Understanding Billions of Triples with Usage Summaries

Shahan Khatchadourian and Mariano P. Consens

University of Toronto
shahan@cs.toronto.edu, consens@cs.toronto.edu

Abstract. Linked Data is a way to share and consume interlinked semantic web datasets. Usage summaries can help to understand the structure within and across interlinked datasets by partitioning entities based on how they are described, such as grouping entities that are instances of the same types and described with the same predicates. Because Linked Data is growing to billions of triples, scalable techniques for generating usage summaries are essential.

In this work, we implement a novel Hadoop-based technique for generating usage summaries of billions of triples. We analyze and compare usage summaries generated for the *entire* BTC 2010 and 2011 datasets. We generate usage summaries involving classes and predicates, and of recommended patterns, such as for inferencing and interlinking.

1 Introduction

In order to continue to promote the production and consumption of datasets from the Linked Open Data (LOD)[1] cloud, it is worthwhile to understand what kinds of descriptions each dataset provides and how the descriptions between datasets interact. Patterns have been recommended in [3] for the publication and consumption of Linked Data; however, as the size of Linked Data grows to many billions of triples, challenges that affect scalability arise.

Usage, proposed in [2], is a way to characterize the semi-structure of semantic web datasets based on how an entity is used, such as whether it is an instance or part of the schema (like a class or predicate). ExpLOD's [4] usage summaries extend this notion by succinctly capturing how the actual *descriptions* are used, such as how sets of classes and predicates are used in conjunction. When applied to datasets from the LOD cloud, usage summaries reveal *all* the unique and varied descriptions that occur *within* a dataset as well as *across* interlinked datasets. However, ExpLOD's implementation was bound to datasets that fit in main memory, limiting the size of datasets for which usage summaries could be generated. In this paper, we propose a novel, scalable mechanism to generate usage summaries of billions of Linked Data triples.

[1] http://linkeddata.org/

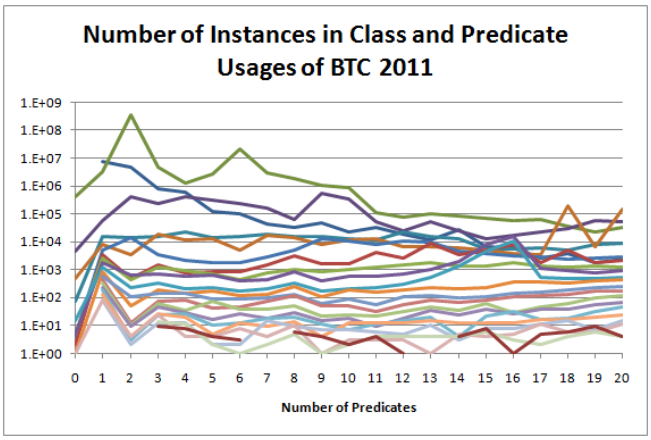

Fig. 1. Instances in class and predicate usage summary of BTC 2011

Related Work Analysis of the BTC dataset in previous years has generally relied on statistics based on popularity of single entities, such as ranking each class or predicate based on the number of instances that it has been used to describe. Such statistics do not show how different classes and predicates interact and, as such, miss the opportunity for users to understand the actual descriptions and how they vary both within and across different datasets.

Our novel approach differs from previous efforts in several ways. First, we extract *all* the unique descriptions and are not limited to a specific domain as in [5]. Second, by employing a flexible and scalable usage summary mechanism, we generate and compare many usage summaries of billions of triples, and is not limited to portions of the dataset like in [1]. Our work goes well-beyond previous efforts by processing the BTC 2010 and 2011 datasets in their *entirety*.

Contributions Our contributions in this work are as follows:

1. We generate and compare numerous usage summaries of the BTC 2011 and BTC 2010 datasets.
2. Using several patterns described in [3] that are applicable to publishing and consuming Linked Data, such as for interlinking or inferencing, we generate usage summaries of these patterns to show how they are used in varied descriptions within the BTC datasets.
3. A descripion of our scalable, Hadoop-based implementation that can generate usage summaries over datasets containing billions of triples.

2 Usage Summaries

In our previous work, ExpLOD [4], we proposed creating summaries of usage neighbourhoods as a way to help a user's understanding of datasets from the

Linked Open Data cloud. In this section, we introduce the reader to usage, usage neighbourhoods and usage summaries.

A *usage neighbourhood* is a set of entities that are *used* as part of a semantic web description, such as whether an entity is an instance, part of the schema (such as a class or predicate), or acts to create interlinks; they can also overlap such as predicates used for interlinking. For example, an instance that is of the single type *Person* has the *class* usage neighbourhood consisting of the singleton set {*Person*}. As another example, an instance that is a type of more than one class has a class usage neighbourhood that is the set of classes that it is a type of, such as {*Person, Organizer*}. Usage neighbourhoods can involve multiple usages, such as class and predicate usages (what predicates are used to describe the instance). Usage neighbourhoods help dataset consumers understand how usages occur in conjunction and contribute to the semantic web descriptions contained within the dataset.

Instead of exploring the usage neighbourhoods in the dataset on an instance-by-instance basis, which can be substantial if the dataset is sizeable, it is more convenient to group those instances from the dataset that have the same usage neighbourhood into a *usage summary*. For example, a class and predicate usage summary groups those instances that have the same class usage (the instances are of the same types) and predicate usage (the instances are described with the same set of predicates). Two instances with class usage {*Person*} that have predicate usage {*homepage*} will be grouped together as they have the same class and predicate usage, but an instance with class usage {*Person, Organizer*} and predicate usage {*homepage*} will be placed in a different group because it has a different class usage. An advantage of usage summaries is that they allow a user to capture and understand how instances are described.

An RDF dataset graph is a graph with directed edges and labeled nodes created by constructing a node for each distinct URI that occurs as a subject or object in a triple, as well as a unique node for each statement's predicate. Technically, a usage summary is a partition of a set of subgraphs from the dataset graph. The subgraphs in each set are *bisimilar*, that is, the subgraphs in a set are pairwise "equivalent" and not with any subgraphs outside of their set. Specifically, two subgraphs are bisimilar if they each have a node with the same label, and recursively, if the labels of nodes connected by outgoing edges are also the same. We simplify the creation of different usage summaries by employing the flexibility of a *bisimulation label*, a function that applies labels to an RDF dataset graph. A usage summary is created by first constructing an RDF dataset graph from the input dataset, applying the bisimulation labels to the nodes, extracting relevant subgraphs that pertain to usage neighbourhoods, then partitioning the subgraphs using bisimulation.

We adopt a slight variation to the bisimulation label proposed in [4] by using "+" after the usage prefix, and "|" between the graph hostname and entity URI. The bisimulation labels we use are based on a combination of the following: (1) Usage prefix: "C" for a class, "P" for a predicate. (2) Graph URI hostname: We consider only the main domain of a statement's graph URI. For example, we use

bestbuy.com from the URI
http://products.semweb.bestbuy.com/products/16293707/semanticweb.rdf. Techniques described in [1] can be used to ascribe instances to datasets. (3) Entity URI: We use the entity URI as is, e.g., http://xmlns.com/foaf/0.1/homePage.

Bisimulation labels are applied to the dataset graph by examining subgraphs to determine each entity's usage. For example, the object of a statement with predicate *rdf:type* is used as a class, and the subject is an instance of that class, which generates a node labeled with the object URI and prefixed with "C", a node labeled "P+rdf:type" for the predicate, and a node with the instance URI. For example, using a bisimulation label constructed from the concatenation of the usage and the entity URI (which we show prefixed here), the two entities described in Section 2 have the following class and predicate usage neighbourhoods:

- {C+Person, C+Organizer, P+rdf:type, P+homepage}
- {C+Person, C+Organizer, P+rdf:type}

Table 1 shows the summaries we have generated and the bisimulation labels they consider.

Summary	usage	graph	entity
usage	x		
graph	x	x	
nbr	x	x	x
interlink	x	x	only if owl:sameAs, rdf:seeAlso, or skos:exactMatch
index	x	x	only if rdf:Seq or rdf:List
inference	x	x	only if skos:broader or skos:narrower
foaf	x	x	only if foaf:*
skos	x	x	only if skos:*
topic	x	x	only if like foaf:topic or foaf:PrimaryTopic

Table 1. Summaries considered and their bisimulation label

3 Usages in Billions of Triples

In this section, we describe the semi-structure of the 2011 BTC dataset with several different types of usage summaries, and include comparisons to usages in the 2010 BTC dataset.

3.1 Class and Predicate Usages

A class and predicate usage summary is an easy-to-understand and informative summary of how classes and predicates are used with each other for describing instances by capturing the unique combinations of their occurrences. In terms of class and predicate usages in the BTC 2010 and BTC 2011 datasets, the 406,170,030 distinct instances in BTC 2010 are described with 3,281,294 distinct usages. Meanwhile, the number of distinct instances in BTC 2011 has decreased to 390,358,846, and which are correspondingly described with 2,790,334 unique class and predicate usages.

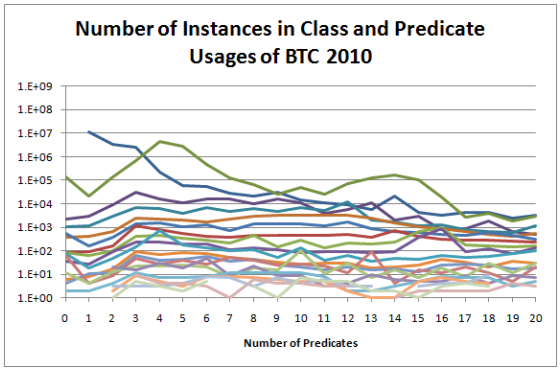

Fig. 2. Instances in class and predicate usage summary of BTC 2010

Figures 1 and 2 shows the number of instances in the class and predicate usages of BTC 2010 and 2011. Each series line represent the class and predicate usages with a fixed number of classes, the top-most line happens to have exactly 0 classes in its class usage, and the lines underneath have decrementally fewer classes; the bottom-most series line represents the class and predicate usages with exactly 20 classes. The graph shows that, as the number of classes increase, the number of instances decrease, and also that the number of instances decrease as the number of predicates increase. When comparing the 2010 and 2011 datasets, we see that the trends appear similar, except that in 2011, the number of instances that are described with around 2 predicates has increased.

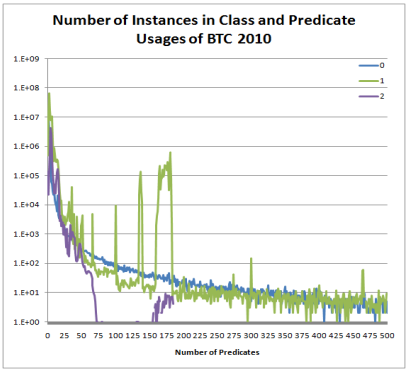

Fig. 3. Tail of BTC 2010

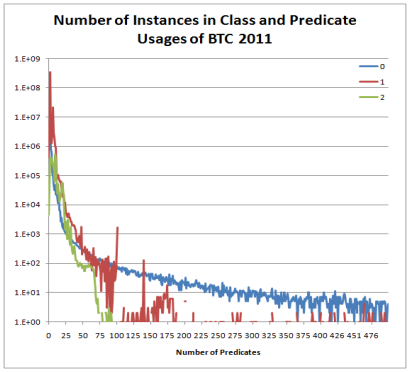

Fig. 4. Tail of BTC 2011

Notice that the two datasets contain usages with 0 classes that have a substantial number of instances described with 20 predicates. In fact, at least 229 predicates are needed before the number of instances in usages with 0 classes have fewer than 10 instances. Figures 3 and 4 shows the number of instances in usages with 0,1, and 2 classes with up to 500 predicates. We note that class and predicate usages with 0 classes have a long tail, while applying 1 or more classes does not have a similar, stable tail. BTC 2010 also has a prominent number of instances that are described using 1 class and fewer than 200 predicates, but which trend does not appear in BTC 2011.

When considering the origin of triples (captured with the graph hostname in the bisimulation label), the most popular class and predicate usage neighbourhood in both BTC 2010 and BTC 2011 is:
[C+hi5.com|foaf:Person, P+hi5.com|rdfs:seeAlso, P+hi5.com|foaf:nick], which indicates that the most commonly occurring description in the datasets are about instances described by the hi5.com graph hostname that are of type *foaf:Person*, are described only by *foaf:nick* predicates, and also have an interlinking predicate *rdfs:seeAlso*. In fact, 340,583,367 instances in the BTC 2011 dataset have this class and predicate usage neighbourhood.

3.2 Usages of Linked Data Patterns

In this subsection, we explore patterns that have been described in [3]. Since we do not consider value-based equivalence, such as the two literals "1" and "1.0", we exclude identifier-based patterns, as discussed in. Instead, we focus on *structure-based* patterns. The bisimulation labels we use for the following summaries were shown in Table 1, and take into consideration the originating graph hostname of usages.

The patterns we explore in this work are the following:

- Topic Relation, Composite Descriptions: where the predicate is like foaf:topic or foaf:primaryTopic. We also examine the more general case of FOAF schema usage as a relevant schema for describing instances in general.
- Link Base, See Also, Equivalence links: where the predicate is one of owl:sameAs, skos:exactMatch or rdfs:seeAlso
- Materialize inferences: where the predicate is like skos:broader and skos:narrower. We also examine the more general case of SKOS schema usage as a relevant schema for triples expressing inferences.
- Index Resources: where the instance is typed as a rdf:List or rdf:Seq

	Usages		Instances	
	2010	2011	2010	2011
Inference	120	256	68,672	842,938
Interlink	2,824	2,212	69,563,476	363,104,656
Topic	2,380	6,223	222,267	1,071,213
Index	12,324	60	710,972	769,138

Table 2. Pattern summaries of BTC 2010 and BTC 2011

Table 2 shows the number of distinct usages and instances involved in the summaries described above compared between BTC 2010 and BTC 2011. For example, the first row of the table says that in the inference usage summary, BTC 2010 contains 68,672 instaces divided amongst 120 distinct usage neighbourhoods, while BTC 2010 contains 842,938 instances divided amongst 256 distinct usages, a substantial increase; topical descriptions also experience an increase in usages and instances. The number of interlinked instances has increased by 5 times despite being expressed with fewer unique usages. We also recognize that

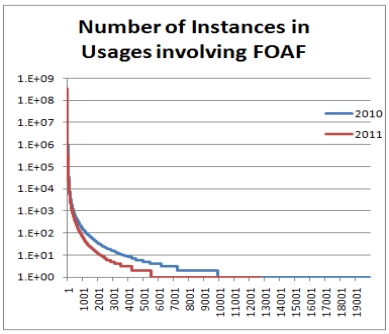

Fig. 5. FOAF schema usage

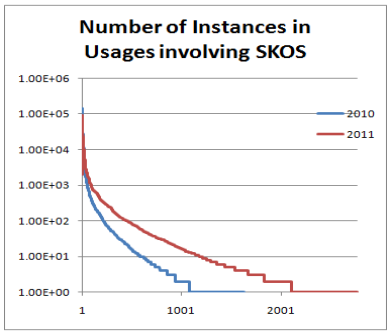

Fig. 6. SKOS schema usage

the number of indexed resources remains similar, but are expressed using much fewer unique usages.

Figure 5 shows the number of instances in that usages involving the FOAF schema for the BTC 2010 and BTC 2011 datasets. It reveals that BTC 2011 has fewer distinct usage neighbourhoods as well as fewer instances than in BTC 2010. This could be as a result of a shutdowns of social networking sites such as vox.com, which was the eleventh-most common usage FOAF usage neighbourhood in BTC 2010, but removed from BTC 2011. Figure 6 shows SKOS usage in the BTC 2010 and BTC 2011 datasets, and reveals that the BTC 2011 dataset has increased the number of distinct usage neighbourhoods and describes more instances.

4 Implementation

For simplicity, we first generate the most detailed summary, the *nbr* summary specifically, which is based on the bisimulation label that concatenates the usage prefix, graph hostname, and entity URI. We then generate coarser summaries (that use include fewer details in their bisimulation label). Although we have a basic SPARQL-based implementation using Jena, which we intend to describe as future work, we implented the summaries using low-level string- and line-based parsing. We did this for economic reasons - processing graphs takes at least an order of magnitude longer.

The *nbr* summary, from which all other summaries were generated, is computed using two Hadoop jobs. The first job takes a BTC dataset, GZipped NQuad files read using the NxParser library[2], as input, and computes the *nbr* usage neighbourhood for each distinct subject in the dataset; each unique blank node identifier that appeared as a statement's subject was also included. The job outputs GZipped tab-separated files containing the subject URI, and its nbr usage neighbourhood.

The second job takes the output of the first job and switches the key and value, so that the reduce task groups subjects by their usage neighbourhood (using string comparison). The output of this job is the *nbr* usage summary, which is stored as compressed tab-separated files containing each distinct usage

[2] `http://code.google.com/p/nxparser/`

neighbourhood and the number of instances with that usage neighbourhood. These two jobs completed on 40 Amazon Elastic MapReduce[3] large instances in under 40 minutes each. Since the nbr summary output is only 10% of the original dataset size, it becomes feasible to analyze the remainder of summaries locally. Other summaries are generated using a similar approach.

5 Conclusions

In this work, we have described a scalable technique to generate usage summaries of Linked Data containing billions of triples. Using flexible bisimulation labels, we generated and compared several usage summaries to aid understanding of how classes and predicates are used in the entire BTC datasets. Additionally, we have reported the variations of patterns that have been recommended for interlinking and inferencing.

A Requirements

Our work, which uses the entire BTC 2010 and 2011 datasets, drives an analysis of Linked Data using increasingly more detailed usage summaries. Usage summaries that describe how instances, classes, and predicates are used together benefit Linked Data consumers by giving them insight to the descriptions contained in the dataset. In addition, recommended publication patterns are also explored as usage, showing the ease and flexibility of our approach. Since our implementation is Hadoop-based, it is scalable; however, processing these datasets required only a few dollars worth of initial investment to generate the most detailed summary desired, after which coarser summaries were generated locally, each summary taking generally taking less than an hour to compute. We also generate usage summaries based on properties relevant to real-world semantic web applications, such as interlinking and inferencing. Our results will be available online at `http://www.cs.toronto.edu/~shahan/swc2011/`.

References

1. C. Böhm, J. Lorey, D. Fenz, E. Kny, M. Pohl, and F. Naumann. Creating voiD descriptions. Semantic Web Challenge 2010.
2. L. Ding and T. Finin. Characterizing the semantic web on the web. In *ISWC*, pages 242–257, 2006.
3. L. Dodds and I. Davis. Linked data patterns. `http://patterns.dataincubator.org/book`, Sept. 2011.
4. S. Khatchadourian and M. P. Consens. ExpLOD: Summary-based exploration of interlinking and RDF usage in the linked open data cloud. In *ESWC (2)*, pages 272–287, 2010.
5. G. T. Williams, J. Weaver, M. Atre, and J. A. Hendler. Scalable reduction. Semantic Web Challenge 2009.

[3] `http://aws.amazon.com/elasticmapreduce/`