Aether – Generating and Viewing Extended VoID Statistical Descriptions of RDF Datasets

Eetu Mäkelä(✉)

Semantic Computing Research Group (SeCo), Aalto University, Espoo, Finland
`eetu.makela@aalto.fi`
`http://www.seco.tkk.fi/`

Abstract. This paper presents the Aether web application for generating, viewing and comparing extended VoID statistical descriptions of RDF datasets. The tool is useful for example in getting to know a newly encountered dataset, in comparing datasets between versions and in detecting outliers and errors. Examples are given on how the tool has been used to shed light on multiple important datasets.

1 Introduction

Dataset descriptions, of which VoID descriptions [1] are the current norm, are RDF descriptions of the contents of RDF datasets. They contain for example information about the licensing and access endpoints of the dataset, as well as statistical spreads about its content and interlinking. Use cases for such descriptions [2,3] include improving discovery and selection of datasets for a particular task [4], as well as query optimization, particularly in federared querying [5,6].

However, only 14 % of the 438 endpoints catalogued by the SPARQL endpoint status tool Sparqles[1] currently present a VoID description of their contents. Further, many of those that do lack the statistic spreads of the contents. Finally, some descriptions, most notably the official one of DBPedia[2], are heavily outdated and/or still follow an older, incompatible version of the VoID specification.

This may be a chicken-and-egg problem in the sense that despite the availability of automated tools [3,6] for offline dataset description creation, they are still not easy enough to use. Neither are there many tools that would make use of such data if it were available. The Aether web application presented here[3] aims to tackle both sides of this problem, by being able to automatically generate extended VoID statistical spreads from a SPARQL 1.1 endpoint, as well as by allowing such spreads to be viewed in a graphical interface.

More specifically, the original goal of the Aether tool was to be able to generate and visualize such descriptions of a dataset that a user encountering the

[1] http://sparqles.okfn.org/discoverability
[2] At http://dbpedia.org/void/Dataset
[3] online at http://demo.seco.tkk.fi/aether/, with code available at http://github.com/jiemakel/aether/

V. Presutti et al. (Eds.): ESWC Satellite Events 2014, LNCS 8798, pp. 429–433, 2014.
DOI: 10.1007/978-3-319-11955-7_61

dataset for the first time would be able to make sense of its content and general outlook. In addition, the tool also has features for comparing datasets, particularly useful for seeing how they change between versions. It has also become apparent that the tool can aid in detecting outliers and errors in a dataset, by e.g. highlighting subjects and objects with disproportionate amounts of references.

In the following, first some extensions and clarifications to the VoID vocabulary that were necessary for attaining the goals set for the tool are discussed. Then, the Aether tool itself is presented, along with example use cases.

2 Extended VoID Description

The VoID vocabulary [1] defines statistics properties for disclosing the number of triples, entities and classes in a dataset, as well as the number of distinct properties, subjects and objects. In addition, the vocabulary defines properties whereby a dataset may be split into class- or property-based partitions. Combining these, one is able to for example state that there are a certain number of triples with the property "foaf:name". In the Aether viewer interface, these combinations are grouped into bar charts by partition type, intuitively visualizing e.g. the top 50 properties with the most triples, as seen in Fig. 1.

For answering the goals set forth for Aether, the two partitions and six statistics defined in VoID are however insufficient. Indeed before, first the RDFStats tool [6] and later the LODStats project [3] have defined further dimensions and statistics for datasets. Particularly the LODStats extended set of 32 statistical criteria[4], deriving from (1) a survey of VoID and RDFStats statistics, (2) analysis

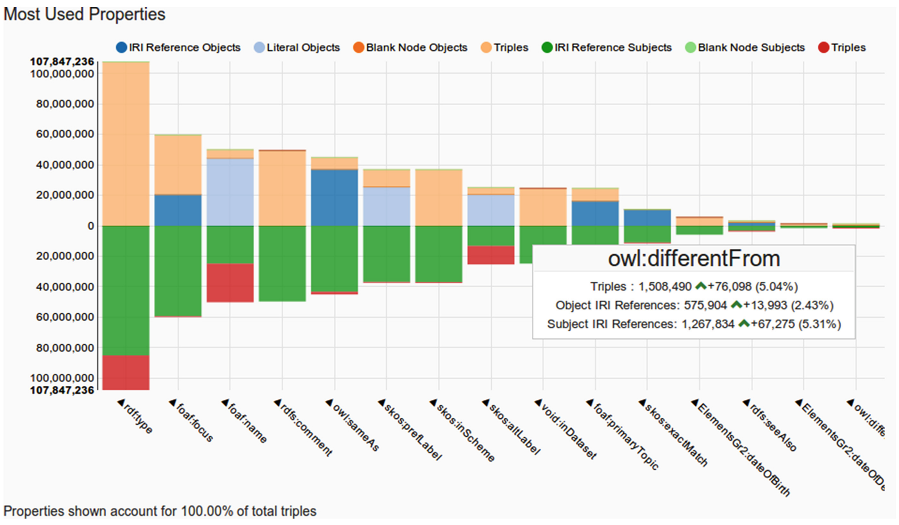

Fig. 1. Part of the Aether visualization interface showing property-related statistics compared between two different versions of the VIAF dataset

[4] https://github.com/AKSW/LODStats/wiki/Statistical-Criteria

of RDF data model elements and (3) expert interviews, seems to naturally serve as a great starting point for a comprehensive description of a dataset.

Upon further inspection however, it seems that while the criteria themselves are good, their formal serialization is problematic. First, the namespace IRIs do not resolve. Then, by looking at the SPARQL endpoint which does contain definitions, it seems that those definitions do not match the spirit of the RDF Data Cube vocabulary [7], on which they are nominally based on. Instead of defining the different statistical measures as properties, there is a single measure property of "ls-qb:value", with the different measurements defined as dimensions. On the other hand, none of the partitioning dimensional properties are defined (and indeed, none of the partitioned data is available as data from http://stats.lod2.eu/), even though it is visualized there.

Another observation about stats.lod2.eu is that its access mechanisms always return either just the VoID, or just the (partial) Data Cube descriptions for a dataset. This already tells that combining the two different statistics presentation and addressing mechanisms is not without problems. Based on this, it was decided that the description used by Aether should either fully function like VoID statistics and partitions, or like Data Cube slices and measurements. From the options, VoID was chosen as it is the current commonly acknowledged base.

For the purposes of the Aether tool, is was worth noting that the statistics properties of VoID fall in two camps: one concerning entities without regard to triples (the class partition and the entities and classes statistics) and the other concerning triples and their parts (the property partition and the distinct triples, properties, subjects and objects statistics). From the specification, it is unclear if or how these interact with each other (e.g. what would an entities count mean for a property partition, or a distinct object count for a class partition).

For Aether, this proved problematic as there was a desire to support a drill-down interaction, where the user could select any partition from the visualizations, with all visualizations then updating to contain only the contents matching that partition. To solve the problem, a triple-centric approach was taken, creating new statistics and partitions that explicitly relate the triple and entity worlds to each other. For example, separate partitions were created for subject, property and object classes, each only containing classes whose instances appear in the corresponding position in a triple belonging to the current partition.

In the end, this led to the creation of the void-ext vocabulary at http://ldf.fi/void-ext, which extends the VoID vocabulary with a total of 18 statistics and 14 partitions. In this paper, in place of discussing the properties individually, they are presented later alongside the discussion of the user interface.

3 The Aether Tool

The Aether tool itself divides into two functionalities. The first of these allows the creation of an extended VoID description from any SPARQL 1.1 conformant endpoint[5]. The feasibility of this approach depends on the SPARQL endpoint and

[5] The queries used can be read from the source starting at http://github.com/jiemakel/aether/blob/1.0.0/app/scripts/void/voidService.coffee#L230.

the various dimensions of the dataset. To give some examples, a dataset of 30 000 triples residing in a TDB-backed Fuseki SPARQL endpoint was processed in 7 s. Two datasets of some 200 000 triples containing mostly uniform instance data took 3 and 7 min, respectively. An instance dataset of 750 000 triples was processed in 45 min. On the other hand, about the same time was required to process an ontology of 350 000 triples. And when that same ontology was processed from a purely memory-backed model instead of a TDB-backed one, processing time skyrocketed to a whopping 12 h, due to the memory-backed model providing less useful statistics to Fuseki's SPARQL optimizer. The largest dataset thus far processed using the SPARQL queries, comprising of the 23 million triples of the Finnish edition of DBPedia 3.9, took almost 5 h.

For processing larger datasets, an offline tool can be used. Using this tool, a description of for example the 2.2 billion triples in the complete international version of DBPedia 3.9 was created in approximately 42 h. Unfortunately, the offline tool is tied to a custom triple store implementation that is not yet publicly available, and thus also neither is the tool.

The other part of the Aether tool is the visualization interface, of which a part was already depicted in Fig. 1. In whole, the interface is divided into a configuration selector, plus six parts divulging statistics about the dataset.

At the top of the interface is the configuration selector, which allows the reading of a VoID description from a SPARQL endpoint, or also actually querying that endpoint itself live for the requested statistics. A second endpoint or description can also be specified, to which the primary statistics will be compared.

The first part of the interface containing statistics is termed General Information. Shown are for example the total amount of triples and distinct RDF nodes, as well as the amount of classes and literal datatypes and languages. Distinct triple part counts are also given, as are pie charts on how the RDF nodes divide between blank nodes, IRIs and literals, along with bar charts showing the length distribution of IRIs and literals, respectively.

The next section of the interface is an auxiliary part termed Namespace Information, which associates all namespaces encountered with prefixes so that the rest of the visualizations can show only short forms of the IRIs.

Then follow four parts relating to subdivisions by triple part: property, resource object, literal object and subject. For sections other than literal objects, bar charts are drawn on divisions by most referenced IRI, namespace and class. For literal objects on the other hand, the most referenced literals, datatypes and languages are presented. In addition, the chart for most referenced properties contains additional embedded information, relating the individual properties to their corresponding numbers of distinct subjects, resource objects and literals.

All items shown in any visualization among these four parts is also clickable, causing the whole description to update to show only information pertaining to that partition (e.g. showing statistics constrained to only those triples where the property is from the foaf namespace). Double-clicking on the other hand causes a suitable live query to be launched against the SPARQL endpoint that is the source of the data, if one is available.

4 Examples of Use

Many of the visualizations have already been found useful for divulging interesting information about a dataset. For example, when viewing statistics about the European cultural aggregator Europeana, the RDF node counts show a huge number of IRI resources, but looking at the property graph, it can be seen that all primary metadata in Europeana is actually encoded as literals! The number of IRI references comes from applying the Europeana Data Model, which requires aggregate and proxy IRIs to be minted for each work. In fact, there seem to be only about 20 million objects in Europeana, even if there are 120 million IRIs. Most literals also do not seem to have appropriate language codes.

Comparing statistics for a dataset between versions on the other hand can tell about the growth of that dataset. For example, between 2014-02-15 & 2013-11-24, the Virtual Internet Authority File VIAF has grown by 178,673 persons. In the same 3 months, 15,884 new links to DBPedia have been created and 576,387 more entity identifiers have been deprecated (merged with others). Using the tool, one can also quickly find that the person with the most distinct names in VIAF is the 13th century Persian poet and mystic Rumi, who is now known by 419 names, 6 more than in November.

For more examples, as well as to see these ones live in the interface, head to the application at http://demo.seco.tkk.fi/aether/.

References

1. Alexander, K., Cyganiak, R., Hausenblas, M., Zhao, J.: Describing linked datasets with the VoID vocabulary. W3C Interest Group Note, March 2011
2. Alexander, K., Cyganiak, R., Hausenblas, M., Zhao, J.: Describing linked datasets. [8]
3. Auer, Sören, Demter, Jan, Martin, Michael, Lehmann, Jens: LODStats – an extensible framework for high-performance dataset analytics. In: ten Teije, Annette, Völker, Johanna, Handschuh, Siegfried, Stuckenschmidt, Heiner, d'Acquin, Mathieu, Nikolov, Andriy, Aussenac-Gilles, Nathalie, Hernandez, Nathalie (eds.) EKAW 2012. LNCS, vol. 7603, pp. 353–362. Springer, Heidelberg (2012)
4. Toupikov, N., Umbrich, J., Delbru, R., Hausenblas, M., Tummarello, G.: Ding! dataset ranking using formal descriptions. [8]
5. Görlitz, O., Staab, S.: Splendid: Sparql endpoint federation exploiting void descriptions. In: Hartig, O., Harth, A., Sequeda, J. (eds.): COLD. CEUR Workshop Proceedings, vol. 782. CEUR-WS.org (2011)
6. Langegger, A., Wöß, W.: Rdfstats - an extensible rdf statistics generator and library. In: Tjoa, A.M., Wagner, R. (eds.): DEXA Workshops, pp. 79–83. IEEE Computer Society (2009)
7. Cyganiak, R., Reynolds, D., Tennison, J.: The RDF data cube vocabulary. W3C Recommendation, January 2014
8. Bizer, C., Heath, T., Berners-Lee, T., Idehen, K. (eds.): Proceedings of the WWW2009 Workshop on Linked Data on the Web, LDOW 2009, Madrid, Spain, 20 April 2009 (Bizer, C., Heath, T., Berners-Lee, T., Idehen, K. (eds.): LDOW. CEUR Workshop Proceedings, vol. 538. CEUR-WS.org (2009)