LinkLion: A Link Repository for the Web of Data

Markus Nentwig[1(✉)], Tommaso Soru[2], Axel-Cyrille Ngonga Ngomo[2], and Erhard Rahm[1]

[1] Database Group, Department of Computer Science, University of Leipzig, Leipzig, Germany
{nentwig,rahm}@informatik.uni-leipzig.de
[2] AKSW, Department of Computer Science, University of Leipzig, Leipzig, Germany
{tsoru,ngonga}@informatik.uni-leipzig.de

Abstract. Links between knowledge bases build the backbone of the Web of Data. Consequently, numerous applications have been developed to compute, evaluate and infer links. Still, the results of many of these applications remain inaccessible to the tools and frameworks that rely upon it. We address this problem by presenting LinkLion, a repository for links between knowledge bases. Our repository is designed as an open-access and open-source portal for the management and distribution of link discovery results. Users are empowered to upload links and specify how these were created. Moreover, users and applications can select and download sets of links via dumps or SPARQL queries. Currently, our portal contains 12.6 million links of 10 different types distributed across 3184 mappings that link 449 datasets. In this demo, we will present the repository as well as different means to access and extend the data it contains. The repository can be found at http://www.linklion.org.

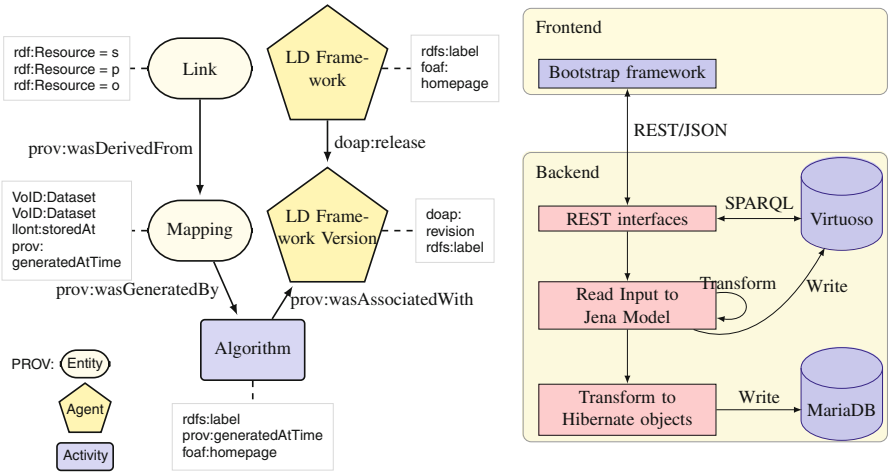
1 Introduction

In addition to being central for question answering across several datasets, links also play a key role in various other domains such as data fusion and federated SPARQL queries. It is a well-known problem that links make up less than 3 % of the RDF triples on the Web of Data [4]. This problem is being addressed by link discovery and ontology matching tools and frameworks [2,3]. However, due to the architectural choices behind the Web of Data, the results of a link discovery (LD) framework cannot be added directly to the datasets involved in the link discovery process. Further, the direct addition of links to a knowledge base fails to provide means to track the source of these links for later reference. Moreover, the availability of some endpoints still remains a major issue,[1] making the direct addition of linking results to some endpoints unattractive.

We address these drawbacks by presenting the open-source link repository LinkLion. The main goal of LinkLion is to facilitate the publication, retrieval

[1] http://labs.mondeca.com/sparqlEndpointsStatus.html

V. Presutti et al. (Eds.): ESWC Satellite Events 2014, LNCS 8798, pp. 439–443, 2014.
DOI: 10.1007/978-3-319-11955-7_63



(a) Overview of the LinkLion ontology. New classes such as Link, Mapping, Algorithm and LD Framework are specified as subclasses of the PROV vocabulary.

(b) Visualization of front and back end to store the mappings in the Virtuoso and MariaDB.

Fig. 1. Overview LinkLion ontology and architecture.

and use of links between knowledge bases. Our repository thus provides dedicated functionality for the upload, storage, querying and download of large sets of links. Currently, it contains 63 million triples which describe 12.6 million links of 10 different types (e.g., `owl:sameAs`, `dbo:spokenIn`, `foaf:made`, `spatial:P`)[2] distributed on 3184 mappings that link 449 datasets. These links were retrieved from the Web as well as computed by tools such as LIMES [3] and Silk [6]. Our repository provides a SPARQL query interface as well as commodity interfaces to access the mappings. In contrast to other portals such as BioPortal[3], LinkLion focuses exclusively on links and provides dedicated functionality for manipulating them. Moreover, we do not limit ourselves to a single domain such as the life sciences. In the following, we give a brief overview of the repository and show the use cases that will be presented during the demo. The repository can be accessed at http://www.linklion.org. The code of the repository is available at http://github.com/AKSW/LinkingLodPortal. The SPARQL endpoint can be found at http://www.linklion.org:8890/sparql.

2 Implementation

An overview of LinkLion's architecture is given in Fig. 1b. The back end consists of a triple store in which we save data according to the vocabulary shown in Fig. 1a.

[2] We used the prefixes available at http://prefix.cc.
[3] http://www.bioontology.org/BioPortal

The ontology[4] was designed with usability and reuse in mind. Especially, we wanted to allow end users of the portal to select dedicated portions of certain mappings at will. This meant designing an ontology that allowed amongst others (1) retrieving all links that pertain to a particular resource or set of resources, (2) gathering all mappings between datasets of interest as well as (3) getting aggregated information on how particular links came about. We implemented this vision by storing the output of a link discovery tool under an instance of the mapping class. Individual mappings can be described by metadata including the datasets that they link, the tool (incl. a version number) used to generate the links and the creation date of the mapping. We refrained from using blank nodes for links. Instead, we gave each link a unique ID. Note that we reused existing vocabularies (especially PROV[5], VoID[6] and DOAP[7]) as much as we could. The use of a triple store pays off as end users can choose to provide more metadata such as the link specification used or parameters of the algorithm they used to discover the link without us having to alter our schema. For the sake of scalability, we yet also provide the core of the data in the triple store as SQL dump. The functionality of the back end is exposed by RESTful interfaces, which allow a programmatic access to LinkLion from code written in virtually any modern programming language.

The front end of our repository provides an easy way to use some of the functionality of LinkLion (see Fig. 2a). First, it allows users to upload new mappings. Users are asked to provide a source file in the N-Triples format[8]. well as the algorithm used within this framework have to be provided (note that we consider humans to also be linking frameworks). The data (and especially the mapping) given by the user is then checked for consistency and uploaded into the underlying triple store. The content of the triple store can be browsed directly from the web page (see Fig. 2b). Especially, the front end includes search functionality and pagination which allow end users to search for mappings that link to or from a dataset of interest. The upload and browsing functionality will be presented during the demo.

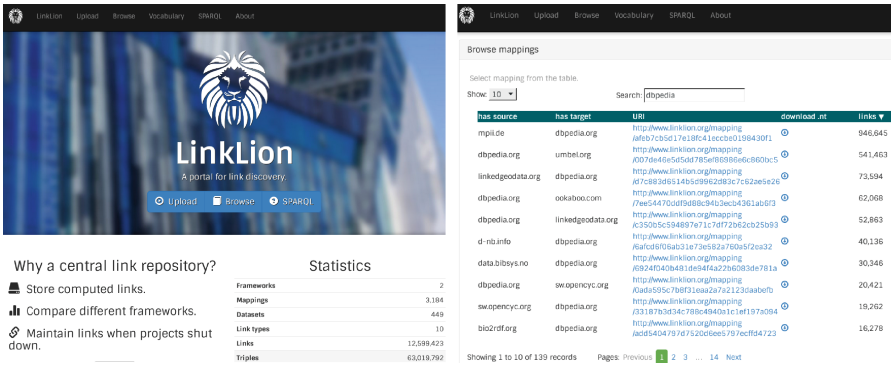
3 Use Cases

In this section, we present and motivate a selection of use cases that will be presented during the demo session.

3.1 Gather All Links and Mappings to a Given Resource

Gathering and fusing all information on a resource of interest is of central importance to applications such as Question Answering systems, Linked Data Browsers

[4] Available at http://www.linklion.org/ontology
[5] http://www.w3.org/TR/prov-o/
[6] http://www.w3.org/TR/void/
[7] https://github.com/edumbill/doap/
[8] http://www.w3.org/2001/sw/RDFCore/ntriples/



(a) LinkLion Homepage with statistics regarding the content.

(b) Mapping Browser. The search for DBpedia returns 139 mappings.

Fig. 2. Front-end views.

and Quality Assessment tools. LinkLion allows to gather all links pertaining to a particular resource (`dbpedia:Thailand` in our example) by means of the following SPARQL query. By using this information, novel repair-based algorithms for link discovery such as Colibri can find errors or inconsistencies in the data [5].

```
SELECT ?link WHERE { { ?link rdf:subject dbpedia:Thailand }
             UNION { ?link rdf:object dbpedia:Thailand } }
```

The portal also allows gather all mappings that contain links pertaining to a particular resource, e.g.. `dbpedia:Thailand`, as shown below.

```
SELECT DISTINCT ?mapping WHERE { ?link prov:wasDerivedFrom ?mapping .
                                 { ?link rdf:subject dbpedia:Thailand }
                           UNION { ?link rdf:object dbpedia:Thailand } }
```

3.2 Get Support for a Link

Ensemble learning techniques have been shown to improve the results of manifold machine-learning applications such as Named Entity Recognition frameworks. Our repository facilitates the use of ensemble learning for combining the results of different link discovery tools. Especially, LinkLion allows us to retrieve (if any) the list of mappings that contain a given link, as well as the algorithms and the frameworks that generated it. In the following example, the support for `dbpedia:Thailand owl:sameAs` < http://sws.geonames.org/1605651/ > is queried.

```
SELECT ?mapping ?algorithm ?framework WHERE {
    ?mapping prov:wasGeneratedBy ?algorithm .
    ?algorithm prov:wasAssociatedWith ?framework .
    ?link prov:wasDerivedFrom ?mapping ;
          rdf:predicate owl:sameAs .
    { ?link rdf:subject dbpedia:Thailand;
            rdf:object <http://sws.geonames.org/1605651/> }
    UNION { ?link rdf:object dbpedia:Thailand;
                  rdf:subject <http://sws.geonames.org/1605651/> } }
```

3.3 Link Composition

With the growth of the Linked Data Web, it becomes ever more important to regard link discovery as a holistic process that goes beyond linking a pair of knowledge bases. Algorithms based on composition can exploit sequences of links to enrich their mapping composition graphs [1]. Moreover, algorithms which link several knowledge bases at the same time [5] can achieve higher accuracies. By using LinkLion, composition and concurrent linking algorithms are now enabled to gather the data they require without having to manage all the links by themselves. In the query below, all resources related to `dbpedia:Thailand` over two links are retrieved from the repository.

```
SELECT DISTINCT ?resource WHERE { {
    ?link rdf:subject dbpedia:Thailand ; rdf:object ?x .
    ?link2 rdf:subject ?x ; rdf:object ?resource }
    UNION { ?link rdf:object dbpedia:Thailand ; rdf:subject ?x .
            ?link2 rdf:object ?x ; rdf:subject ?resource } }
```

4 Summary

This paper presents LinkLion, a repository for links between knowledge bases of the Web of Data. The repository enables users to upload results of a link discovery process and furthermore allows them to add information on how the results were created. LinkLion therefore provides management and distribution capabilities through a both open-access and open-source web interface. Resulting sets of links can be reviewed in the portal and via SPARQL queries, additionally the results can be downloaded via dumps. In future work, LinkLion will be extended to support a closely collaboration with external link discovery frameworks and applications.

References

1. Hartung, M., Groß, A., Rahm, E.: Composition methods for link discovery. In: BTW (2013)
2. Kirsten, T., Gross, A., Hartung, M., Rahm, E.: GOMMA: a component-based infrastructure for managing and analyzing life science ontologies and their evolution. J. Biomed. Semant. **2**, 6 (2011)
3. Ngonga Ngomo, A.-C.: On link discovery using a hybrid approach. J. Data Semant. **1**(4), 203–217 (2012)
4. Ngonga Ngomo, A.-C., Auer, S.: LIMES: a time-efficient approach for large-scale link discovery on the web of data. In: Proceedings of the 22nd International Joint Conference on Artificial Intelligence, IJCAI'11, vol. 3, pp. 2312–2317. AAAI Press (2011)
5. Ngonga Ngomo, A.-C., Sherif, M.A., Lyko, K.: Unsupervised link discovery through knowledge base repair. In: Presutti, V., d'Amato, C., Gandon, F., d'Aquin, M., Staab, S., Tordai, A. (eds.) ESWC 2014. LNCS, vol. 8465, pp. 380–394. Springer, Heidelberg (2014)
6. Volz, J., Bizer, C., Gaedke, M., Kobilarov, G.: Silk - a link discovery framework for the web of data. In: Bizer, C., Heath, T., Berners-Lee, T., Idehen, K. (eds.) LDOW. CEUR Workshop Proceedings, vol. 538. CEUR-WS.org (2009)