A Roadmap for navigating the Life Sciences Linked Open Data Cloud

Ali Hasnain[1], Syeda Sana e Zainab[1], Maulik R. Kamdar[1], Qaiser Mehmood[1], Claude N. Warren, Jr[2], Qurratal Ain Fatimah, Helena F. Deus[3], Muntazir Mehdi[1], and Stefan Decker[1]

[1] Insight Center for Data Analytics, National University of Ireland, Galway
`firstname.lastname@insight-centre.org`
[2] Xenei.com `claude@xenei.com`
[3] Foundation Medicine Inc. Cambridge, MA `hdeus@foundationmedicine.com`

Abstract. Multiple datasets that add high value to biomedical research have been exposed on the web as a part of the Life Sciences Linked Open Data (LSLOD) Cloud. The ability to easily navigate through these datasets is crucial for personalized medicine and the improvement of drug discovery process. However, navigating these multiple datasets is not trivial as most of these are only available as isolated SPARQL endpoints with very little vocabulary reuse. The content that is indexed through these endpoints is scarce, making the indexed dataset opaque for users. In this paper, we propose an approach for the creation of an active Linked Life Sciences Data Roadmap, a set of configurable rules which can be used to discover links (roads) between biological entities (cities) in the LSLOD cloud. We have catalogued and linked concepts and properties from 137 public SPARQL endpoints. Our Roadmap is primarily used to dynamically assemble queries retrieving data from multiple SPARQL endpoints simultaneously. We also demonstrate its use in conjunction with other tools for selective SPARQL querying, semantic annotation of experimental datasets and the visualization of the LSLOD cloud. We have evaluated the performance of our approach in terms of the time taken and entity capture. Our approach, if generalized to encompass other domains, can be used for road-mapping the entire LOD cloud.

Keywords: Linked Data (LD), SPARQL, Life Sciences (LS), Semantic Web, Query Federation

1 Introduction

A considerable portion of the Linked Open Data cloud is comprised of datasets from Life Sciences Linked Open Data (LSLOD). The significant contributors includes the Bio2RDF project[1], Linked Life Data[2], Neurocommons[3], Health care and Life Sciences knowledge base[4] (HCLS Kb) and the W3C HCLSIG Linking

[1] `http://bio2rdf.org/` (l.a.: 2014-03-31)
[2] `http://linkedlifedata.com/` (l.a.: 2014-07-16)
[3] `http://neurocommons.org/page/Main_Page` (l.a.: 2014-07-16)
[4] `http://www.w3.org/TR/hcls-kb/` (l.a.: 2014-07-16)

Open Drug Data (LODD) effort[5]. The deluge of biomedical data in the last few years, partially caused by the advent of high-throughput gene sequencing technologies, has been a primary motivation for these efforts. There had been a critical requirement for a single interface, either programmatic or otherwise, to access the Life Sciences (LS) data. Although publishing datasets as RDF is a necessary step towards unified querying of biological datasets, it is not sufficient to retrieve meaningful information due to data being heterogeneously available at different endpoints [14,2]. Despite the popularity and availability of bio-ontologies through ontology registry services[6], it is still very common for semantic web experts to publish LS datasets without reusing vocabularies and terminologies. The popularity of bio-ontologies has also led to the use of overlapping standards and terminologies, which in turn has led to a low adoption of standards [14]. For example, the exact term "Drug" is matched in 38 bioportal ontologies[7] - it is not clear which of these should be chosen for publishing LD.

In the LS domain, LD is extremely heterogeneous and dynamic [16,8]; also there is a recurrent need for *ad hoc* integration of novel experimental datasets due to the speed at which technologies for data capturing in this domain are evolving. As such, integrative solutions increasingly rely on federation of queries [6,7,5]. With the standardization of SPARQL 1.1, it is now possible to assemble federated queries using the "SERVICE" keyword, already supported by multiple tool-sets (SWobjects, Fuseki and dotNetRDF). To assemble queries encompassing multiple graphs distributed over different places, it is necessary that all datasets should be query-able using the same global schema [17]. This can be achieved either by ensuring that the multiple datasets make use of the same vocabularies and ontologies, an approach previously described as *"a priori integration"* or conversely, using *"a posteriori integration"*, which makes use of mapping rules that change the topology of remote graphs to match the global schema [6] and the methodology to facilitate the latter approach is the focus of this paper. Moreover for LD to become a core technology in the LS domain, three issues need to be addressed: *i)* dynamically discover datasets containing data on biological entities (e.g. Proteins, Genes), *ii)* retrieve information about the same entities from multiple sources using different schemas, and *iii)* identify, for a given query, the highest quality data.

To address the aforementioned challenges, we introduce the notion of an active Roadmap for LS data – a representation of entities as *"cities"* and the links as the *"roads"* connecting these *"cities"*. Such a Roadmap would not only help understand which data exists in each LS SPARQL endpoint, but more importantly enable assembly of multiple source-specific federated SPARQL queries. In other words, the ability to assemble a SPARQL query that goes from **A** (e.g. a neuroreceptor) to **B** (e.g. a drug that targets that neuroreceptor), requires a Roadmap that clarifies all the possible *"roads"* or *"links"* between those two entities. Our initial exploratory analysis of several LS endpoints revealed that they

[5] `http://www.w3.org/wiki/HCLSIG/LODD` (l.a.: 2014-07-16)
[6] `http://bioportal.bioontology.org/` (l.a.: 2014-07-12)
[7] `http://bioportal.bioontology.org/search?query=Drug` (l.a.: 2014-07-12)

not only use different URIs but also different labels for similar concepts (e.g. Molecule vs Compound). Our methodology for developing the active Roadmap consisted of two steps: *i)* catalogue development, in which metadata is collected and analyzed, and *ii)* links creation and Roadmap development, which ensures that concepts and properties are properly mapped to a set of Query Elements (Qe) [19]. Hasnain et. al [9] described the Link Creation mechanism, linking approaches as well as the linking statistics and in this paper we focus primarily on the Cataloguing mechanism that facilitates linking as a second step. We assumed in this work that federated queries are assembled within a context – as such, our Roadmap relies on identifying the global schema onto which entities should be mapped. This entails the initial identification of a set of Qe[8], in the context of cancer chemoprevention[19], identified by the domain experts participating in the EU GRANATUM project[9].

The rest of this paper is organized as follows: In Section 2, we discuss the related research carried out towards integrating heterogeneous LD. In Section 3, we introduce the catalogue and link generation methodologies to build Roadmap. In Section 4, we showcase four applications of the generated Roadmap - notably a domain-specific federated query engine which reasons over the Roadmap to query the LSLOD. We evaluate the performance of Cataloguing Mechanism in terms of time taken and entity capture in Section 5.

2 Related Work

Approaches to facilitate the *"A posteriori integration"* is currently an area of active research. One approach is through the use of available schema: semantic information systems have used ontologies to represent domain-specific knowledge and enable users to select ontology terms in query assembly [13]. BLOOMS, for example, is a system for finding schema-level links between LOD datasets using the concept of ontology alignment [11], but it relies mainly on Wikipedia. Ontology alignment typically relies on starting with a single ontology, which is not available for most SPARQL endpoints in the LOD cloud and therefore could not be applied in our case. Furthermore, ontology alignment does not make use of domain rules (e.g. if two sequences are the same, they map to the same gene) nor the use of URI pattern matching for alignment – these issues had already been discussed by Hasnain et. al [9]. Other approaches such as the VoID [1] and the SILK Framework [18] enable the identification of rules for link creation, but require extensive knowledge of the data prior to links creation. Query federation approaches have developed some techniques to meet the requirements of efficient query computation in the distributed environment. FedX [15], a project which extends the Sesame Framework [3] with a federation layer, enables efficient query processing on distributed LOD sources by relying on the assembly of a catalogue of SPARQL endpoints but does not use domain rules for links creation. Our approach for link creation towards Roadmap development is a combination of the several linking approaches as already explained by Hasnain et. al [9]: *i)* similarly

[8] `http://srvgal78.deri.ie/RoadMapEvaluation/#Query_Elements`(l.a.: 2014-07-19)
[9] `http://www.granatum.org`(l.a.: 2014-07-05)

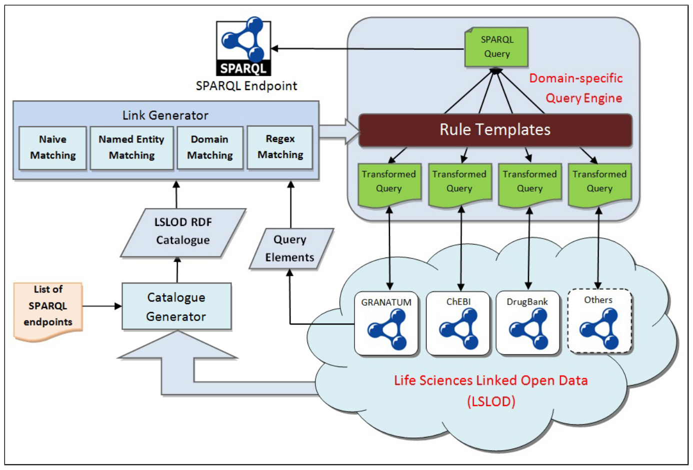


Fig. 1: Roadmap and Query Engine Architecture

to ontology alignment, we make use of label matching to discover concepts in LOD that should be mapped to a set of *Qe*, *ii)* we create *"bags of words"* for discovery of schema-level links similar to the approach taken by BLOOMS, and *iii)* as in SILK, we create domain rules that enable the discovery of links.

3 Methodology

We developed an active Roadmap for navigating the LSLOD cloud. Our methodology consists of two stages namely catalogue generation and link generation. Data was retrieved from 137 public SPARQL endpoints[10] and organized in an RDF document - the LSLOD Catalogue. The list of SPARQL endpoints was captured from publicly available Bio2RDF datasets and by searching for datasets in CKAN[11] tagged *"life science"* or *"healthcare"*.

3.1 Methodology for Catalogue Development

Connecting the different concepts i.e. making links between them, is an ultimate goal of this research to facilitate the navigation across the LSLOD Cloud. As an example from a drug discovery scenario, it is often necessary to find the links between *"cancer chemopreventive agent"* and *"publication"*. For enabling this, a preliminary analysis of multiple SPARQL Endpoints containing data from both the Life Sciences and the Health care domains was undertaken. A semi-automated method was devised to retrieve all classes (concepts) and associated properties (attributes) available through any particular endpoint by probing data instances. The workflow definition for probing instances through endpoint analysis using the W3C SPARQL algebra notation[12] is as follows:

[10] `http://goo.gl/ZLbLzq`
[11] `http://wiki.ckan.org/Main_Page` (l.a.: 2014-05-05)
[12] `http://www.hpl.hp.com/techreports/2005/HPL-2005-170.pdf` (l.a.: 2014-07-30)

1. For every SPARQL endpoint S_i, find the distinct Classes $C(S_i)$:
$$C(S_i) = \mathit{Distinct}\ (\mathit{Project}\ (?\mathit{class}\ (\mathit{toList}\ (\mathit{BGP}\ (\mathit{triple}\ [\ ]\ a\ ?\mathit{class}\ ))))) \tag{1}$$
2. Collect the Instances for each Class $C_j(S_i)$:
$$I_i : C_j(S_i) = \mathit{Slice}\ (\mathit{Project}\ (?I\ (\mathit{toList}\ (\mathit{BGP}\ (\mathit{triple}\ ?a\ a\ < C_j(S_i) >\ )))),\ \mathit{rand}()) \tag{2}$$
3. Retrieve the Predicate/Objects pairs for each $Ii : C_j(S_i)$:
$$I_i(P, O) = \mathit{Distinct}\ (\mathit{Project}\ (?p,\ ?o\ (\mathit{toList}\ (\mathit{BGP}\ (\mathit{triple}\ < I_i : C_j(S_i) > ?p\ ?o\ )))) \tag{3}$$
4. Assign Class $C_j(S_i)$ as domain of the Property P_k :
$$\mathit{Domain}(P_k) = C_j(S_i) \tag{4}$$
5. Retrieve Object type (O_T) and assign as a range of the Property P_k :
$$\mathit{Range}(P_k) = O_T; O_T = \begin{cases} \mathit{rdf} : \mathit{Literal} & \text{if } (O_k\ \mathit{is}\ \mathit{String}) \\ \mathit{dc} : \mathit{Image} & \text{if } (O_k\ \mathit{is}\ \mathit{Image}) \\ \mathit{dc} : \mathit{InteractiveResource} & \text{if } (O_k\ \mathit{is}\ \mathit{URL}) \\ \mathit{Project}\ (?R\ (\mathit{toList}\ (\mathit{BGP} & \\ (\mathit{triple}\ < O_k >\ \mathit{rdf} : \mathit{type}\ ?R))) & \text{if } (O_k\ \mathit{is}\ \mathit{IRI}) \end{cases} \tag{5}$$

It is worth noting that step 2 is heuristic – performing step 3 on a list of random instances is only necessary to avoid query timeout as the alternative (`triple [ ] ?p ?o`) would generally retrieve too many results. Step 5 effectively creates links between two entities ($C_j(S_i)$ and the Object Type O_T), but only when the object of the triples (O_k) retrieved in step 3 are URIs. We found that the content-type of properties can take any of the following formats:

1. Literal (i.e non-URI values e.g: *"Calcium Binds Troponin-C"*)
2. Non-Literal; these can further be divided into one of following types:
   (a) URL (e.g.: `<http://www.ncbi.nlm.nih.gov/pubmed/1002129>`) which is not equivalent to a URI because is cannot retrieve structured data.
   (b) Images (e.g: `<http://www.genome.jp/Fig/drug/D00001.gif>`)
   (c) URI (e.g.: `<http://bio2rdf.org/pubchem:569483>`); the most common types of URI formats that we have discovered were:
      i. Bio2RDF URIs (e.g.: `<http://bio2rdf.org/gi:23753>`)
      ii. DBpedia URIs (e.g.: `<http://dbpedia.org/resource/Ontotext>`)
      iii. Freie Universität Berlin URIs e.g.: `<http://www4.wiwiss.fu-berlin.de/drugbank/resource/drugs/DB00339>`
      iv. Other URIs (e.g.: `<http://purl.org/ontology/bibo/Journal>`)

RDFS, Dublin Core[13] and VoID[14] vocabularies were used for representing the data in the LSLOD catalogue. A slice of the catalogue is presented as follows[15]:

[13] `http://dublincore.org/documents/dcmi-terms/` (l.a.: 2014-07-12)
[14] `http://vocab.deri.ie/void` (l.a.: 2014-07-12)
[15] In this paper, we omit URI prefixes for brevity. All prefixes can be looked up at `http://prefix.cc/` (l.a.: 2014-07-31)

Listing 1: An Extract from the LSLOD Catalogue for KEGG dataset

```
<http://kegg.bio2rdf.org/sparql> a void:Dataset ;
void:class <http://bio2rdf.org/ns/kegg#Enzyme> ;
void:sparqlEndpoint <http://kegg.bio2rdf.org/sparql>,
<http://s4.semanticscience.org:12014/sparql> .
<http://bio2rdf.org/ns/kegg#Enzyme> rdfs:label "Enzyme";
void:exampleResource <http://bio2rdf.org/ec:3.2.1.161>.
<http://bio2rdf.org/ns/kegg#xSubstrate> a rdf:Property;
rdfs:label "#xSubstrate" ;
voidext:domain <http://bio2rdf.org/ns/kegg#Enzyme> ;
voidext:range  <http://bio2rdf.org/kegg_resource:Compound>.
<http://bio2rdf.org/kegg_resource:Compound>
void:exampleResource <http://bio2rdf.org/cpd:C00001>;
void:uriRegexPattern "^http://bio2rdf\\.org/cpd:.*" ;
voidext:sourceIdentifier "cpd" .
```

The RDF above is an illustrative example of a portion of the catalogue generated for the KEGG SPARQL endpoint[16]. VoID is used for describing the dataset and for linking it with the catalogue entries: the `void#Dataset` being described in this catalogue entry is "KEGG" SPARQL endpoint. In cases where SPARQL endpoints were available through mirrors (e.g. most Bio2RDF endpoints are available through Carleton Mirror URLs) or mentioned using alternative URLs (e.g. `http://s4.semanticscience.org:12014/sparql`), these references were also added as a second value for the `void#sparqlEndpoint` property. Listing 1 also includes one identified Class (`http://bio2rdf.org/ns/kegg#Enzyme`),and one property using that class as a domain (`http://bio2rdf.org/ns/kegg#xSubstrate`). Classes are linked to datasets using the `void#class`property; the labels were collected usually from parsing the last portion of the URI and probed instances were also recorded (`http://bio2rdf.org/ec:3.2.1.161`)as values for `void#example Resource`. Properties (`http://bio2rdf.org/ns/kegg#xSubstrate`) collected by our algorithm (steps 3,4,5) were classified as `rdfs:property`.

When the object of a predicate/object pair is of type URI, (e.g. as for KEGG shown in Listing 1) the algorithm attempts to perform link traversal in order to determine its object type (O_T). In most cases, however, the URI was not dereferenceable and in such cases an alternative method relies on querying the SPARQL endpoint for the specific "type" of the instance. In example above, dereferencing the object URI `<http://bio2rdf.org/cpd:C00001>` resulted in class `<http://bio2rdf.org/kegg_resource:Compound>`. We call this as a "range class" used as the range of the property `<http://bio2rdf.org/ns/kegg#xSubstrate>`. Actual object URI `<http://bio2rdf.org/cpd:C00001>` is classified as `void#exampleResource` of `<http://bio2rdf.org/kegg_resource:Compound>`and the URI regular expression pattern is recorded under `void#uriRegexPattern`. We found that, in many cases, the `\sourceIdentifier"` or the identifier that appears before the ":" symbol in case of many URIs could be used for discovering the appropriate type for the non-dereferenceable URI when none was provided.

[16] `http://kegg.bio2rdf.org/sparql` (l.a.: 2014-02-01)

Although this is not a standardised method, we found it to be useful in mapping classes nonetheless. For non-dereferenceable URIs with no actual class as O_T (termed Orphan URIs), a new O_T is created using UUID, which is classified as `voidext#OrphanClass` (Listing 2).

Listing 2: Orphan Classes captured in Catalogue

```
<http://bio2rdf.org/ns/kegg#xSubstrate> a rdf:Property;
voidext:domain <http://bio2rdf.org/ns/kegg#Reaction> ;
voidext:range roadmap:CLASS2c2ab5b75a454f678a9056dfc1d1214.
roadmap:CLASS2c2ab5b75a454f678a9056dfc1d1214
a voidext:OrphanClass;
void:exampleResource <http://bio2rdf.org/cpd:c00890> ;
void:uriRegexPattern "http://bio2rdf\\.org/cpd:*" ;
voidext:sourceIdentifier "cpd" .
```

3.2 Methodology for Link Generation

During this phase `subClassOf` and `subPropertyOf` links were created amongst different concepts and properties to facilitate *"a posteriori integration"*. The creation of links between identified entities (both chemical and biological) is not only useful for entity identification, but also for discovery of new associations such as protein/drug, drug/drug or protein/protein interactions that may not be obvious by analyzing datasets individually. A link is a property of an entity that takes URI as a value. The following RDF statement is both a property of the chemoprevention agent *acetophenone* as well as a link between that chemoprevention agent and a publication:`<pubmed_id:18991637>dc:hasPart"acetophenone"`. Leveraging the class descriptions and its properties in the LSLOD catalogue, links were created (discussed previously in [9]) using several approaches: *i)* Naïve Matching/ Syntactic Matching/ Label Matching, *ii)* Named Entity Matching, *iii)* Domain dependent/ unique identifier Matching, and *iv)* Regex Matching.

Regular Expression Matching Regular expression matching can be considered as a special case of "Naïve Matching". The regular expressions for all those URIs that may or may not be dereferenced (Orphan URIs) were captured during catalogue generation phase. By looking to the similar regular expressions it can be concluded that two distinct URIs belong to the same class. Considering the same regular expressions of instances of Orphan and non-Orphan URIs, an Orphan URI can safely be linked with a non-Orphan URIs.

4 Roadmap Applications

The Roadmap is exposed as a SPARQL endpoint[17] and relevant information is also documented[18]. As of 31st May 2014, the Roadmap consists of 263731 triples representing 1861 distinct classes, 3299 distinct properties and 13027 distinct Orphan Classes catalogued from 137 public SPARQL endpoints.

[17] `http://srvgal78.deri.ie:8006/graph/Roadmap` (l.a.: 2014-07-31)

[18] Roadmap Homepage: `https://code.google.com/p/life-science-roadmap/`

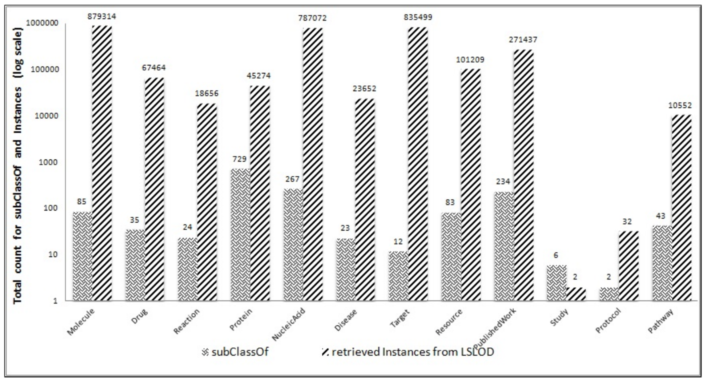


Fig. 2: Number of retrieved Instances and Subclasses linked to any *Qe*

4.1 Domain-specific Query Engine

Fundamentally, the Domain-specific Query Engine (DSQE) is a SPARQL query engine that transforms the expressions from one vocabulary into those represented using vocabularies of known SPARQL endpoints and combines those expressions into a single query using SPARQL "SERVICE" calls. The engine executes the resulting statement and returns the results to the user (Fig. 1). DSQE is implemented on top of the Apache Jena query engine and extends it by intercepting and rewriting the SPARQL algebra[19]. Most of the algebra is unmodified and we concentrate on the base graph patterns (BGP) which are effectively the triples found in the original SPARQL query.

DSQE comprises of two major components: the SPARQL Algebra rewriter and the Roadmap. The algebra rewriter examines each segment of the BGP triples and attempts to expand the terms based on the vocabulary mapping into terms of the endpoint graphs and stores the result for each. Hence, it effectively builds the BGPs for all known graphs in parallel. In the final stage, the rewriter wraps the BGPs with SERVICE calls, using the Roadmap to determine the "relavent" endpoint, and unions them together along with the original BGP. The result is passed back into the standard Jena query process and the execution continues normally. Each endpoint is therefore accessed via a SERVICE call and the relevant underlying data is incorporated in the result. Early implementations [6] have shown such algebraic rewrite to be functional and efficient. Our enhancement was two fold - *i)* Based on the presence of OWL2 `hasKey` properties, we added the ability to identify similar subjects with different URIs, *ii)* we added the ability to rewrite generic queries for different endpoints.

Identifying Subjects with Different URIs For a given query:
`SELECT ?p ?o WHERE { ?x a <T>; ?p ?o }` and two endpoints E1 and E2 with synonymous topics T1 and T2 for T and K1 and K2 synonyms for K. We want to expand the query as shown in Listing 3. To ensure that there exists a triple that matches `?alias <K> ?key` in the local graph, we use a temporary graph that

[19] `http://www.w3.org/TR/sparql11-query/#sparqlAlgebra` (l.a.: 2014-07-30)

only exists for the duration of the query and is merged with the normal local graph during execution. Hence, when DSQE executes the query we retrieve the necessary information from the remote SPARQL endpoints to merge the results together even if they use different values for the Subject.

Listing 3: SPARQL Construct

```
SELECT ?p ?o WHERE { { SERVICE <E1> {
      SELECT (?Ap as ?p) (?Ao as ?o) (?Akey as ?key)
      WHERE { ?Ax a <T1> ; <K1> ?Akey ; ?Ap ?Ao  }  }
    FILTER ( insertKeyFilter( K, ?key ))   }
  UNION { SERVICE <E2> {
      SELECT (?Bp as ?p) (?Bo as ?o) (?Bkey as ?key)
      WHERE { ?Bx a <T2> ; <K2> ?Bkey ; ?Bp ?Bo }   }
      FILTER ( inserKeyFilter( K, ?key ))
  } ?alias <K> ?key ?p ?o  }
```

An instance[20] of the DSQE is deployed in the context of cancer chemoprevention drug discovery [10]. To facilitate the user to build federated SPARQL queries for execution against the LSLOD, the DSQE provides a 'Standard' and a 'Topic-based' query builder. The user can select a topic of interest (e.g. `Molecule`) and a list of associated Qe are automatically generated. The user can also select relevant filters (numerical and textual) through the 'Topic-based' builder. Our implementation exposes a REST API and provides a visual query system, named ReVeaLD [12], for intuitive query formulation and domain-specific uses [10]. The catalogued subclasses of few Qe and as a result the total number of distinct instances retrieved per Qe while querying using DSQE is shown in Fig. 2.

4.2 Roadmap for drug discovery

Mining LD for drug discovery can become possible once domain experts are able to discover and integrate the relevant data necessary to formulate their hypothesis [10]. For domain users, it is not always obvious where the data is stored or what are the appropriate terminologies used to retrieve/publish it. Although multiple SPARQL endpoints contain molecular information, not all of them use the same terminology, as we have shown in our LSLOD catalogue. For example, one of the most intensive uses of LD is to find links which translate between `Disease` → `Drugs` → `Protein targets` → `Pathways`. Such data is not available at a single source and new datasets relevant for such query needs to be added *ad hoc*. Our Roadmap provides a possible solution to this dilemma by enabling the dynamic discovery of the SPARQL endpoints that should be queried and the links between these concepts (Listing 4).

Listing 4: Roadmap Links between concepts relevant for drug discovery

```
?disease a :Disease ; :treatedWith ?Drug .
?Drug    a :Drug ; :interactsWith ?target .
?target  a :Target ; :involvedIn ?pathway .
?pathway a :Pathway .
```

[20] `http://srvgal78.deri.ie:8007/graph/Granatum`

Consider Qe to be any of the possible query elements and XQe to be an instance of that query element, the following can be used by a SPARQL 1.1 engine to list all possible XQe regardless of where they are stored:

$$C_i \sqsubseteq Qe \ \&\& \ XC_i : C_i \Rightarrow XQe, XC_i : Qe \tag{6}$$

Listing 5: SPARQL Construct to list all possible instances

```
CONSTRUCT { ?x a [QeRequested] } WHERE {
?SparqlEndpoint void:class [QeRequested]} UNION
{?QeMatched rdfs:subClassOf [QeRequested].}
SERVICE ?SparqlEndpoint {?x a ?QeMatched. } }
```

In Listing 5, `[QeRequested]` can be `Disease`, `Drug`, `Protein` or `Pathway`. Similarly, to discover which properties link two Qe together, the Roadmap can be used to create new graphs that map elements to those available in the chosen set of query elements. For any "linked" Qe, there is a set of incoming links (properties that use Qe as its `rdfs:domain`) and a set of outgoing links (properties that use Qe as its `rdfs:range`). We denote the collection of incoming and outgoing links from a particular concept as $C_i(IC_i, OC_i)$. When concepts from available graphs are mapped to Qe, it becomes possible to create new incoming (I_i) and outgoing (O_i) links connected to Qe. This is illustrated by the following principle:

$$C_i(IC_i,\ OC_i) \ \&\& \ C_i \sqsubseteq Qe \Rightarrow Qe(\{IC_i, IQe\},\ \{OC_i, IQe\}) \tag{7}$$

We exemplify the above principle by the following SPARQL CONSTRUCT:

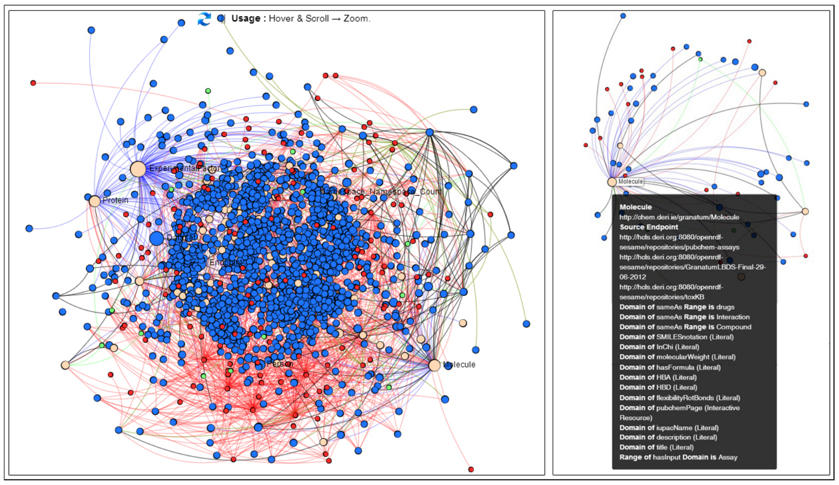
Listing 6: SPARQL Construct to create new incoming and outgoing links

```
CONSTRUCT { ?DomainQe ?PropertyDomainRange ?RangeQe } WHERE {
?PropertyDomainRange rdfs:domain [DomainQe] .
?PropertyDomainRange rdfs:range [RangeQe]
{ FILTER (?DomainQe == [DomainQe]) } UNION
{ ?DomainQe rdfs:subclassOf [DomainQe] }
{ FILTER (?RangeQe == [RangeQe]) } UNION
{ ?RangeQe rdfs:subclassOf [RangeQe] }}
```

4.3 SPARQL Endpoint selection based on availability

While probing instances through SPARQL endpoint analysis, there was clear evidence that a particular class or a property may be present at multiple SPARQL endpoints. We also noticed the problems of service disruption and perennial unavailability of some of the endpoints throughout. Our Roadmap has opened an avenue for easy, continuous and uninterrupted accessibility of data from several SPARQL endpoints, in cases where the same data may be available from multiple underlying data sources. Mondeca Labs provides a service[21] which monitors the availability status of SPARQL endpoints [4]. The status of any endpoint can be - *i)* Operating normally, *ii)* Available but problems within last 24 hours, *iii)* Service disruption, or *iv)* Still alive? Exploiting the knowledge available in our

[21] `http://labs.mondeca.com/sparqlEndpointsStatus.html` (l.a.: 2014-07-30)



Molecule
http://chem.deri.ie/granatum/Molecule
Source Endpoint
http://hcls.deri.org:8080/openrdf-sesame/repositories/pubchem-assays
http://hcls.deri.org:8080/openrdf-sesame/repositories/GranatumLBDS-Final-29-06-2012
http://hcls.deri.org:8080/openrdf-sesame/repositories/toxKB
Domain of sameAs **Range is** drugs
Domain of sameAs **Range is** Interaction
Domain of sameAs **Range is** Compound
Domain of SMILESnotation (Literal)
Domain of InChi (Literal)
Domain of molecularWeight (Literal)
Domain of hasFormula (Literal)
Domain of HBA (Literal)
Domain of HBD (Literal)
Domain of flexibilityRotBonds (Literal)
Domain of pubchemPage (Interactive Resource)
Domain of iupacName (Literal)
Domain of description (Literal)
Domain of title (Literal)
Range of hasInput **Domain is** Assay

Fig. 3: Visualizing the mapped LSLOD Cloud

Roadmap in conjunction with this service, we can direct our query to only that endpoint which is currently available, quick to respond and has similar data. In Listing 7, we present a sample SPARQL query determining the availability status of SPARQL endpoints providing data on `Protein`-similar concepts in the LSLOD.

Listing 7: Availability of SPARQL Endpoints for Protein-similar concepts

```
SELECT * WHERE { SERVICE
<http://hcls.deri.org:8080/openrdf-sesame/repositories/roadmap>
{{ ?sparqlEndpoint void:class ?proteinClass .
FILTER (?proteinClass = granatum:Protein)}
UNION { ?sparqlEndpoint void:class ?proteinClass .
    ?proteinClass rdfs:subClassOf granatum:Protein . }}
SERVICE <http://labs.mondeca.com/endpoint/ends> {
    ?dataset void:sparqlEndpoint ?sparqlEndpoint .
        ?dataset ends:status ?status .
        ?status dcterms:date ?statusDate .
        ?status ends:statusIsAvailable ?isAvailable . }}
ORDER BY DESC(?statusDate)
```

4.4 Visualization of the mapped LSLOD Cloud

A Visualization interface[22] is also developed to enable the domain users for intuitively navigating the LSLOD Cloud using the generated Roadmap (Fig. 3). The linked concepts and literals are displayed as nodes arranged in a force-directed concept map representation, as previously introduced in [12]. The *Qe* used for linking are displayed as *Light Brown*-colored nodes, whereas catalogued concepts (*Blue*-colored nodes) are linked to these *Qe* using the `voidext:subClassOf`.

[22] `http://srvgal78.deri.ie/roadmapViz/` (l.a.: 2014-07-31)

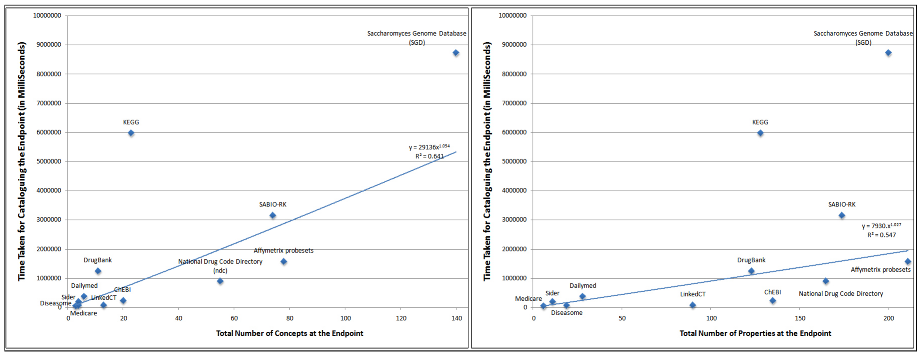


Fig. 4: Time taken to catalogue 12 SPARQL endpoints

Properties with the content-types `rdf:Literal` and `dc:InteractiveResource` are shown as *Red* and *Green*-colored nodes respectively, whereas URI-type properties are represented as *Black*-colored edges between the domain and the range concepts. The final result is a densely-clustered network graph representing the mapped LSLOD. Hovering over any particular node reduces this graph to display only the first-level associations. An information box is also displayed (Fig. 3), which provides additional details to the user regarding the source SPARQL endpoint from which the concept was catalogued, its super classes and the list of properties for which the selected concept may act as a domain/range, along with the name of the associated node or the O_T (Literal, Interactive Resource).

4.5 Google Refine Tool

Drug compounds and molecular data are available in machine-readable formats from many isolated SPARQL endpoints. In order to harvest the benefits of their availability, an extension[23] to the popular Google Refine[24] tool was devised for providing researchers with an intuitive, integrative interface where they can use the Roadmap to semantically annotate their experimental data. Researchers load a list of molecules into the application and link them to URIs. The extension is able to make use of the URIs in conjunction with the Roadmap to collect all possible literal properties that are associated with the "Molecule" instances and help users enhance their datasets with extra molecular properties.

5 Evaluation

We have previously evaluated the performance of our Link Generation methodology by comparing it against the popular linking approaches [9].

5.1 Experimental Setup

We evaluated the performance of our catalogue generation methodology (shown in Section 3.1). The aim of this evaluation was to determine if we could catalogue any SPARQL endpoint and link it to the Roadmap in a desirable amount

[23] `http://goo.gl/S809N2` (l.a.: 2014-07-31)
[24] `http://refine.deri.ie` (l.a.: 2014-07-31)

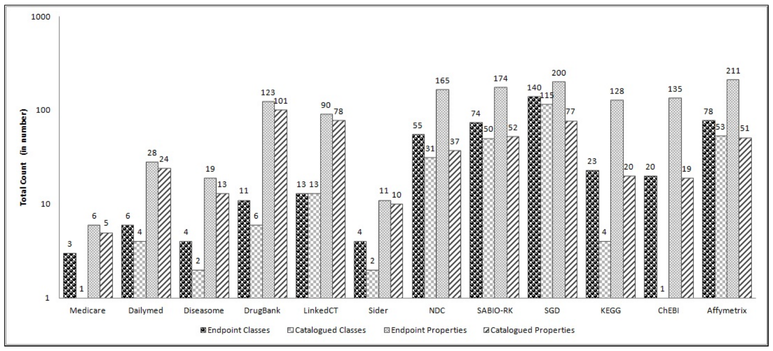


Fig. 5: Comparative plot of available versus catalogued Query Elements

of time with a satisfactory percentage capture (catalogued elements/total elements). We proceeded by recording the times taken to probe instances through endpoint analysis of 12 different endpoints whose underlying data sources were considered relevant for drug discovery - Medicare, Dailymed, Diseasome, DrugBank, LinkedCT, Sider, National Drug Code Directory (NDC), SABIO-RK, Saccharomyces Genome Database (SGD), KEGG, ChEBI and Affymetrix probesets. The cataloguing experiments were carried out on a standard machine with 1.60Ghz processor, 8GB RAM using a 10Mbps internet connection. We recorded the total available concepts and properties at each SPARQL endpoint as well as those actually catalogued in our Roadmap (Fig. 5). Total number of triples exposed at each of these SPARQL endpoints and the total time taken for cataloguing was also recorded. We attempted to select those SPARQL endpoints which have a better latency for this evaluation, as the availability and the uptime of the SPARQL endpoint is an important factor for cataloguing. Best fit regression models were then calculated.

5.2 Evaluation Results

As shown in Fig. 4, our methodology took less than 1000000 milliseconds (<16 minutes) to catalogue seven of the SPARQL endpoints, and a gradual rise with the increase in the number of available concepts and properties. We obtained two power regression models ($T = 29206 * C_n^{1.113}$ and $T = 7930 * P_n^{1.027}$) to help extrapolate time taken to catalogue any SPARQL endpoint with a fixed set of available concepts (C_n) and properties (P_n), with R^2 values of 0.641 and 0.547 respectively. Using these models and knowing the total number of available concepts/properties, a developer could determine the approximate time (ms) as a vector combination. A comparative plot of the total number of concepts and properties available at the endpoints against those catalogued by our methodology was also prepared. We obtained a >50% concept capture for 9 endpoints, and a >50% property capture for 6 endpoints. KEGG and SGD endpoints had taken an abnormally large amount of time for cataloguing than the trendline. The reason for this may include endpoint timeouts or network delays.

6 Discussion

There is great potential in using semantic web and LD technologies for drug discovery. However, in most cases, it is not possible to predict *a priori* where the relevant data is available and its representation. An additional consequence of the deluge of data in life sciences in the past decade is that datasets are too large (in the order of terabytes) to be made available through a single instance of a triple store, and many of the source providers are now enabling native SPARQL endpoints. Sometimes only a fraction of the dataset is necessary to answer a particular question – the decision on "which" fraction is relevant will vary on the context of the query. In this paper we describe the concept and methodology for devising an active Linked Life Sciences Data Roadmap. Our methodology relies on systematically issuing queries on various life sciences SPARQL endpoints and collecting its results in an approach that would otherwise have to be encoded manually by domain experts or those interested in making use of the web of data for answering meaningful biological questions. In an effort to explore how often concepts are reused, we define a methodology that maps concepts to a set of Qe, which were defined by domain experts as relevant in a context of drug discovery.

6.1 Catalogue Development and Links Creation

The number of classes per endpoint varied from a single class to a few thousands. Our initial exploration of the LSLOD revealed that only 15% of classes are reused. However, this was not the case for properties, of which 48.5% are reused. Most of the properties found were domain independent (e.g. `type`, `xref`, `sameAs`, `comment`, `seeAlso`); however, these are not relevant for the Roadmap as they cannot increase the richness of information content. These properties can be more easily resolved through the concepts that are used as domain/range. In class matching, most orphan classes, which were created in cases where object URI were non-dereferenceable, could be mapped to Qe through matching URI regular expression patterns and source identifiers. From these results, we found that maintaining consistency of the catalogue even when the URIs were non-dereferenceable is critical as the merging of data with other sources enable the classification of the instances and identification of "roads" between different SPARQL endpoints. We faced multiple challenges during catalogue development which can hinder the applicability of our approach:

- Some endpoints return timeout errors when a simple query (`SELECT DISTINCT ?Concept WHERE {[ ] a ?Concept}`) is issued.
- Some endpoints have high downtime and cannot be generally relied.
- Many endpoints provide non-deferenceable URI and some derefenceable URI do not provide a "type" for the instance.

Nevertheless, we still found the Roadmap approach highly applicable for solving complex biological problems in drug discovery [10]. Although a very low percentage of linking becomes possible through naïve matching or manual/domain matching, the quality of links created are highly trusted [9]. For Orphan classes 34.9% of classes were linked by matching the URI regex patterns. It is also worth noticing that 23% of identified classes, and 56.2% of the properties remained unlinked, either because they are out of scope or cannot match any *Qe*. This means that the quality as well as the quantity of links created is highly dependent on the set of *Qe* used: if the *Qe* `gr:Gene` is available, `kegg:Gene` would be mapped to it as opposed to the current case where it is mapped to `gr:nucleicAcid`. In such cases, additions to the *Qe* could be suggested. It is worth noting that these *Qe* were created to fit the drug discovery scenario, and can be replaced in alternative contexts e.g *"Protein-Protein Interaction"*. Changing the *Qe* will result in different Roadmaps. The aim of the LSLOD Roadmap is to enable *"a posteriori integration"* e.g. adding the relation `R1:{kegg:Drug voidext:subClassOf gr:Drug}` ensures that DSQE would infer all instances of `kegg:Drug` as instances of `gr:Drug` but not vice versa.

6.2 Semantic Inconsistencies and Future Work

In some cases, we found that catalogued classes would be better described as properties e.g. term `Symbol` is used both as a property and as a class. Since RDFS semantics does not allow a Class to be made a subclass of an entity of type "Property", these could not match using our methods. Moreover some URIs are used both as a class and as a property e.g. `http://bio2rdf.org/ncbi_resource:gene`. This can cause an inconsistency since a reasoning engine would either accept a URI as a property or a class. The algorithm used to create catalogue covers only classes containing instances and therefore our catalogue may not be capturing uninstantiated classes in any endpoints. Since our Roadmap was aimed only at linking classes for which roads can be discovered, we considered uninstantiated classes to be out of scope of our Roadmap. Class and property labels, used for discovering links, were generally obtained from the URI itself; however, there may be cases where labels for those entities may have been made available as part of ontologies or through external SPARQL endpoints. Those labels were not investigated in the Roadmap presented – however, Bioportal has exposed their annotated ontologies as a SPARQL endpoint and therefore in future we expect to make use of the labels provided by the original creators of the data.

7 Conclusion

Our preliminary analysis of existing SPARQL endpoint reveals that most Life Sciences and bio-related data cannot be easily mapped together. In fact, in the majority of cases there is very little ontology and URI reuse. Furthermore, many datasets include orphan URI - instances that have no "type"; and multiple URIs that cannot be dereferenced. Our Roadmap is a step towards cataloguing and linking the LSLOD through several different techniques. We evaluated the proposed Roadmap in terms of cataloguing time and entity capture and also showcased a few applications - namely query federation, selective SPARQL querying, drug discovery, semantic annotation and LSLOD visualization.

Acknowledgments

This research has been supported in part by Science Foundation Ireland under Grant Number SFI/12/RC/2289 and SFI/08/CE/I1380 (Lion 2). The authors would like to acknowledge Gofran Shukair for developing Google Refine Tool.

References

1. Alexander, K., Hausenblas, M.: Describing linked datasets-on the design and usage of void, the'vocabulary of interlinked datasets. In: In Linked Data on the Web Workshop (LDOW 09), in conjunction with WWW09. Citeseer (2009)
2. Bechhofer, S., Buchan, I., De Roure, D., Missier, P., et al.: Why linked data is not enough for scientists. Future Generation Computer Systems 29(2), 599–611 (2013)
3. Broekstra, J., Kampman, A., Van Harmelen, F.: Sesame: A generic architecture for storing and querying RDF and RDF schema. In: The Semantic Web—ISWC 2002, pp. 54–68. Springer (2002)
4. Buil-Aranda, C., et al.: SPARQL Web-Querying Infrastructure: Ready for Action? In: 12th International Semantic Web Conference, pp. 277–293. Springer (2013)
5. Cheung, K.H., Frost, H.R., Marshall, M.S., et al.: A journey to semantic web query federation in the life sciences. BMC bioinformatics 10(Suppl 10), S10 (2009)
6. Deus, H.F., Prud'hommeaux, E., Miller, M., Zhao, J., Malone, J., Adamusiak, T., et al.: Translating standards into practice–one semantic web API for gene expression. Journal of biomedical informatics 45(4), 782–794 (2012)
7. Deus, H.F., Zhao, J., Sahoo, S., Samwald, M.: Provenance of microarray experiments for a better understanding of experiment results (2010)
8. Goble, C., Stevens, R., Hull, D., et al.: Data curation+ process curation= data integration+ science. Briefings in bioinformatics 9(6), 506–517 (2008)
9. Hasnain, A., Fox, R., Decker, S., Deus, H.F.: Cataloguing and linking life sciences LOD Cloud. In: 1st International Workshop on Ontology Engineering in a Data-driven World collocated with EKAW12 (2012)
10. Hasnain, A., Kamdar, M.R., Hasapis, P., Zeginis, D., Warren Jr, C.N., et al.: Linked Biomedical Dataspace: Lessons Learned integrating Data for Drug Discovery. In: International Semantic Web Conference (In-Use Track), October 2014 (2014)
11. Jain, P., Hitzler, P., Sheth, A.P., Verma, K., Yeh, P.Z.: Ontology alignment for linked open data. In: The Semantic Web–ISWC 2010, pp. 402–417. Springer (2010)
12. Kamdar, M.R., Zeginis, D., Hasnain, A., Decker, S., Deus, H.F.: ReVeaLD: A user-driven domain-specific interactive search platform for biomedical research. Journal of Biomedical Informatics 47(0), 112 – 130 (2014)
13. Petrovic, M., Burcea, I., Jacobsen, H.A.: S-ToPSS: semantic toronto publish/subscribe system. In: Proceedings of the 29th international conference on Very large data bases-Volume 29. pp. 1101–1104. VLDB Endowment (2003)
14. Quackenbush, J.: Standardizing the standards. Molecular systems biology 2(1) (2006)
15. Schwarte, A., Haase, P., Hose, K., Schenkel, R., Schmidt, M.: Fedx: a federation layer for distributed query processing on linked open data. In: The Semanic Web: Research and Applications, pp. 481–486. Springer (2011)
16. Stein, L.D.: Integrating biological databases. Nature Reviews Genetics 4(5), 337–345 (2003)
17. Studer, R., Grimm, S., Abecker, A.: Semantic web services: concepts, technologies, and applications. Springer (2007)
18. Volz, J., Bizer, C., Gaedke, M., Kobilarov, G.: Discovering and maintaining links on the web of data. Springer (2009)
19. Zeginis, D., et al.: A collaborative methodology for developing a semantic model for interlinking Cancer Chemoprevention linked-data sources. Semantic Web (2013)