Table of Contents

CBI – Cognescent Business Intelligence

Scope

Basically the aim of the project is to enable multiple source gathering of 'plain' RDF to the maximum level of inference possible. With 'plain' I mean RDF without schema information, only SPO triples and no previous knowledge of external sources or ontologies.

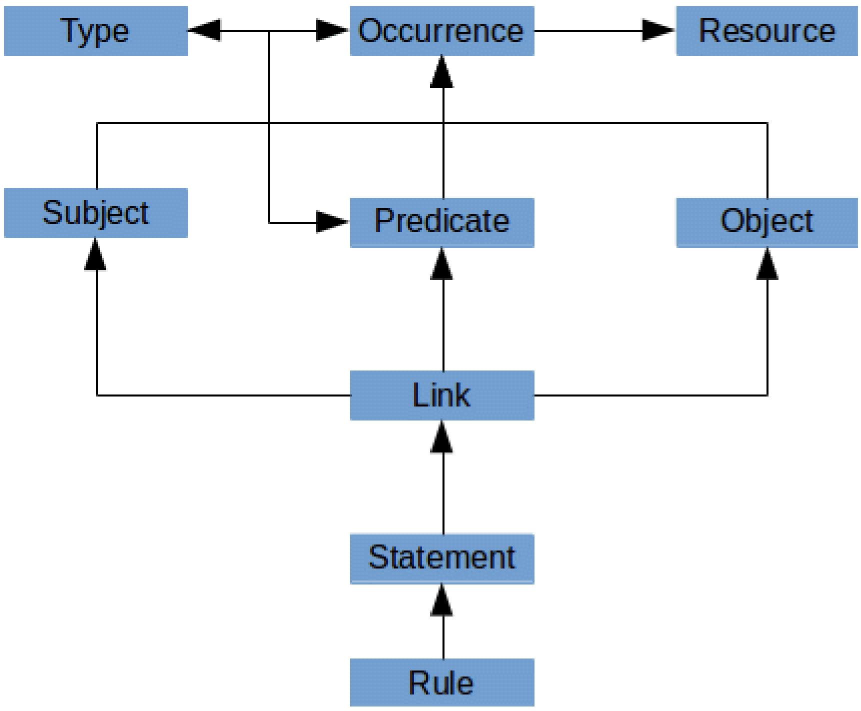
For this to be accomplished an arrangement of concepts is made which allows, for example, infer the type of a subject only by the triples in which it occurs (Occurrences are a concept which represents Subjects, Predicates and Objects). Occurrences refers to a Resource, which is an abstract representation of an URI in the source RDF but also has capabilities to be aligned and merged.

The process of 'feeding' RDF to a running instance of the service should allow, via an internal indexing engine, to post RDF fragments to a specific endpoint and have an 'augmented' response with all the instance knowledge regarding that 'query'. If the 'query' response is empty, the service 'remembers' in a conversational way it's contents so they become part of the knowledge base and are available for further inference.

The intention is to enable as much 'algorithmic' reasoning as possible and merge / align ontologies from diverse sources.

Core Concepts

The following diagram (pseudo - UML) depicts some of the core concepts of the service:



Resource

Any Subject, Predicate, Object of an statement (Link in the diagram) is an instance of an Occurrence (of a Resource). A Resource can have many Occurrences. Beyond the source URI of the RDF statement which gave birth to the Resource (sourceURI) the Resource has the following metadata:

- **classID:** This is a numeric ID assigned to all Resources at creation time coming from a prime number sequence unique to all Resources. Kind of primary key.
- **instanceID:** This is a numeric identifier which gets 'augmented' with each Resource Occurrence as explained below. Useful for algorithmic inference.
- **normalizedURI:** Once this Resource is merged/aligned, it gets a new URI which is network retrieveable (by the service endpoint).
- **mergedClassIDs, mergedURIs:** Upon merging/alignement hold previous Resources history.

Link

A link is the representation of a statement coming from an RDF source.

Occurrence (SPO)

When a Resource occurs into a Link, its instanceID gets augmented by the following rules:

- **Subject:** Its instanceID is updated by the product of it by the instanceID of the Predicate of the Link.
- **Predicate:** Its instanceID is updated by the product of it by the instanceID of the Subject and the instanceID of the Object of the Link.
- **Object:** Its instanceID is updated by the product of it by the instanceID of the Predicate of the Link.

This makes a Resource 'remembers' where it has occurred and is useful for merging / algebraic inference. The first time a Resource appears its instanceID is its classID.

Type

Roughly, Type Inference can be performed regarding which Predicates comes out from a set of Subjects. If a set of Subjects share the same set of Predicates the resulting set could be regarded as an 'inferred' same type.

Inference

Basic syllogistic Inference

Being Links the representation of syllogistic statements of the kind:

(Socrates, kind, Human)
(Human, life, Mortal)
(Socrates, kind x life, Mortal)

And being this labels for the parts of the reasoning:

From = (a, rel1, b);
Rel = (b, rel2, c);
Dest = (a, rel1 x rel2, c);

Then, being '1' the identity relationship, we can infer:

From → (a, rel1, b) = (b, rel1 x rel2, c) x (b, 1, b) / (b, rel2, c);
Rel → (b, rel2, c) = (a, rel1 x rel2, c) x (b, 1, b) / (a, rel2, b);
Dest → (a, rel1 x rel2, c) = (a, rel1, b) x (b, rel2, c) / (b, 1, b);

And the Identity resolution formula:

(a, 1, c) = (a, rel1, b) x (b, rel2, c) / (b, rel1 x rel2, b);

The conclusion of the syllogism statements can be used for further reasoning using rules.

Algorithmic Inference

Mappings

Mapping are semantic hashing of Resources and can be used for similarity ranking and index lookups or to build further structures, Templates, whose purpose is to make information available for better navigation and presentation. One gets a Mapping from a Resource, existing or created ad-hoc for criteria and it gets populated with its metadata.

The metadata part of a Mapping takes care of extracting the predicates in and out the Resource has. Predicates out determine the Type of the Resource and Predicates in determine the Roles of the Resource in the template. The importance of the Role is that it, in turn, determines the Metaclasses of the Template. Given a Role:

(x, employee, Resource) : Employee Role.

where Resource is the Resource of the Mapping, a Metaclass is created for the Mapping having the properties of Resource as an employee (position, salary).

The Mapping has a bidirectional connection between Metaclasses and Roles via its State and can, for a given Role, determine which properties are bound to the Resource via the Metaclass. Metaclasses are created first for no-role situations (all Resource out predicates) and later when parsing roles.

Templates

Templates holds sets of Links in Subject, Predicate, Object arrays. It arranges Classes it knows about and Properties of Subjects of which also knows. It purpose is to simplify presentation and navigation handling and also to provide means of template like transforms features.

Transformations, Merge and Alignement

Having one source template, a Transform is a set of Mappings (SPO → SPO) which, applied to source Template, results in a new Template. Transform SPOs are Mappings applied to source Resources and having its similarity results as output.

Formal Concept Analysis (FCA)

This section is regarding ontology merge and data sources alignment.
Given two RDF graphs two FCA relations could be built as having each of them in the object side the subjects of the two graph's statements and in the properties side the objects of the two graph's statements then having relation A and relation B from each graph. The graphs could be built by using graphs properties as axis and Subjects as objects and Objects as properties.

If there exist in relation A an situation such as this object, property pairs apply:
(a, 1); (a, 2); (a, 3) and (b, 2); (b, 3)

And in relation B exists:
(x, 4); (x, 5); (x, 6) and (y, 5); (y, 6)

There is a probability of a being similar to x and b being similar to y.

Semiotic framework

Signs, Concepts and Objects can be derived from Occurrences, Types and Resources. This information could be valuable in the case of ontology merging and identity resolution in contexts.

Datasources

Plain RDF, Relational, OLAP, CSV, Spreadsheets and any possible connectors (JSON for example) as sources of structured data are candidates and some of them have been tested using as an intermediate layer produced XML RDF.