Web Semantics: Science, Services and Agents on the World Wide Web 10 (2012) 76–110

Contents lists available at SciVerse ScienceDirect

Web Semantics: Science, Services and Agents on the World Wide Web

journal homepage: http://www.elsevier.com/locate/websem

Scalable and distributed methods for entity matching, consolidation and disambiguation over linked data corpora

Aidan Hogan [a,*], Antoine Zimmermann [b], Jürgen Umbrich [a], Axel Polleres [c], Stefan Decker [a]

[a] Digital Enterprise Research Institute, National University of Ireland, Galway, Ireland
[b] INSA-Lyon, LIRIS, UMR5205, Villeurbanne F-69621, France
[c] Siemens AG Österreich, Siemensstrasse 90, 1210 Vienna, Austria

ARTICLE INFO

Article history:
Available online 18 November 2011

Keywords:
Entity consolidation
Web data
Linked Data
RDF

ABSTRACT

With respect to large-scale, static, Linked Data corpora, in this paper we discuss scalable and distributed methods for entity consolidation (aka. smushing, entity resolution, object consolidation, etc.) to locate and process names that signify the same entity. We investigate (i) a baseline approach, which uses explicit `owl:sameAs` relations to perform consolidation; (ii) extended entity consolidation which additionally uses a subset of OWL 2 RL/RDF rules to derive novel `owl:sameAs` relations through the semantics of inverse-functional properties, functional-properties and (max-)cardinality restrictions with value one; (iii) deriving weighted concurrence measures between entities in the corpus based on shared inlinks/outlinks and attribute values using statistical analyses; (iv) disambiguating (initially) consolidated entities based on inconsistency detection using OWL 2 RL/RDF rules. Our methods are based upon distributed sorts and scans of the corpus, where we deliberately avoid the requirement for indexing all data. Throughout, we offer evaluation over a diverse Linked Data corpus consisting of 1.118 billion quadruples derived from a domain-agnostic, open crawl of 3.985 million RDF/XML Web documents, demonstrating the feasibility of our methods at that scale, and giving insights into the quality of the results for real-world data.

1. Introduction

Over a decade since the dawn of the Semantic Web, RDF publishing has begun to find some traction through adoption of Linked Data best practices as follows:

(i) use URIs as names for things (and not just documents);
(ii) make those URIs dereferenceable via HTTP;
(iii) return useful and relevant RDF content upon lookup of those URIs;
(iv) include links to other datasets.

The Linked Open Data project has advocated the goal of providing dereferenceable machine readable data in a common format (RDF), with emphasis on the re-use of URIs and interlinkage between remote datasets—in so doing, the project has overseen exports from corporate entities (e.g., the BBC, BestBuy, Freebase), governmental bodies (e.g., the UK Government, the US government), existing structured datasets (e.g., DBPedia), social networking sites (e.g., flickr, Twitter, livejournal), academic communities (e.g., DBLP, UniProt), as well as esoteric exports (e.g., Linked Open Numbers, Poképédia). This burgeoning web of structured data has succinctly been dubbed the "Web of Data".

Considering the merge of these structured exports, at a conservative estimate there now exists somewhere in the order of thirty billion RDF triples published on the Web as Linked Data.[1] However, in this respect, size is not everything [73]. In particular, although the situation is improving, individual datasets are still not well-interlinked (cf. [72])—without sufficient linkage, the ideal of a "Web of Data" quickly disintegrates into the current reality of "Archipelagos of Datasets".

There have been numerous works that have looked at bridging the archipelagos. Some works aim at aligning a small number of

* Corresponding author.
E-mail addresses: aidan.hogan@deri.org (A. Hogan), antoine.zimmermann@insa-lyon.fr (A. Zimmermann), juergen.umbrich@deri.org (J. Umbrich), axel.polleres@siemens.com (A. Polleres), stefan.decker@deri.org (S. Decker).

[1] http://www4.wiwiss.fu-berlin.de/lodcloud/state/.

1570-8268/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.websem.2011.11.002

related datasets (e.g., [48,58,49]), thus focusing more on theoretical considerations than scalability, usually combining symbolic (e.g., reasoning with consistency checking) methods and similarity measures. Some authors have looked at inter-linkage of domain specific RDF datasets at various degrees of scale (e.g., [61,59, 38,54,46]). Further research has also looked at exploiting shared terminological data—as well as explicitly asserted links—to better integrate Linked Data collected from thousands or millions of sources (e.g., [30,50,15,35]); the work presented herein falls most closely into this category. One approach has tackled the problem from the publishers side, detailing a system for manually specifying some (possibly heuristic) criteria for creating links between two datasets [72]. We leave further detailed related work to Section 9.

In this paper, we look at methods to provide better linkage between resources, in particular focusing on finding *equivalent* entities in the data. Our notion of an *entity* is a representation of something being described by the data; e.g., a person, a place, a musician, a protein, etc. We say that two entities are equivalent if they are *coreferent*; e.g., refer to the same person, place, etc.[2] Given a collection of datasets that speak about the same referents using different identifiers, we wish to identify these coreferences and somehow merge the knowledge contribution provided by the distinct parties. We call this merge *consolidation*.

In particular, our work is inspired by the requirements of the Semantic Web Search Engine project [32], within which we aim to offer search and browsing over large, static, Linked Data corpora crawled from the Web.[3] The core operation of SWSE is to take user keyword queries as input, and to generate a ranked list of matching entities as results. After the core components of a crawler, index and user-interface, we saw a clear need for a component that consolidates—by means of identifying and canonicalising equivalent identifiers—the indexed corpus: there was an observable lack of URIs such that coreferent blank-nodes were prevalent [30] even within the same dataset, and thus we observed many duplicate results referring to the same thing, leading to poor integration of data from our source documents.

To take a brief example, consider a simple example query: "WHO DOES TIM BERNERS-LEE KNOW?". Knowing that Tim uses the URI `timblfoaf:i` to refer to himself in his personal FOAF profile document, and again knowing that the property `foaf:knows` relates people to their (reciprocated) acquaintances, we can formulate this request as the SPARQL query [53] as follows:

```
SELECT ?person
WHERE {
  timblfoaf:i foaf:knows ?person.
}
```

However, other publishers use different URIs to identify Tim, where to get more complete answers across these naming schemes, the SPARQL query must use disjunctive `UNION` clauses for each known URI; here we give an example using a *sample* of identifiers extracted from a real Linked Data corpus (introduced later):

```
SELECT ?person
WHERE {
  {timblfoaf:i foaf:knows ?person.}
  UNION {identicau:45563 foaf:knows ?person.}
  UNION {dbpedia:Berners-Lee foaf:knows ?person.}
  UNION {dbpedia:Dr._Tim_Berners-Lee foaf:knows
  ?person.}
  UNION {dbpedia:Tim-Berners_Lee foaf:knows
  ?person.}
  UNION {dbpedia:TimBL foaf:knows ?person.}
  UNION {dbpedia:Tim_Berners-Lee foaf:knows
  ?person.}
  UNION {dbpedia:Tim_berners-lee foaf:knows
  ?person.}
  UNION {dbpedia:Timbl foaf:knows ?person.}
  UNION {dbpedia:Timothy_Berners-Lee foaf:knows
  ?person.}
  UNION {yagor:Tim_Berners-Lee foaf:knows ?person.}
  UNION {fb:en.tim_berners-lee foaf:knows ?person.}
  UNION {swid:Tim_Berners-Lee foaf:knows ?person.}
  UNION {dblpperson:100007 foaf:knows ?person.}
  UNION {avtimbl:me foaf:knows ?person.}
  UNION {bmpersons:Tim+Berners-Lee foaf:knows
  ?person.}
  ...
}
```

We see disparate URIs not only across data publishers, but also within the same namespace. Clearly, the expanded query quickly becomes extremely cumbersome.

In this paper, we look at bespoke methods for identifying and processing coreference in a manner such that the resultant corpus can be consumed as if more complete agreement on URIs was present; in other words, using standard query-answering techniques, we want the enhanced corpus to return the same answers for the original simple query as for the latter expanded query.

Our core requirements for the consolidation component are as follows:

- the component *must* give **high precision** of consolidated results;
- the underlying algorithm(s) *must* be **scalable**;
- the approach *must* be **fully automatic**;
- the methods *must* be **domain agnostic**;

where a component with poor precision will lead to garbled final results merging unrelated entities, where scalability is required to apply the process over our corpora typically in the order of a billion statements (and which we feasibly hope to expand in future), where the scale of the corpora under analysis precludes any manual intervention, and where—for the purposes of research—the methods should not give preferential treatment to any domain or vocabulary of data (other than core RDF(S)/OWL terms). Alongside these *primary requirements*, we also identify the following *secondary criteria*:

- the analysis *should* demonstrate **high recall**;
- the underlying algorithm(s) *should* be **efficient**;

where the consolidation component should identify as many (correct) equivalences as possible, and where the algorithm should be applicable in reasonable time. Clearly the secondary requirements are also important, but they are superceded by those given earlier, where a certain trade-off exists: we prefer a system that

[2] Herein, we avoid philosophical discussion on the notion of identity; for interesting discussion thereon, see [26].

[3] By static, we mean that the system does not cater for updates; this omission allows for optimisations throughout the system. Instead, we aim at a cyclical indexing paradigm, where new indexes are bulk-loaded in the background on separate machines.