RDFS.subClassOf	:Dog **rdfs:subClassOf** :Animal	subClassOf("Dog", "Animal").
RDFS.subPropertyOf	:hasRescuedDog **rdfs:subPropertyOf** :hasOwner	subPropertyOf("hasRescuedDog", "hasOwner").
RDFS.comment	:Dog **rdfs:comment** ”Animal"	comment("Dog", "Animal").
RDFS.label	:Dog **rdfs:label** "Dog"	label("Dog", "Dog").
RDFS.domain	:hasOwner **rdfs:domain** :Person	domain("hasOwner", "Person").
RDFS.range	:hasOwner **rdfs:range** :Dog	range("hasOwner", "Dog").
OWL:oneOf	:Color **owl:oneOf** (:White :Brown)	oneOf("Color", "White"). oneOf("Color", "Brown").

RDF/RDFS/OWL → ASP facts

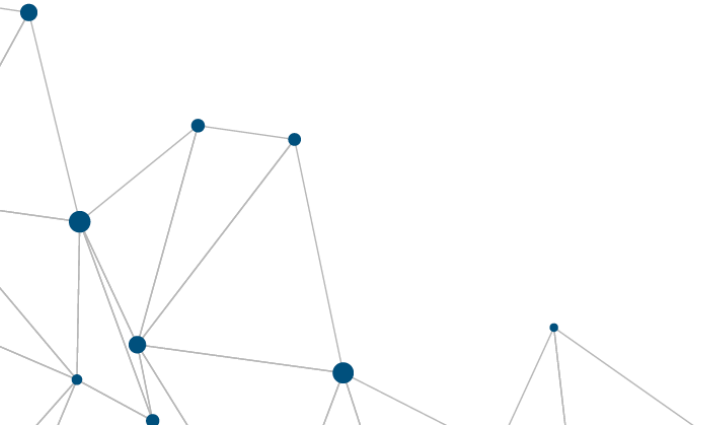
RDFLib (python library)	Example (Turtle)	ASP fact
OWL.equivalentClass	:PetDog **owl:equivalentClass** :DomesticDog .	equivalentClass("PetDog", "DomesticDog").
OWL.differentFrom	:Dog1 **owl:differentFrom** :Dog2 .	differentFrom("Dog1", "Dog2").
OWL.sameAs	:Rex **owl:sameAs** :Dog1 .	sameAs("Rex", "Dog1").
OWL.inverseOf	:hasOwner **owl:inverseOf** :ownsDog .	inverseOf("hasOwner", "ownsDog").
OWL.disjointWith	:Dog **owl:disjointWith** :Cat .	disjointWith("Dog", "Cat").
OWL.disjointUnionOf	:Animal **owl:disjointUnionOf** (:Dog :Cat :Bird) .	disjointUnionOf("Animal", "Dog"). disjointUnionOf("Animal", "Cat"). disjointUnionOf("Animal", "Bird").
Property	x:GITT_TEST mbco:has_participant ex:POTENTIOSTAT	property("GITT_TEST","has_participant","POTENTIOSTAT").
Property	x:GITT_TEST mbco:has_participant [a mbco:agent]	property("GITT_TEST","has_participant","agent",1,1).

RDF/RDFS/OWL → ASP facts

RDFLib (python library)	Example (Turtle)	ASP fact
Range + unionOf	mbco:has_participant rdfs:range [owl:unionOf (mbco:agent mbco:software)] .	range("has_participant","unionOf","agent"). range("has_participant","unionOf","software").
OWL.Restriction someValuesFrom	rdfs:subClassOf [a **owl:Restriction** ; owl:onProperty :has_setting ; owl:someValuesFrom owl:Thing] .	Restriction_some(”process”, ”has_setting”, ”Thing”)
OWL.Restriction. allValuesFrom	rdfs:subClassOf [a **owl:Restriction** ; owl:onProperty :has_setting ; owl:allValuesFrom owl:Thing] .	Restriction_all(”process”, ”has_setting”, ”Thing”)

Thank You!

Any Questions?



UNIVERSITÄT KLAGENFURT