High Performance Semantic Factoring of Giga-Scale 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 extant large triple stores to SPARQL endpoints over SQL-structured relational databases, it will become increasingly important to be able to bring high performance computational resources to bear on their analysis, interpretation, and visualization, especially with respect to their innate semantic structure. Our research group built a novel high performance hybrid system comprising computational capability for semantic graph database processing utilizing the large multi-threaded architecture of the Cray XMT platform, conventional clusters, and large data stores. In this paper we describe that architecture, and present the results of our deploying that for the analysis of the Billion Triple dataset with respect to its semantic factors, including basic properties, connected components, namespace interaction, and typed paths.

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

1 Introduction

As semantic graph database (SGD) 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 bring high performance computational resources to bear on their analysis, interpretation, and visualization, especially with respect to their innate semantic structure.

Prior Billion Triple Challenge (BTC) submissions[4] have explored many interesting and provocative topics, ranging from query rewriting to mobile applications, ontology hijacking to navigation. But the ability to understand the semantic structure of a vast SGD 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 SGD problems different from other large network science problems, perhaps most prominently their formal nature as structures which are not only large, but have high data complexity in that they are typed

* Corresponding author: Cliff Joslyn, Pacific Northwest National Laboratory, cjoslyn@pnl.gov, 206-528-3042.

[4] http://challenge.semanticweb.org

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 structure of the links, seen as predicates. But standard methods in network science (e.g. connected components, minimum path, centrality, etc.) have been developed for networks of high 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.

To address these issues, our research group has built a novel high performance hybrid system comprising computational capability for semantic graph database processing utilizing high capacity standard servers together with the large multi-threaded architecture of the Cray XMT platform. We have brought these capabilities to bear on the BTC 2010 dataset (BTC10).

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 analyze the connected component structure of BTC10, and finally factor BTC10 according to network motifs which are short, typed paths, specifically link type bigrams and trigrams. In this way the inherent semantic structure of BTC10 is revealed to users.

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.

Our Cray XMT has 128 *Threadstorm* processors, each of which supports 128 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 a platform with fine-grain synchronization support in the hardware and runtime systems. It is common 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 BTC 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 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 these two libraries 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 BTC10 Base Statistics

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.. 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.

Abbreviation	Prefix
bestbuy:	http://products.semweb.bestbuy.com/company.rdf
cyc:	http://sw.cyc.com/CycAnnotations_v1
dbin:	http://www.dbin.org/SWDesktopIntegration
dg:	http://data-gov.tw.rpi.edu/vocab/p/
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/
geo:	http://rdf.geospecies.org/ont/geospecies
mindswap:	http://owl.mindswap.org/2003/ont/owlweb.rdf
ontoware:	http://smw.ontoware.org/2005/smw
owl:	http://www.w3.org/2002/07/owl
po:	http://www.proteinontology.info/po.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
smwiki:	http://semantic-mediawiki.org/swivt/1.0
sniff:	http://www.w3.org/2000/10/swap/util/sniffSchema
swoogle:	http://daml.umbc.edu/ontologies/webofbelief/1.4/swoogle.owl
swvocab:	http://www.w3.org/2003/06/sw-vocab-status/ns

Table 1. Prefix abbreviations used.

The namespace abbreviations used below are shown in Table 1. Below we only show selected high frequency objects below. See `http://cass-mt.pnl.gov/btc2010/stats` for complete tables.

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

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

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.

Note the far smaller number of subjects compared to objects, indicating a much larger in-degree of objects to out-degree of subjects. The most prevalent subjects are "containers", each (e.g. Bestbuy) pointing to a single category of a large number of objects (e.g. Offers). The most prevalent objects are types, virtually all (e.g. `foaf:person`) the destination of `rdf:type` predicates.

Table 3 shows the top 20 of the 95.2K link types present, comprising 37% of all 1.4B link instances present. Fig. 1 shows the top 20 edge counts per node types in BTC10 (literals omitted). For example we have about 70M triples with the predicate `foaf:known` connecting subject and object of type `foaf:person`, the highest count. Node labels are node types, indicated as objects of the `rdf:type` predicate. Many nodes in the dataset have more than one type in which case they contribute to more than one edge count and node label in the figure.

Effectively, Fig. 1 begins to show the statistical structure of the "extant ontology" of BTC10. Extending beyond the top 20 edges quickly becomes visually difficult, see `http://cass-mt.pnl.gov/btc2010/top218links.pdf` for a figure showing the top 218 edges, comprising all the edges with counts $\leq 100K$.

s			*o*		
s	K	%	*o*	M	%
bestbuy:BusinessEntity_BestBuy	413.6	0.043%	xmlns.com/foaf/0.1/Person	68.39	15.921%
data-gov.tw.rpi.edu/raw/91/index.rdf#me	148.0	0.015%	xmlns.com/foaf/0.1/OnlineAccount	10.15	2.363%
data-gov.tw.rpi.edu/raw/90/index.rdf#me	54.0	0.006%	rdf:Statement	6.06	1.412%
sws.geonames.org/2635167/	16.1	0.002%	xmlns.com/foaf/0.1/Document	4.67	1.086%
sws.geonames.org/4862182/	15.9	0.002%	purl.org/rss/1.0/item	4.65	1.082%
sws.geonames.org/5279468/	15.8	0.002%	dgtwc:DataEntry	4.00	0.932%
sws.geonames.org/6255149/	15.8	0.002%	purl.org/dc/dcmitype/Text	3.22	0.750%
sws.geonames.org/5037779/	15.8	0.002%	www.w3.org/2003/01/geo/wgs84_pos#Point	2.45	0.570%
sws.geonames.org/5001836/	15.8	0.002%	rdf.opiumfield.com/lastfm/spec#Neighbour	2.36	0.551%
www.livejournal.com/interests.bml?int=???????	12.0	0.001%	purl.org/ontology/mo/Performance	2.27	0.529%
www.livejournal.com/interests.bml?int=??????	11.2	0.001%	purl.org/NET/c4dm/timeline.owl#Interval	2.26	0.525%
www.livejournal.com/interests.bml?int=????????	10.9	0.001%	purl.org/NET/c4dm/event.owl#Event	2.26	0.525%
www.w3.org/TR/2001/REC-smil20-20010807/	9.3	0.001%	www.geonames.org/ontology#Feature	2.21	0.513%
www.livejournal.com/interests.bml?int=?????	9.0	0.001%	www.last.fm/	1.67	0.389%
www.livejournal.com/interests.bml?int=?????????	8.8	0.001%	xmlns.com/wordnet/1.6/Person	1.66	0.386%
dowhatimean.net/2010/01/prefixcc-mkii#comment-	8.8	0.001%	xmlns.com/foaf/0.1/chatEvent	1.66	0.386%
www.nettrust-site.net/fdic	8.4	0.001%	purl.uniprot.org/core/classifiedWith	1.56	0.363%
www.fao.org/aims/aos/languagecode.owl#I	7.8	0.001%	www.w3.org/2000/01/rdf-schema#seeAlso	1.54	0.358%
wow.sfsu.edu/ontology/rich/FoodWebs.owl#LittleRockLake	7.7	0.001%	purl.uniprot.org/core/Domain_Assignment_Statement	1.50	0.349%
www.fao.org/aims/aos/languagecode.owl#L	7.2	0.001%	purl.org/goodrelations/v1#ProductOrServiceModel	1.46	0.341%

Table 2. (Left) Top 20 subjects, count (thousands), and %; (Right) top 20 objects, count (millions), and %.

p	Count (M)	%
rdf:type	152.8	10.7%
rdfs:seeAlso	86.7	6.1%
foaf:Person	71.6	5.0%
foaf:OnlineAccount	70.5	4.9%
rdf:Statement	15.3	1.1%
foaf:Document	11.5	0.8%
rss:item	10.8	0.8%
dgtwc:DataEntry	10.8	0.8%
dcmitype:Text	10.4	0.7%
geo:Point	10.4	0.7%
opmspec:Neighbour	10.4	0.7%
mo:Performance	8.4	0.6%
timeline:Interval	8.3	0.6%
event:Event	8.2	0.6%
geonames:Feature	8.0	0.6%
wordnet:Person	7.5	0.5%
foaf:chatEvent	6.1	0.4%
core:Domain_Assignment_Statement	6.1	0.4%
gr:ProductOrServiceModel	6.1	0.4%
swrc:Person	5.6	0.4%

Table 3. Top 20 predicates (millions).

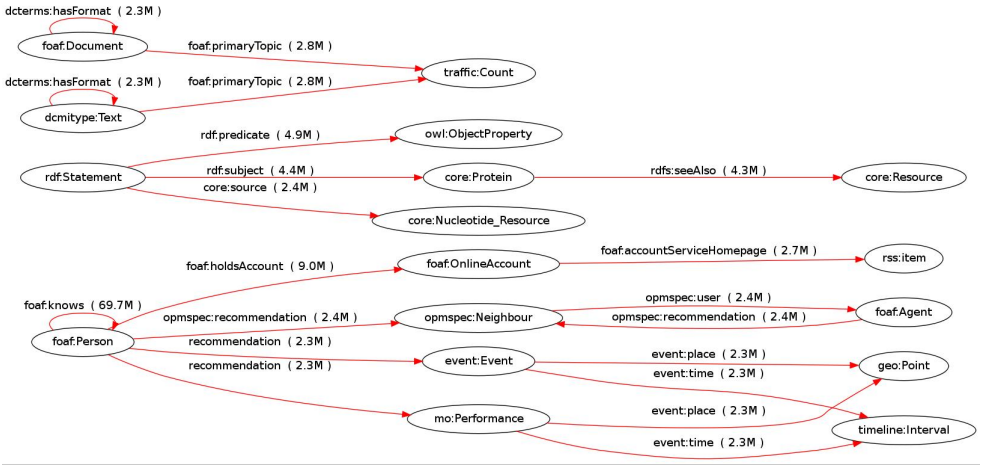

Fig. 1. Top 20 edge types, and the node types they connect.

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 4. These are dominated by `rdf:type`, `rdfs:seeAlso`, and `rdfs:label`, with `rdf:type` alone comprising 10.7%.

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

Reification in particular is prominent in BTC10. There are 4.4M edges of predicate `rdf:subject` between `rdf:statement` and `core:protein`, the main source. The absence of the `rdf:object` predicate in Fig. 1 indicates that objects are spread among many types, and/or are to literals. Overall, that `rdf:subject`, `rdf:object`, and `rdf:predicate` each have 6.1M instances precisely, comprising a clear portion of BTC10 that is reified.

4 Namespace Interaction

To understand the relationships between the sources which generated the dataset, we explore n summary metric for linkages among namespaces. Projects like the Linking Open Data initiative and the Comprehensive Knowledge Archive Network rely on manual attribution and curation of provenance. In BTC10 we must use an approximation method for attributing triples to sources.

We call triples "linked" when two or more of the subject, predicate, or object map to different fully-qualified domain names (FQDNs). Table 5 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 BTC 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.

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 5. Sources of cross-linking by entity position

5 Connected Components

In the previous section we discussed how prefixes can be used to understand the interconnectedness of the BTC graph. In this section we discuss a more graph theoretic approach: connected components. This approach is used to find the set of maximal 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 the large component. We then find the remaining components by using a "bully-strategy" [1].

To pose the BTC 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 BTC, we find that there are 208.3K components, with a giant component of 278.4K 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:definedBy, 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.

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

6 Typed Path Structures

As noted above, SGD structures are formally represented as typed, directed networks. Where network analysis is frequently done in terms of paths connecting

nodes, here we need to deal with directed paths which are themselves typed, casting a *path type* as the vector of the link types which comprise the path.

We are interested in seeking the path types of the long paths which occur with high frequency. We hypothesize these as 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 $n = 2, 3$.

We hypothesize that such information will be important not only for users to understand the semantic structure of BTC10, but 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.

Our bigram and trigram analysis involved filtering the ontological metadata link types, specifically the `rdf:`, `rdfs:`, and `owl:` namespaces. Table 6 shows the distribution of the top 20 bigrams of the 824.6K consecutive link type pairs, comprising 86.3% of all 8.1B consecutive link pairs present; and Table 7 shows the distribution of the top 20 trigrams of the 5.6M consecutive link type triples, comprising 29.6% of all 102.8B link triples.

p_1	p_2	Count (M)	%
dgtwc:isPartOf	dgtwc:partial_data	2912.13	35.8%
foaf:interest	purl:title	2036.23	25.0%
gs:isUnknownAboutIn	gs:hasUnknownExpectationOf	516.19	6.3%
gs:isExpectedIn	gs:hasExpectationOf	142.62	1.8%
gs:isUnknownAboutIn	gs:hasLowExpectationOf	139.13	1.7%
gs:isUnexpectedIn	gs:hasUnknownExpectationOf	139.13	1.7%
gs:isUnknownAboutIn	gs:hasExpectationOf	132.04	1.6%
gs:isExpectedIn	gs:hasUnknownExpectationOf	132.04	1.6%
gs:isUnexpectedIn	gs:hasLowExpectationOf	124.14	1.5%
sioc:follows	sioc:follows	116.87	1.4%
gs:isUnexpectedIn	gs:hasExpectationOf	84.12	1.0%
gs:isExpectedIn	gs:hasLowExpectationOf	84.12	1.0%
foaf:knows	foaf:knows	81.69	1.0%
foaf:primaryTopic	foaf:maker	77.68	1.0%
foaf:knows	foaf:nick	68.93	0.8%
fao:hasScope	fao:isScopeOf	60.16	0.7%
fao:hasType	fao:isTypeOf	52.24	0.6%
foaf:accountServiceHomepage	purl:title	41.69	0.5%
foaf:knows	foaf:holdsAccount	39.26	0.5%
foaf:based_near	gs:hasUnknownExpectationOf	36.16	0.4%

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

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 35.8%. The constituent predicates have frequencies of 0.0038% and 0.027% respectively, far below the top 20 shown in Table 3. 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.

7 Appendix

Concerning the minimal requirements of the Billion Triple Challenge, in this work we focused explicitly on analyzing the BTC10 data set, and in providing a collection of services to help users analyze their inherent semantic structure.

p_1	p_2	p_3	Count (B)	%
sioc:follows	sioc:follows	sioc:follows	10.85	10.6%
foaf:knows	foaf:knows	foaf:knows	2.19	2.1%
gs:hasUnknownExpectationOf	gs:isUnknownAboutIn	gs:hasUnknownExpectationOf	2.15	2.1%
gs:isUnknownAboutIn	gs:hasUnknownExpectationOf	gs:isUnknownAboutIn	2.15	2.1%
gs:isUnknownAboutIn	gs:hasUnknownExpectationOf	foaf:isPrimaryTopicOf	1.98	1.9%
gs:isUnknownAboutIn	gs:hasUnknownExpectationOf	skos:closeMatch	1.37	1.3%
rdf:predicate	http://sw.nokia.com/VOC-1/partOf	http://sw.nokia.com/VOC-1/term	1.32	1.3%
foaf:primaryTopic	gs:isUnknownAboutIn	gs:hasUnknownExpectationOf	1.08	1.1%
skos:closeMatch	gs:isUnknownAboutIn	gs:hasUnknownExpectationOf	0.78	0.8%
gs:hasUnknownExpectationOf	gs:isUnknownAboutIn	gs:hasExpectationOf	0.64	0.6%
gs:isExpectedIn	gs:hasUnknownExpectationOf	gs:isUnknownAboutIn	0.64	0.6%
gs:hasUnknownExpectationOf	gs:isUnknownAboutIn	gs:hasLowExpectationOf	0.64	0.6%
gs:isUnexpectedIn	gs:hasUnknownExpectationOf	gs:isUnknownAboutIn	0.64	0.6%
gs:isExpectedIn	gs:hasExpectationOf	foaf:isPrimaryTopicOf	0.62	0.6%
gs:isUnknownAboutIn	gs:hasLowExpectationOf	foaf:isPrimaryTopicOf	0.62	0.6%
gs:isUnknownAboutIn	gs:hasExpectationOf	foaf:isPrimaryTopicOf	0.61	0.6%
gs:isUnknownAboutIn	gs:hasLowExpectationOf	gs:isUnexpectedIn	0.56	0.5%
gs:hasLowExpectationOf	gs:isUnexpectedIn	gs:hasUnknownExpectationOf	0.56	0.5%
gs:isUnexpectedIn	gs:hasLowExpectationOf	foaf:isPrimaryTopicOf	0.53	0.5%
gs:isUnknownAboutIn	gs:hasUnknownExpectationOf	gs:isExpectedIn	0.52	0.5%

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

We used no additional data in performing this work. We have a real-time[7] end-user application available to show the results of our namespace, link-type, and link-type n-gram filtering in the context of connected component analysis.

Concerning the additional desirable features, our work goes substantially beyond storage or retrieval of the triples to analyze the semantic structure of BTC10. Our use of a high-performance platform aims explicitly at scalable results, as demonstrated in our performance results detailed in the text above.

Acknowledgments

This work was funded by the Center for Adaptive Supercomputing Software – Multithreaded Architectures (CASS-MT) at the Dept. of Energys 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.*

[7] http://cass-mt.pnl.gov/btc2010