High Performance Descriptive Semantic Analysis of Semantic Graph Databases*

Cliff Joslyn[1], Bob Adolf[1], Sinan al-Saffar[1], John Feo[1], Eric Goodman[2], David Haglin[1], Greg Mackey[2], and David Mizell[3]

[1] Pacific Northwest National Laboratory
[2] Sandia National Laboratories
[3] Cray Inc.

Abstract. As semantic graph database technology grows to address components ranging from large triple stores to SPARQL endpoints over SQL-structured relational databases, it will become increasingly important to be able to understand their inherent semantic structure, whether codified in explicit ontologies or not. Our group is researching novel methods for what we call *descriptive semantic analysis* of RDF triplestores, to serve purposes of analysis, interpretation, visualization, and optimization. But data size and computational complexity makes it increasingly necessary to bring high performance computational resources to bear on this task. Our research group built a high performance hybrid system comprising computational capability for semantic graph database processing utilizing the multi-threaded architecture of the Cray XMT platform, conventional servers, and large data stores. In this paper we describe that architecture and our methods, and present the results of our analyses of basic properties, connected components, namespace interaction, and typed paths of the Billion Triple Challenge 2010 dataset.

Keywords: Semantic graph databases, high performance computing, semantic networks.

1 Introduction

As semantic graph database (SGDB) technology grows to address components ranging from extant large triple stores to SPARQL endpoints over SQL-structured relational databases, it will become increasingly important to be able to understand their inherent semantic structure, whether codified in explicit ontologies or not, for tasks ranging from analysis, interpretation, and visualization to optimization. But the ability to understand the semantic structure of a vast SGDB awaits both the development of a coherent methodology and the high-performance computational platforms within which to exercise such methods.

A number of factors make SGDB problems different from those where network science and graph theoretical methods are typically applied. Perhaps most prominently, their formal nature are as structures which are not only large graphs, but have high data complexity in that they are typed and directed networks: types on nodes and links carry the specifically *semantic* information of their assertions, while the directionality of the links indicates the argument

* Corresponding author: Cliff Joslyn, Pacific Northwest National Laboratory, cjoslyn@pnl.gov, 206-552-0351.

structure of the links, seen as predicates. But standard methods in network science (e.g. connected components, minimum path, centrality, etc.) have generally been developed for networks of large size but low data type complexity, that is for untyped, and undirected graphs.

Where such methods ignore semantics in order to reduce complexity, it is becoming increasingly important to develop methods that tackle high data complexity directly. This paper describes some novel methods for analyzing such semantic structures in graph data, and their significance on large graphs. To that end, our research group built a novel high performance hybrid system comprising computational capability for semantic graph database processing utilizing high capacity standard servers together with the multi-threaded architecture of the Cray XMT platform. We have brought these capabilities to bear on the 2010 Billion Triple Challenge[4] dataset (BTC10) [6].

In this paper we describe these systems and our work to interrogate BTC10 with respect to its large-scale semantic structure. We first describe our hybrid computational platform and the Cray XMT machine at its core. We then provide base statistics on BTC10 node and link types and namespaces, including factoring the ontological semantic meta-data from the `rdf`, `rdfs`, and `owl` namespaces. We then consider interaction among namespaces in BTC10, and analyze the connected component structure of BTC10, with and without semantic filtering. We then perform semantic analysis over classes and predicates, building up a statistical ontological map. Finally we factor BTC10 according to network motifs which are short, typed paths, specifically link type bigrams and trigrams. This analysis reveals the inherent semantic structure of BTC10.

There is some previous work related to some of the mapping efforts we use here for namespaces [5, 7, 8], and performing network scientific analyses of large graph data [8], including connected components. We understand our work to be novel both in looking at the predicate sensitivity of connected components, its ability to do statistical mapping down to the predicate level, its statistical modeling approach to graph motif analysis, and of course its scaling to graphs with billions of edges.

2 High-Performance Computational Architecture

Our high-performance computing platform includes a Cray XMT and a high-end server. We use the high-end server—with 48 GBs of memory and two quad-core 2.96 GHz Intel Xeon CPUs—to perform initial investigations into the BTC10 using both leading commercial triple store software and custom software to perform scans of the data with regular memory accesses.

But for problems, such as graph problems, which are dominated by unpredictable memory references, that is, with almost no locality, the Cray XMT can significantly outperform distributed-memory parallel architectures based on commodity processors. The XMT also has a significant amount of shared memory (1024 GBs) so that the entire graph can fit into memory at once, obviating the usual requirement of paging data into limited RAM.

[4] http://www.cs.vu.nl/~pmika/swc/submissions.html

Our Cray XMT has 128 custom *Threadstorm* processors, each of which supports 128 hardware thread contexts, so that each Threadstorm can be viewed as a 128-way hyperthreaded processor. For unpredictable memory reference patterns, cache memory is ineffective. To overcome the latency of memory references with no cache hits, programs are designed and written for high amounts of concurrency. Thus at any time, each of the Threadstorms is likely to have at least one of its 128 threads ready to compute while other threads await arrival of data from memory. This architecture is designed for running programs with large memory footprints and 12,000 threads in a single program. With 1TB of shared memory, we were not memory-constrained in our processing of the BTC10 data.

The amount of parallelism in applications running on the XMT can only be supported by fine-grain synchronization support in the hardware and runtime systems. It is commonly understood that fundamental data structures need to be specialized to run on a system with this much concurrency. Members of our team have recently developed hashing data structures that are used extensively in our BTC10 work [4].

Our productivity in exploring BTC10 on the Cray XMT was facilitated by two open source libraries that specifically target the Cray XMT: the Multi-Threaded Graph Library (MTGL)[5] and the Semantic Processing Executed Efficiently and Dynamically (SPEED-MT)[6] library. The first is a set of graph algorithms and data structures designed to run scalably on shared-memory platforms such as the XMT. The second is a novel scalable Semantic Web processing capability being developed for the XMT.

We used a character string tokenization package from the SPEED-MT library to translate the verbose BTC10 data into 64-bit integers, to increase computational efficiency and reduce the memory footprint. The XMT's large global memory allowed us to hash each URI, blank node, or literal into a shared hash table and assign each a unique integer identifier. The process of translating from strings to integers took a total of 1h 35m, with 75% of the time being file I/O.

3 First-Pass Semantic Data Analysis

We acquired BTC10 and verified it as an RDF graph with 3.2B $\langle s, p, o, q \rangle$ quads, which we projected to 1.4B unique $\langle s, p, o \rangle$ triples, ignoring the quad field (useful for provenance and other operations but not for analyzing the main content).

We identified duplicates by hashing the triples, now of integers, into a shared hash table in 10 min. 37 s. A parallel for loop iterated over the triples, inserting them into a hash table class that is part of MTGL. The hash table class implements a thread synchronization method described in more detail in [4] that scales effectively for both uniform and power law distributions. Hash class collisions are handled with linear probing. The entire process of converting the data from string to integers, removing the quad field, and deduplicating compressed BTC10 from 624 GBs to 32 GBs. This does not include the mapping file back to the integers, which would be another 21.5 GBs.

[5] https://software.sandia.gov/trac/mtgl
[6] https://software.sandia.gov/trac/MapReduceXMT

Abbreviation	Prefix
bestbuy:	http://products.semweb.bestbuy.com/company.rdf
dgtwc:	http://data-gov.tw.rpi.edu/2009/data-gov-twc.rdf
fao:	http://www.fao.org/aims/aos/languagecode.owl
foaf:	http://xmlns.com/foaf/0.1/
freebase:	http://rdf.freebase.com/ns/
geonames:	http://www.geonames.org/ontology#
geospecies:	http://rdf.geospecies.org/ont/geospecies
linkedmdb:	http://data.linkedmdb.org/resource/oddlinker/
owl:	http://www.w3.org/2002/07/owl
purl:	http://purl.org/dc/elements/1.1/
rdf:	http://www.w3.org/1999/02/22-rdf-syntax-ns
rdfs:	http://www.w3.org/2000/01/rdf-schema
sioc:	http://rdfs.org/sioc/ns
skos:	http://www.w3.org/2004/02/skos/core

Table 1. Some prefix abbreviations used.

Some of the namespace abbreviations used below are shown in Table 1. Complete documentation of all the abbreviations used is too cumbersome for publication, but are straightforward and should be able to be determined from context.

We measured BTC10's very low graph density of 1.8 $\times 10^{-8}$ links/node^2. The left of Table 2 shows the distribution of the top 20 of the 58.6M non-blank subjects present, comprising only 0.085% of all 960.7M non-blank subjects; the right shows the distribution of the top 20 of the 95.5M non-blank, non-literal objects present, comprising 29.3% of all 429.5M non-blank, non-literal objects. As with namespaces, URIs are also abbreviated and summarized to assist with meaningfulness; full details are available in the BTC10 dataset.

Note that there are far fewer subjects than objects, by two orders of magnitude, indicating a much larger in-degree of objects compared to out-degree of subjects. The most prevalent subjects are "containers", each (e.g. Best Buy) pointing to a single category of a large number of objects (e.g. Offers in this case). The subject list also contains web documents and blogs pointing to different editors, a dataset container pointing to all the comprising files, and an ontology to all its edges and nodes with hasEdge and hasLink predicates. On the other hand most prevalent objects are types, virtually all (e.g. `foaf:Person`) due to these types being the destination of `rdf:type` predicates. Another explanation for an object having a high in-degree is that it represents a global service to many entities. Such is the case with `http://www.last.fm`, an account servicing page for nodes of type `foaf:OnlineAccount`.

Namespaces which deal with "semantic meta-data", or ontological typing information, are generally prominent. Specifically the `rdf:, rdfs:`, and `owl:` namespaces comprise 20.0% of all link instances. A histogram of the top 10 is shown in Table 3. These are dominated by `rdf:type, rdfs:seeAlso`, and `rdfs:label`, with `rdf:type` alone comprising 10.7%.

Reification is prominent in BTC10. There are 4.4M edges of predicate `rdf:subject` between `rdf:Statement` and `core:protein`, the main source. Overall, `rdf:subject, rdf:object`, and `rdf:predicate` each have 6.1M instances precisely, comprising a significant fraction of BTC10 that is reified.

Subjects	K	*Objects*	M
bestbuy:BusinessEntity_BestBuy	412.6	foaf:Person	68.39
data-gov dataSet-91	148.0	foaf:OnlineAccount	10.15
data-gov dataSet-90	54.0	rdf:Statement	6.06
geonames:United Kingdom	16.1	foaf:Document	4.67
geonames:Iowa	15.9	rss:item	4.65
geonames:Wisconsin	15.8	dgtwc:DataEntry	4.00
geonames:North America	15.8	dcmitype:Text	3.22
geonames:Minnesota	15.8	geospecies:Point	2.45
geonames:Michigan	15.8	opfield:Neighbour	2.36
liveJournal Entry 1	12.0	MusicOnt:Performance	2.27
liveJournal Entry 2	11.2	timeline:Interval	2.26
liveJournal Entry 3	10.9	event:Event	2.26
SMIL Webpage	9.3	geonames:Feature	2.21
liveJournal Entry 4	9.0	www.last.fm/	1.67
liveJournal Entry 5	8.8	wordnet:Person	1.66
Prefixcc Webpage	8.8	foaf:chatEvent	1.66
www.nettrust-site.net/fdic	8.4	uniprot:classifiedWith	1.56
fao:Language Codes Ontology 1	7.8	rdfs:seeAlso	1.54
sfsu:FoodsWebs Ontology	7.7	uniprot:Domain_Assignment_Statement	1.50
fao:Language Codes Ontology 2	7.2	goodrelations:ProductOrServiceModel	1.46

Table 2. (Left) Top 20 subjects (thousands); (Right) top 20 objects (millions).

4 Namespace Interaction

To understand the relationships between the sources which generated the dataset, we explore a summary metric for linkages among namespaces. Projects like the Linking Open Data initiative[7] and the Comprehensive Knowledge Archive Network[8] rely on manual attribution and curation of provenance. In BTC10 we must use an approximation method for attributing triples to sources. While conceptually URI's and their namespaces should only serve to provide a unique identifier, in practice namespaces can be used for clustering and developing a basic understanding about the sources of the data.

p_1	Count (M)
rdf:type	152.8
rdfs:seeAlso	86.7
rdfs:label	10.8
rdf:subject	6.1
rdf:object	6.1
rdf:predicate	6.1
owl:sameAs	4.7
rdfs:comment	4.1
rdfs:subClassOf	1.7
rdfs:isDefinedBy	1.3

Table 3. Top ten semantic meta-data link types (millions).

We call triples "linked" when two or more of the subject, predicate, or object map to different fully-qualified domain names (FQDNs). Table 4 shows the sources of linked data for the top 50 FQDNs as broken down by pair-wise position relationships. This data shows that a majority of triples in the BTC10 data use at least one entity created by a different organization, but most of this interlinking stems from the reuse and sharing of predicates. This entire process, including FQDN extraction, individual-, and pair-wise relationship counts, was computed in just over half an hour using 64 processors on the XMT.

5 Connected Components

In the previous section we discussed how prefixes can be used to understand the interconnectedness of the BTC10 graph. In this section we discuss a more

[7] http://linkeddata.org
[8] http://ckan.net

Relationship	Distinct FQDNs		Identical FQDNs		Literal or Blank	
Subject-Predicate	1976.0 M	62%	464.9 M	14%	725.1 M	22%
Subject-Object	528.9 M	16%	1997.5 M	63%	620.3 M	19%
Predicate-Object	1313.5 M	41%	1776.8 M	56%	59.5 M	1%

Table 4. Sources of cross-linking by entity position

graph theoretic approach: connected components. This approach is used to find the set of maximally connected subgraphs within a larger graph. For instances where there is one large connected component that encompasses the majority of vertices, a technique that works well is to first run breadth-first search, starting at the node with the largest out-degree, to find a large component. We then find the remaining components by using a "bully-strategy" [1].

To pose the BTC10 data in terms of connected components, we treat subjects and objects as vertices in a graph, and the predicates as edges connecting them. However, connected components is generally only applied to undirected graphs, so we ignore the directionality of the predicates. Running connected components on BTC10, we find that there are 208.3K components, with a giant component of 278.4M vertices, or 99.8% of the total.

To gain a better understanding of the structure of the graph, we experimented by iteratively removing edge types. We first removed ontological information by incrementally deleting the top 10 `rdfs:` predicates and the top seven `owl:` predicates. We also examined deleting in stages the overall top 25 predicate types. However, while we did see an increase in the number of components, a large component continued to dominate, rarely straying below 90% of the graph. In fact the process was more akin to shedding the leaf nodes of the graph, as the order of the graph diminished to half of the original.

This process did illuminate several large jumps in the number of components when certain edges were removed. Deleting only these predicate types, namely `rdf:type`, `rdfs:subClassOf`, `rdfs:isDefinedBy`, `owl:imports`, and `foaf:knows`, we arrived at 9.0M components with the largest comprising 81.1% of the induced graph, while only losing about 1% of the original vertices.

Finally, we performed a more extensive semantic filtering, in particular the following five steps:

1. Retracted `owl:sameAs` cliques to a single new meta-node
2. Removed reification definitions, specifically triples where $p =$ `rdf:predicate`, `rdf:subject`, `rdf:object`, or `rdf:Statement`.
3. Removed reification itself in addition to its definition, that is, all paths reaching to and from the reifying node of type `rdf:Statement`.
4. Removed edges where o is a literal.
5. Removed all edges where $p =$ `rdf:type`.

This procedure resulted in producing 980.7K components, with the largest component now only 111.3M nodes, or 54.95% of the total. The distribution of the sizes of the top 10 components in shown in Table 5.

Connected components on the XMT achieved 46x speedup from 1 to 128 processors, with computation time for 128 processors 10.3 seconds (see Fig. 1).

Component #	Size (M)	%
1	111.3	57.8%
2	73.6	38.2%
3	3.8	2.0%
4	1.7	0.9%
5	0.5	0.3%
6	0.4	0.2%
7	0.3	0.1%
8	0.2	0.1%
9	0.1	0.1%
10	0.1	0.0%

Table 5. Sizes of the top ten components after semantic filtering.

6 Class Analysis and Extant Ontology

The semantic structure of meaningful RDF triples is illustrated in Fig. 2, where triples $t = \langle s, p, o \rangle$ are shown as directed edges from s to o with the label p, or $s \xrightarrow{p} o$. t is cast as a logical predicate of the form $t = p(C_s, C_o)$, where the RDF predicate p is a relation $p \subseteq C_s \times C_o$ on classes C_s, C_o. Thus a basic semantic analysis requires looking at the distribution of the classes $C = C_s \cup C_o$ (noting that resources appear on both sides of predicates), and of the predicates p. We will conclude by building a *statistical ontological map*, or an *extant ontology*, as a directed graph on nodes as classes C, and edges as predicates p.

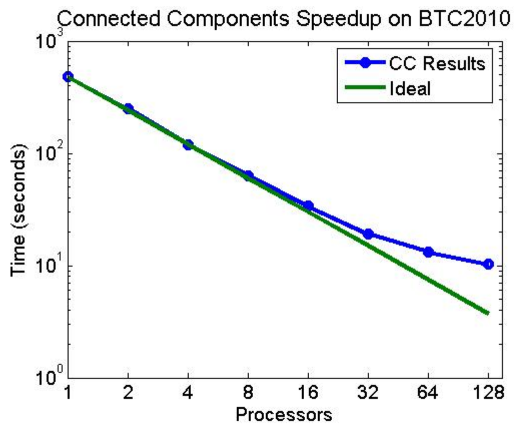

Fig. 1. Scaleup of component calculations.

BTC10 contains 168K different classes C but only a small number of those classes are widely used in the data. Fig. 3 shows that 16 of the most frequent classes would cover 80% of the used types while we can cover 95% of the data if we use 64 classes only. The top 16 classes are shown in Fig. 4.

Fig. 5 shows the top 16 of the 95.2K predicates, comprising 35% of all 1.4B link instances, as shown by the cumulative percentage line. The cumulative predicate coverage of Fig. 5 is extended to the first 350 predicates as represented by the solid line in Fig. 6. The 64 most frequent predicates cover about 50% of the data. In a second run we removed the edges leading to terminal nodes, as they do not link graph nodes but are rather node properties. We then re-calculated the cumulative predicate distribution and obtained the dashed line in Fig. 6.

$$\begin{tikzcd} s \arrow[r, "p"] \arrow[d, "\texttt{rdf:type}"'] & o \arrow[d, "\texttt{rdf:type}"] & & & \\ C_s & C_o & \Longrightarrow & C_s \arrow[r, "p"] & C_o \end{tikzcd}$$

Fig. 2. The semantic structure of rdf triples.

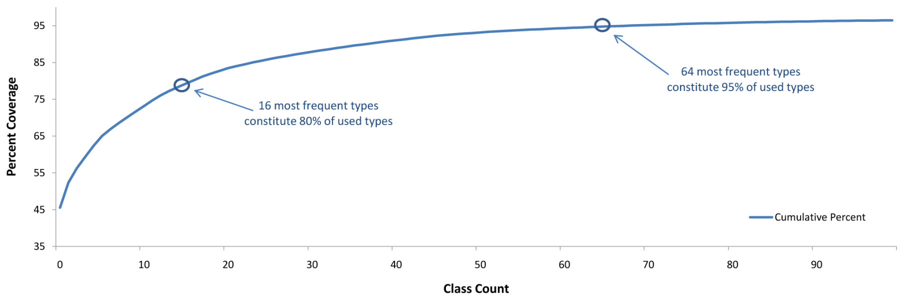

Fig. 3. Cumulative class distribution in BTC10.

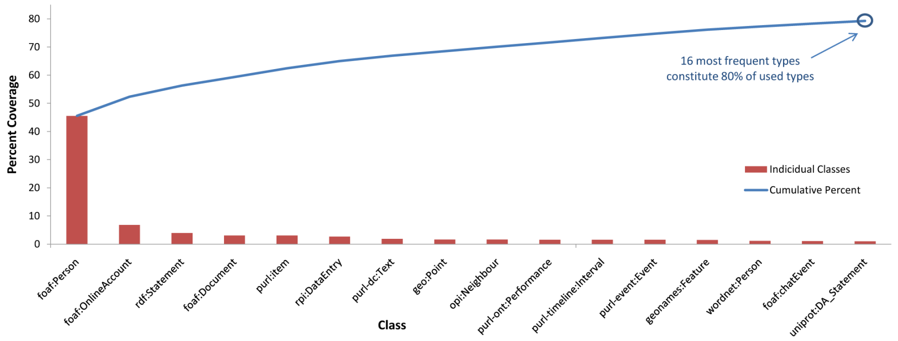

Fig. 4. Top 16 classes in BTC10.

Removing edges leading to literals significantly reduced graph size to 37% its original size (from 1.4B to 530M edges).

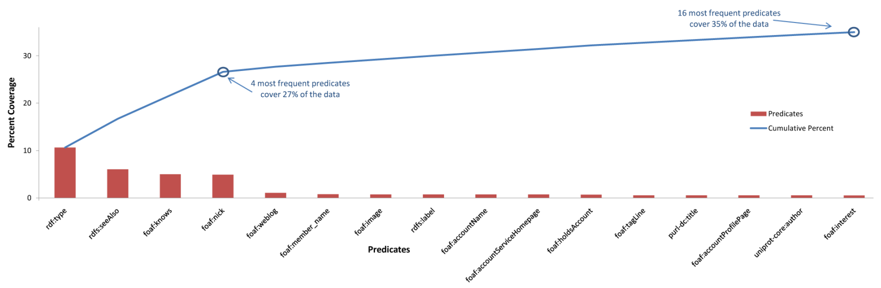

Fig. 5. Top 16 predicates in BTC10.

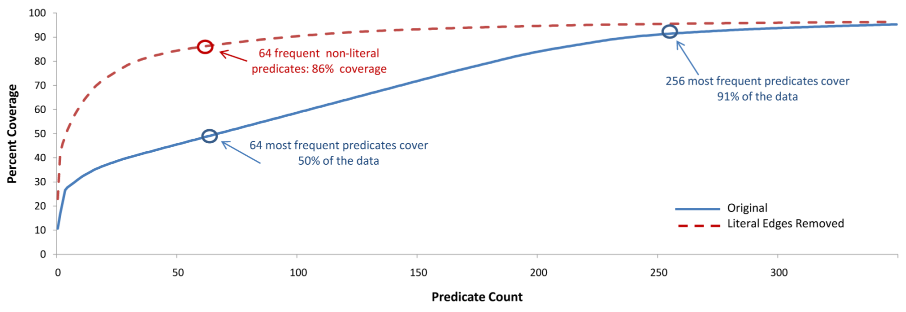

Fig. 6. Cumulative distribution of predicates in BTC10.

Semantic graph visualization is notoriously difficult, especially when large. However, in addition to the various statistics and figures we presented thus far, we devised a new method to visualize the instance graph of this dataset. We create an **extant ontology** as a graph $G = \langle C, p \rangle$ with nodes as classes C and edges as predicates p, but where each edge $e = C_s \xrightarrow{p} C_o \in G$ is both labeled by its predicate p, and also attributed with the count $c(e)$ of the number of occurrences of e, or with its relative frequency of occurrence.

Fig. 7 shows the extant ontology for the top 30 edge counts in BTC10. For example we have about 70M triples with the predicate `foaf:knows` connecting subject and object of class `foaf:Person`, the highest count. Note that many nodes in the dataset have more than one type, so that they contribute to more than one edge count and node label in the figure.

Effectively, Fig. 7 begins to show the statistical structure of the most significant part of BTC10. Extending beyond the top 30 edges quickly becomes visually difficult on paper, however we have computed the extant ontologies for up to the top 750 edges. We are making the .pdf and .dot graph visualization files for these bigger figures available online[9].

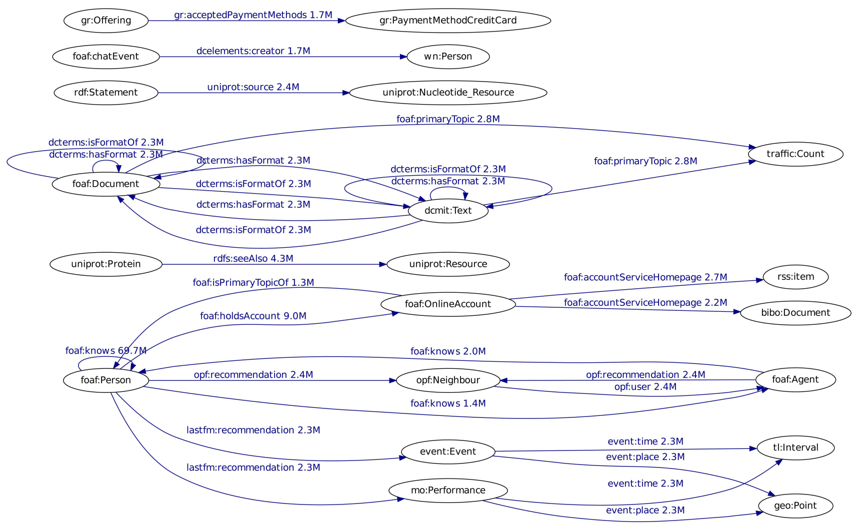

Fig. 7. The extant Ontology for the Top 30 Link-node-types in BTC10.

7 Path Type Analysis

We wish to identify the most prominent semantic structures and semantic constraints present in BTC10, not only simply to understand the BTC10, but also for future developments to exploit this semantic structure to provide targeted inferential support, and to optimize search and visualization methods to the specific ontology, connectivity, and distributional statistics of datasets and queries.

Semantic graphs are typed and directed. Where network analysis is frequently done in terms of paths connecting nodes, here we need to deal with directed paths which are themselves typed. Thus for a path of length n from an initial node of class C_s to a terminal node of class C_o, we cast its *path type* as the vector of the predicates $\langle p_1, p_2, \ldots, p_n \rangle$ which comprise the path.

We are interested in seeking the path types of the long paths which occur with high frequency. We hypothesize that these are the semantic structures which carry a large portion of the semantic information in the network in terms of interacting link types. Towards this end, we first consider the short paths which make them up, that is the chains of two and three link types which are connected linearly. These small, linear graph motifs are link-type n-grams for $n = 2, 3$. Note that the 1-grams are just the predicates themselves, and are shown in the extant ontology of Fig. 7.

For the bigram and trigram analysis we performed the most extensive semantic filtering as was also used for component analysis in Sec. 5. Table 6 shows the distribution of the top 20 bigrams of the 1.3M consecutive link type pairs,

[9] http://cass-mt.pnl.gov/hpcsw2011

comprising 53.0% of all 17.0B consecutive link pairs present; and Table 7 shows the distribution of the top 20 trigrams of the 72.7M consecutive predicate triples, comprising 7.54% of all 1.04T link triples.

p_1	p_2	Count (M)	%
dgtwc:isPartOf	dgtwc:partial_data	2,912.1	17.10%
geonames:inCountry	geospecies:hasUnknownExpectationOf	1,701.8	9.99%
geonames:inCountry	freebase:type.object.key	918.0	5.39%
geonames:inCountry	foaf:depiction	905.1	5.31%
geospecies:isUnknownAboutIn	geospecies:hasUnknownExpectationOf	516.2	3.03%
geonames:inCountry	geonames:wikipediaArticle	202.3	1.19%
geonames:inCountry	freebase:location.location.contains	178.0	1.05%
linkedmdb:link_target	geospecies:hasUnknownExpectationOf	158.0	0.93%
foaf:maker	geospecies:hasUnknownExpectationOf	144.9	0.85%
geospecies:isExpectedIn	geospecies:hasExpectationOf	142.6	0.84%
geospecies:isUnknownAboutIn	geospecies:hasLowExpectationOf	139.1	0.82%
geospecies:isUnexpectedIn	geospecies:hasUnknownExpectationOf	139.1	0.82%
geospecies:isExpectedIn	geospecies:hasUnknownExpectationOf	132.0	0.78%
geospecies:isUnknownAboutIn	geospecies:hasExpectationOf	132.0	0.78%
geonames:inCountry	geospecies:hasLowExpectationOf	125.5	0.74%
geospecies:isUnexpectedIn	geospecies:hasLowExpectationOf	124.1	0.73%
sioc:follows	sioc:follows	116.9	0.69%
geonames:inCountry	freebase:location.location.people_born_here	115.8	0.68%
geospecies:isUnknownAboutIn	freebase:type.object.key	115.3	0.68%
geospecies:isUnknownAboutIn	foaf:depiction	113.5	0.67%

Table 6. Top 20 link type bigrams (millions).

p_1	p_2	p_3	Count (B)	%
sioc:follows	sioc:follows	sioc:follows	10.85	1.05%
owl:disjointWith	owl:disjointWith	owl:disjointWith	6.86	0.66%
geonames:inCountry	geospecies:hasUnknownExpectationOf	foaf:isPrimaryTopicOf	6.86	0.66%
geonames:inCountry	geospecies:hasUnknownExpectationOf	geospecies:isUnknownAboutIn	5.80	0.56%
geonames:inCountry	freebase:location.country.admin_divisions	geospecies:hasUnknownExpectationOf	5.24	0.51%
geonames:inCountry	geospecies:hasUnknownExpectationOf	skos:closeMatch	4.63	0.45%
geonames:inCountry	purl:hasPart	purl:hasPart	4.27	0.41%
geonames:inCountry	freebase:location.location.contains	geospecies:hasUnknownExpectationOf	4.09	0.40%
geonames:inCountry	purl:hasPart	geospecies:hasUnknownExpectationOf	3.54	0.34%
geonames:inCountry	geonames:wikipediaArticle	geospecies:hasUnknownExpectationOf	2.83	0.27%
geonames:inCountry	geospecies:hasUnknownExpectationOf	geospecies:isExpectedIn	2.57	0.25%
geonames:inCountry	freebase:location.country.admin_divisions	geospecies:hasLowExpectationOf	2.44	0.24%
geonames:inCountry	factbook:landboundary	geospecies:hasUnknownExpectationOf	2.43	0.23%
geonames:inCountry	geospecies:hasUnknownExpectationOf	rdfs:seeAlso	2.36	0.23%
geonames:inCountry	geonames:parentFeature	geospecies:hasUnknownExpectationOf	2.32	0.22%
geonames:inCountry	purl:hasPart	geospecies:hasLowExpectationOf	2.31	0.22%
foaf:knows	foaf:knows	foaf:knows	2.19	0.21%
geos:isUnknownAboutIn	geospecies:hasUnknownExpectationOf	geospecies:isUnknownAboutIn	2.15	0.21%
geos:hasUnknown ExpectationOf	geospecies:isUnknownAboutIn	geospecies:hasUnknownExpectationOf	2.15	0.21%
geonames:inCountry	freebase:location.location.contains	geospecies:hasLowExpectationOf	2.03	0.20%

Table 7. Top 20 link type trigrams (billions).

Note the prominence of low-frequency predicates in both the bigrams and trigrams. For example, consider the most frequent bigram $\langle$`dgtwc:isPartOf, dgtwc:partial_data`$\rangle$, with a frequency of 17.1%. The constituent predicates have frequencies of 0.0038% and 0.027% respectively, far below the top 16 shown in Fig. 5. If these were independent, the expected joint frequency would be minuscule. This pattern of a vast inflation of expected probability is a general phenomenon, indicating the powerful role that these small sequence motifs play in the semantics of BTC10.

8 Conclusions

In this work we focused explicitly on analyzing the BTC10 data set with its 1.4 billion-edge graph. We employed the Cray XMT in most of these analyses and in the process have made important discoveries that not only explain and help visualize the various properties of this data, but also point out to future directions where exploiting these properties is essential to designing even better performing semantic databases and analyses tools. The assumed graph-nature of the data model did suggest that HPC architectures designed for graph-like problems may be a good match for this domain and indeed we have shown the

XMT to be an excellent platform for such tasks. However we also demonstrated that patterns are plentiful in the data. Accordingly, heavy-tail predicate and type distributions, prevalence of terminal edges, n-grams, and extant ontological substructures should all be further studied in order that they may be used in designing a hybrid semantic HPC solution. We are presently working in this direction.

Acknowledgments

Thanks first to some excellent comments from reviewers. This work was funded by the Center for Adaptive Supercomputing Software – Multithreaded Architectures (CASS-MT) at the Dept. of Energy's Pacific Northwest National Laboratory. Pacific Northwest National Laboratory is operated by Battelle Memorial Institute under Contract DE-ACO6-76RL01830.

The authors thank Liam McGrath (PNNL) for assistance on *n*-gram analysis.

References

1. Berry, J; Hendrickson, B; Kahan, S; and Konecny, P: (2006) "Graph Software Development and Performance on the MTA-2 and Eldorado", in: *48th Cray Users Group Meeting*
2. Chavarria-Miranda, D; Marquez, A; Nieplocha, J; Maschhoff, K; C Scherrer: (2008) "Early Experience with Out-of-Core Applications on the Cray XMT", in: *Proc. 22nd IEEE Int. Parallel and Distributed Processing Symp.*, pp. 1-8, `10.1109/IPDPS.2008.4536360`
3. Feo, John; Harper, David; Kahan, Simon; and Konecny, Petr: (2005) "ELDORADO", in: *CF '05: Proceedings of the 2nd conference on Computing frontiers*, pp. 28-34, ACM, Ischia, Italy, `http://doi.acm.org/10.1145/1062261.1062268`
4. Goodman, E; Haglin, David J; Scherrer, Chad; Chavarria, D; Jace Mogill, John Feo: (2010) "Hashing Strategies for the Cray XMT", in: *Proc. 24th IEEE Int. Parallel and Distributed Processing Symp.*
5. Christophe Guéret, Paul T. Groth, Frank van Harmelen, Stefan Schlobach: (2010) "Finding the Achilles Heel of the Web of Data: Using Network Analysis for Link-Recommendation", Int. Semantic Web Conf. (1) 289-304
6. Joslyn, Cliff; Adolf, Bob; al-Saffar, Sinan; Feo, John; Eric Goodman, David Haglin, Gregy Mackey, David Mizell: (2010) "High Performance Semantic Factoring of Giga-Scale Semantic Graph Databases", in: *Semantic Web Challenge 2010, Int. Semantic Web Conf.*, runner-up winner, `http://www.cs.vu.nl/~pmika/swc/submissions/swc2010_submission_15.pdf`
7. Sheila Kinsella, Uldis Bojars, Andreas Harth, John Breslin, S Decker: (2008) "An Interactive Map of Semantic Web Ontology Usage", 12th Int. Conf. Conference Information Visualisation (IV08), London, UK, IEEE Computer Society
8. Weiyi Ge, Jianfeng Chen, Wei Hu, Yuzhong Qu (2010): "Object Link Structure in the Semantic Web", ESWC (2) 257-271