Targeted Linked-data Extractor

Pierre Maillot[1], Thomas Raimbault[1], David Genest[2] and Stéphane Loiseau[2]
[1]*ESILV, Pôle Universitaire Léonard de Vinci, 12 avenue Léonard de Vinci, Courbevoie, France*
[2]*LERIA, Faculté des Sciences d'Angers, 4 Boulevard Lavoisier, Angers, France*

Keywords: Semantic Web, Linked Data, Extraction.

Abstract: The Linked Data Cloud is too big to be locally manipulated by standard computers and all use-cases doesn't need to manipulate the whole cloud. To get exactly what is needed for a specific use-case, we need to obtain the specific parts from each bases of the Linked Data Cloud. This paper proposes a method to smartly extract a sub-part of the Linked Data Cloud driving by a list of resources called *seeds*. This method consist of extracting data starting from *seed* resources and recursively expanding the extraction to their neighbours.

1 INTRODUCTION

Linked Data[1] is not anymore the domain of Semantic Web enthusiasts and academics. Today the Linked Data Cloud counts more than 300 bases in different domains, of sizes varying between a few hundred to several billions statements (known as *triples*). As it became more widely recognized, new uses have appeared and to each use corresponds a specific part of a base, or of the whole Linked Data Cloud.

When there is need to locally manipulate the Linked Data Cloud, some use-cases only need specifics sub-parts of bases of the Linked Data Cloud. Currently only base dumps allow us to manipulate locally data from the Linked Data Cloud. But dumps can not be easily exploited by human (because they contain raw data) or by a *standard* machine (because of their size). In this paper we propose a method to smartly extract real parts of the Linked Data according to given specifications.

Our motivation to develop such a method came from a previous work where we needed a "mini Linked Data Cloud" to evaluate a novel approach for querying a set of RDF bases. Works in similar situations (Bail et al., 2012; Schmidt et al., 2011; Morsey et al., 2011; Schmidt et al., 2008) used either raw dumps, or random-generated datasets, or present their datasets in theory only, without access to the tools they used. To meet our needs, we have extracted some sub-parts of (real) data from the universal DBPedia base to create different bases in accordance with specific domains of the Linked Data Cloud.

Our contribution in this paper is to propose both a method and an implementation, driven by *seed resources*, to extract a focused sub-part of the Linked Data Cloud. These resources have to be selected such as the extracted sub-part contains the expected statements.

This article is organized as follow, in Section 2 we recall some principles about the Semantic Web and the Linked Data. Section 3 details the method to extract sub-part of RDF bases. Section 4 presents an example of utilization of our tool to generate data for a benchmark. Section 5 shows details of our implementation of the method of extraction.

2 THE SEMANTIC WEB TODAY

The semantic web, or *Web of data*, is a collaborative movement led by the W3C and proposes some common methods to store, share, find and combine information. Some specially designed languages are proposed: RDF for data description, RDFS and OWL for definition of ontologies. We do not provide here complete definitions of these languages. We just recall that a RDF (Klyne and Carroll, 2004) document contains triples as (subject,predicate,object), and such a document can be seen as a labeled directed multigraph, where each triple is an arc between the node that represents its subject and the node that represents its object. Figure 1 shows an example of an

[1] http://linkeddata.org/

Maillot P., Raimbault T., Genest D. and Loiseau S..
Targeted Linked-Data Extractor.
DOI: 10.5220/0004758503360341
In *Proceedings of the 6th International Conference on Agents and Artificial Intelligence* (ICAART-2014), pages 336-341
ISBN: 978-989-758-015-4

RDF graph representation. In this figure, white nodes represent factual resources and gray nodes represent types of factual resources.

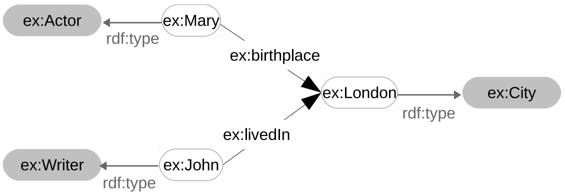

Figure 1: RDF document.

RDFS (Brickley and Guha, 2004) provides basic elements for the description of ontologies intended to structure RDF resources. RDFS allows to (i) declare types of resources (rdfs:Class) and their properties (rdf:Property) ; (ii) specify the signature of properties (rdfs:domain and rdfs:range) ; (iii) specify the inheritance links among classes (rdfs:subClassOf) or among properies (rdfs:subPropertyOf). Figure 2 describes some classes and properties, and their inheritance links.

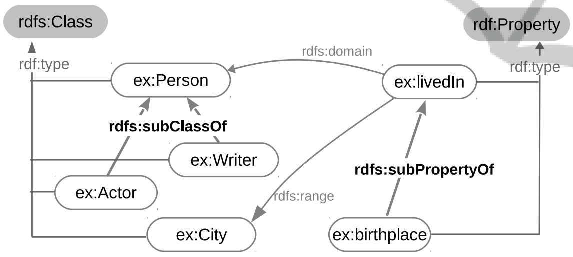

Figure 2: RDFS classes and properties.

A *triplestore* is a specially designed database for storage and retrieval of RDF triples. In order to query such a base, SPARQL (W3C, 2013) is the most widely user language.

Linked Data

The Linked Data is a movement supported by a large community, encouraging the publishing of open RDF bases respecting three best practice rules:

- All resources must be represented by a dereferencable URI.
- Bases must be interlinked
- Each base must be publicly accessible via a SPARQL endpoint.

All Linked Data interlinked bases compose the Linked Data Cloud, counting several hundred bases in various domains and sizes. But today there is no centralized authority maintaining a list of all available bases. Currently the most complete list is available on datahub[2].

Table 1: Linked Data Cloud statistics by domains.

Domains	Number of datasets	Triples	%
Media	25	1,841 M	5.82 %
Geographic	31	6,145 M	19.43 %
Government	49	13,315 M	42.09 %
Publications	87	2,950 M	9.33 %
Cross-domain	41	4,184 M	13.23 %
Life sciences	41	3,036 M	9.60 %
User-generated content	20	134 M	0.42 %
Total	295	31,634 M	

Table 1 contains statistics[3] about the repartition of Linked Data bases in major domains and their sizes as of September 2011.

Table 2 contains information about 6 bases from the Linked Data Cloud. These information where produced by us in September 2013, using either SPARQL1.1 endpoints where they were available (to use queries containing the *COUNT()* function) or provided on the databases homepages. This table give a quick overall sight of the diversity of the Linked Data Cloud.

3 HOW TO EXTRACT FOCUSED SUB-PARTS FROM LINKED DATA SOURCES

Our goal is to extract from one or more RDF bases a sub-part containing relevant statements according to a specific topic. To drive the extraction to obtain the expected focused sub-part we use a set of resources from which statements will be gradually extracted. The extraction itself use a particular treatment for blank nodes to avoid meaningless statement to be retrieved.

3.1 Driving the Extraction

As presented in (Kleinberg, 1999), on the Web it is possible to extract a majority of authorities regarding one topic by retrieving all connected pages to one

[2] Non exhaustive list at: http://datahub.io/group/lodcloud

[3] from http://lod-cloud.net/state/

Table 2: Statistics for some bases of the Linked Data Cloud.

Base	# of triples	# of individuals	# of classes	# of interlinks	#1 instantiated class	#2 instantiated class	#3 instantiated class
DBPedia (en)	400M	3 770K	359	10 893K	foaf:Person (33,61%)	dbOnto:Agent (25,37%)	skos:Concept (22,89%)
LOGD	9 951M	25 987M	248	5 787	purlConv:CatalogedDataset (18,56%)	geo:Point (7,68%)	logd:Point (7,68%)
Enipedia	4 180K	2 102K	99	1 758	mediaWiki:Subject (23,46%)	europa:PollutantRelease (9,55%)	europa:WasteTransfer (9,13%)
Jamendo	1 062K	290K	16	408	purlOnto:Playlist (35,41%)	purlOnto:Signal (15,72%)	purlOnto:Track (15,72%)
LMDB	6 148K	503K	53	162K	lmdb:film (17,01%)	lmdb:actor (10,05%)	lmdb:director (3,40%)
Revyu	20K	10K	5	75	redwood:Tag (37,90%)	redwood:Tagging (21,00%)	owl:Thing (19,96%)
SW Dog Food	539K	56K	92	603	foaf:Person (16,39%)	swrcOnto:InProceedings (7,11%)	foaf:Organization (4,90%)

known authority to a rank *n*. This approach has been made ineffective for internet search algorithms because of the malicious use of links farm and others exploitations of algorithm properties, some of them are listed in (Gyongyi and Garcia-Molina, 2005). However in the Semantic Web, we can adapt this idea by extracting triples around selected resources (*e.g.* important resource about a given domain) to obtain most of the relevant information in this domain. In this case, instead of web pages, we obtain links around a selected resource which all composed knowledge of the domain.

We call seed a set of resources chosen to start the extraction. The seed resources can be either classes or individuals.

To extract the focused sub-part targeted by a seed as target, we firstly use a *DESCRIBE* query about each seed resource. In SPARQL, a DESCRIBE query applied on a resource returns all triples containing this resource both as subject and as object. The result composes the first rank of the extraction. Secondly we list all new resources appearing in the previous resulting triples and we repeat this operation on each of them to retrieve the next rank extraction. The following ranks are obtained by the same way.

Figure 3 is a graphical representation of an extraction from one seed resource to rank 2. Our extractor follow the algorithm 3.1, given a seed L_{SE} and a maximum rank *n*. In other words this algorithm recursively and non-naively extracts parts of RDF bases by expanding the information on a selected resource to its neighbours.

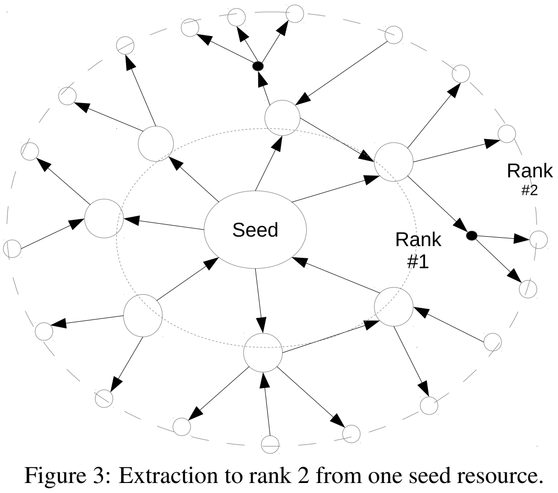

Figure 3: Extraction to rank 2 from one seed resource.

3.2 Parameters

To obtain a focused sub-part of the Linked Data Cloud on a chosen topic there are two parameters to configure the algorithm.

For a chosen topic, the seed has to contain target resources linked to most of the relevant resources to this topic. For example, the seed of an extraction can

Algorithm 3.1: Extraction of statements centred on a resource s.

```
Input: $E : Endpoint$                         ▷ SPARQL Endpoint
Input: $s : Resource$                         ▷ Seed resource
Input: $n : Integer$                          ▷ Extraction rank
Output: $g : Triple\ set$                     ▷ Extracted statements
  function EXTRACTSEEDRANK($E, s, n$)
    $l = []$                                  ▷ Next rank resource list
    if $n > 0$ then
      $g = Q_D(E, s)$
      for all $t \in g$ do
        if $subject(t) == s$ AND $isURI(object(t))$ then
          $l += object(t)$
        else
          if $object(t) == s$ then
            $l += subject(t)$
          end if
                                              ▷ Hidden n-ary relations
          if $subject(t) == s$ AND $isBlankNode(object(t))$ then
            $g += Q_C(E, t)$
          end if
        end if
      end for
      for all $s' \in l$ do
        $g +=$ ExtractSeedRank($s', n - 1$)
      end for
    else
      return $g$
    end if
  end function
```

Algorithm 3.2: Request retrieving all triples with statement x's object as subject.

```
Input: $E$ : Endpoint                         ▷ SPARQL Endpoint
Input: $x : Statement$                        ▷ (subject, predicate, object)
Output: $g : Triples\ set$
  function Q$_C$($E$,$x$)
    return result of query on E
    CONSTRUCT {
    object($x$) ?var1 ?var2.
    }
    WHERE {
    object($x$) ?var1 ?var2.
    }
  end function
```

be a list of individuals from a given class or the class itself (*e.g.* to extract some cinematographic information from DBPedia we can use the "Movie" class, or more precisely a list of all films produced by Quentin Tarantino).

The choice of the maximum rank of the extraction limit the size on the extracted sub-part and also determines how much concentrated the sub-part of a base will be around a seed resource. Note that in theory it is possible to retrieve all statements of a base by choosing the highest class in the ontology (*e.g.* rdfs:Resource or owl:Thing) and a very high maximum rank. Even if this extraction mechanism comes easily to mind, the quality of the extraction result depends heavily on the meticulous choice of the seed (as it is underlined in (Morsey et al., 2011)).

Algorithm 3.3: Request retrieving all statements containing resource r as subject or object.

```
Input: $E$ : Endpoint                         ▷ SPARQL Endpoint
Input: $r : Resource$
Output: $g : Triple\ set$
  function Q$_D$($E$,$r$)
    return result of query on E
    DESCRIBE $r$
  end function
```

3.3 Hidden n-ary Relations

During an extraction, it could be possible to retrieve some triples containing a blank node at the last extracted rank. This kind of triple can be paraphrased as "this resource is linked to an unidentified resource". In RDF, blank nodes are used to represent anonymous resources to enable the creation of n-ary relations in a binary relation representation model. To avoid these meaningless statements in the extraction process, we regroup statements by considering blank nodes as hidden n-ary relations, as proposed in (Tummarello et al., 2005). As a consequence, we consider all groups of triples containing the same blank node in the same rank.

Figure 4 shows that a naive extraction could contain triples with a blank node as object, without the triples giving them their meaning as an n-ary relation.

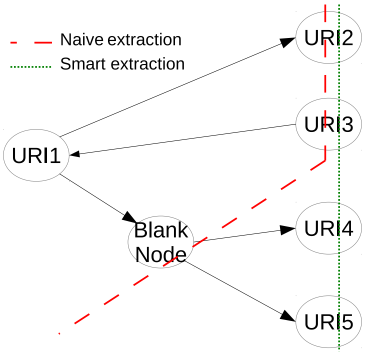

Figure 4: Schema representing the difference between a naive extraction method and the extraction method, starting from URI$_1$ to rank 1.

Table 3: Distribution of class instantiation in DBPedia and in our extracted bases.

Base	Person	Place	Organisation	Work	Other
DBPedia	23,12%	17,06%	5,48%	9,57%	44,76%
Academic Journals	2,76%	1,78%	7,59%	41,29%	46,58%
Berlin	48,66%	8,09%	12,10%	5,63%	25,52%
Cat	34,21%	2,63%	2,63%	2,63%	57,89%
Elvis	9,54%	0,87%	1,95%	75,27%	12,37%
Legal Case	0,95%	0,00%	0,05%	0,08%	98,92%
Moon	0,40%	97,11%	0,13%	0,07%	2,28%
Paris	53,37%	2,18%	11,46%	5,08%	27,91%
Potato	3,23%	0,00%	9,68%	0,00%	87,10%
Tarantino's Movies	41,74%	0,41%	4,55%	7,02%	46,28%
Zola's Books	1,47%	1,47%	0,00%	19,12%	77,94%

For instance, if a statement containing a blank node b_1 as object appears at a rank n, then all statements containing b_1 as subject must appear at rank n, instead of appearing at rank $n+1$ in a naive approach. These groups of statements are called *Concise Bounded Description*, proposed in (Stickler, 2005), and are considered as an optimal form of description of a resource. Figure 4 represents an extraction from the seed composed by the singleton $\{\mathrm{URI}_1\}$ to rank 1, considering the blank node between the seed and both URI$_4$ and URI$_5$ as a 3-ary relation. So, the rank 1 extraction contains the URI$_2$ to URI$_5$.

4 USE-CASE: DATA GENERATION FOR BENCHMARKING

For some uses, especially for benchmarking, it is interesting to locally handle data from the Linked Data Cloud and specially only needed sub-part from this big cloud to a given use-case. There is a limited offer of benchmarks to evaluate query methods or engines on the Linked Data. Most of these benchmarks use dumps from the most known bases of the Linked Data Cloud (Bail et al., 2012; Schmidt et al., 2011) or use random generator (Schmidt et al., 2008). So, the choice is either using huge bases with real data or manageable bases with fake data. Using real data at an acceptable scale for a "standard" computer[4], as base dumps often contain at least several million triples.

In a previous work, to test a new approach for querying a set of distant RDF bases, as presented in (Raimbault and Maillot., 2013), we needed a mini-Linked Data Cloud where each base is composed with real data on a specific domain, is small enough to be processed in our experimental environment, and has a SPARQL endpoint.

We present here the experience of an extraction use-case we used in this previous work, according to the method presented in Section 3. To evaluate our approach we needed to evaluate different method of querying the Linked Data Cloud. For practical reason we chose to only use DBPedia, the biggest multi-domain base in the Linked Data Cloud, to have the same ontology for every base. Even with the same ontology we aimed to keep the "structural" differences between each base as those between each base of the Linked Data Cloud. These differences resided in the instantiation distribution in each class for each base, *i.e.* in each specialized domain, there is more individuals of the classes representatives of the domain subject (*e.g.* Animal in Life Science, Document in Publication, *etc.*) than others.

We extracted with 2 as maximum rank, 10 different bases for our tests[5]. The seeds were chosen to represent some specific domains of the Linked Data Cloud:

- The Cat class (in the "Life Sciences" domain)
- The Scientific Journal class (in the "Publications" domain)
- The Legal Case class (in the "Government" domain)
- The Potato individual (in the "Life Sciences" domain)
- The Moon individual
- The singer Elvis Presley (in the "Media" domain)
- The city of Paris (in the "Geographic" domain)
- The city of Berlin (in the "Geographic" domain)

[4] Less than 10 cores, less than 10Gb RAM.

[5] by using our tool 10 times

- The list of all films realized by Quentin Tarantino (in the “Media” domain)
- The list of all books written by Emile Zola (in the “Media” domain)

Table 3 shows the distribution of class instantiations for each extracted base compared to their source DBPedia[6]. The differences in instantiation repartition show that the seeds-driven extraction returned bases different from their source and different from each other.

With our method, we created a set of bases sharing common resources but treating different topics by extracting specifics parts of a Linked Data base, allowing us to evaluate our approach for querying a set of base of the Linked Data Cloud at a manageable scale.

5 IMPLEMENTATION

Our tool is based on *bash* scripts and simple java tools[7] based on the Jena[8] framework.

Our tool is available at http://der3i.labs.esilv.fr/download/RdfExtractor.zip

6 CONCLUSIONS

We have presented a method for extracting targeted sub-part of RDF bases, driving by a list of selected resources called *seed*. This method non-naively extracts parts of a RDF base by recursively expanding the information on each selected resource and its neighbours. This method can be used to create a dataset with real data for benchmarking query approach on several RDF base, as it is needed through the Linked Data Cloud.

In future works we plan to use this method in a process to create benchmarks for the Linked Data by using statistics gathered during the extraction to automatically suggest associated queries. Furthermore this method could be used as a tool in user interface for the generation of RDF base summaries.

AKNOWLEDGEMENTS

This work has been funded by École Supérieure d’Ingénieur Léonard de Vinci (ESILV) in the frame of the French cooperative project TIMCO, Pôle de Compétitivité Systematic (FUI 13).

REFERENCES

Bail, S., Alkiviadous, S., Parsia, B., Workman, D., Van Harmelen, M., Goncalves, R. S., and Garilao, C. (2012). Fishmark: A linked data application benchmark. In *Joint Workshop on Scalable and High-Performance Semantic Web Systems (SSWS+ HPCSW 2012)*, page 1.

Brickley, D. and Guha, R. V. (2004). RDF Vocabulary Description Language 1.0: RDF Schema. http://www.w3.org/TR/rdf-schema/.

Gyongyi, Z. and Garcia-Molina, H. (2005). Web spam taxonomy. In *First international workshop on adversarial information retrieval on the web (AIRWeb 2005)*.

Kleinberg, J. M. (1999). Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5):604–632.

Klyne, G. and Carroll, J. J. (2004). Resource Description Framework (RDF): Concepts and Abstract Syntax. http://www.w3.org/TR/rdf-concepts/.

Morsey, M., Lehmann, J., Auer, S., and Ngomo, A.-C. N. (2011). Dbpedia sparql benchmark–performance assessment with real queries on real data. In *The Semantic Web–ISWC 2011*, pages 454–469. Springer.

Raimbault, T. and Maillot., P. (2013). Vues d’ensembles de documents RDF. In *Actes du 31ème congrès INFORSID*, pages 387–402, Paris, France.

Schmidt, M., Görlitz, O., Haase, P., Ladwig, G., Schwarte, A., and Tran, T. (2011). Fedbench: a benchmark suite for federated semantic data query processing. In *The Semantic Web–ISWC 2011*, pages 585–600. Springer.

Schmidt, M., Hornung, T., Lausen, G., and Pinkel, C. (2008). Sp2bench: A sparql performance benchmark. *CoRR*, abs/0806.4627.

Stickler, P. (2005). Concise bounded description. *W3C Member Submission*. http://www.w3.org/Submission/CBD/.

Tummarello, G., Morbidoni, C., Puliti, P., and Piazza, F. (2005). Signing individual fragments of an rdf graph. In *Special interest tracks and posters of the 14th international conference on World Wide Web*, pages 1020–1021. ACM.

W3C (2013). SPARQL 1.1 Query Language. http://www.w3.org/TR/sparql11-overview/.

[6] Statistics about DBPedia are from http://wiki.dbpedia.org

[7] For the querying of a remote base.

[8] http://jena.apache.org/