Report on April, 2011, Workshop on Semantic Graph Database Search Patterns*

Cliff Joslyn, Bob Adolf, Sinan al-Saffar, John Feo, and David Haglin

Pacific Northwest National Laboratory

Abstract. We report on a workshop on Semantic Graph Database Search Patterns held in Seattle in April, 2011.

1 The Need for Benchmark Data and Queries for Semantic Graph Datbases

A new database paradigm based on graph data, and specifically semantic graph databases (SGD), is rising in ascendancy. While standards are still emerging, the current technical driver in the community is the use of OWL ontologies to provide typing information over RDF(S) triplebases serving up semantic graphs as data instances, and facilitated by the SPARQL query language. There is an emerging consensus that systems must scale up to the order of $10^{10} - 10^{12}$ edges in the triplebase, and/or on the order of $10^6 - 10^7$ classes in their ontologies. Thus, there is a need to bring high-performance computing to the SGD world.

As we migrate toward this reality, it is important that we have appropriate resources to drive development of SGD implementations. With so many years of experience with the simpler Relational Database models and queries, a paradigm shift is required to begin to understand the kinds of questions that can be asked (and answered) with high performance SGD systems, and the data or type of data against which the queries will be posed. There is the need to develop benchmark standards for datasets and queries which properly control certain aspects of SGD hardware and software realizations while allowing others to vary.

Our research group at the Pacific Northwest National Laboratory in Washington State in the USA, together with colleagues at the Sandia National Laboratories, Cray Inc., and elsewhere, have been pursing this interaction between SGD and HPC, and seeking such standards. But as we entered this field many months ago, we were struck that:

- Discussions with researchers about their graph data and queries failed to clearly articulate or demonstrate the demand for novel technologies. Instead, we discovered primarily simple queries over dense data, which were ammenable to traditional database approaches.
- Discussions with vendors about documenting their performance on very large semantic graph data were frustrating, due to a lack of common (public) benchmarking databases and queries integrating real data and use cases.

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

– And discussions with sponsors about our interest in hybrid data platforms begged the question about judging the need: what, we asked, was a prime graph data problem? When are hybrid solutions demanded?

Realizations of high-performance SGDs can vary widely by hardware platform, size and properties of the ontology (if any), size and properties of the triplebase, software engineering of the SGD engine, and quite significantly, the nature of the queries anticipated to be supported by the platform. Thus benchmark standards for SGD implies not just standard data sets, but standard *queries* with which they are paired. Better yet would be a set of *abstract* standard queries which can be brought to multiple SGD configurations, and then both instantiated and parameterized against them in a consistent manner.

2 Workshop on Semantic Graph Database Search Patterns

Thus we have taken as a goal to develop a set of abstract graph query patterns which when instantiated against a set of large triplebases produce compelling standard queries. Towards that end, we gathered researchers in Seattle, Washington, USA, on April 25-26, 2011, for a workshop on Semantic Graph Database Search Patterns. Through the workshop we hoped to facilitate the integration of SGD with HPC, and understand the *real* need for large graph databases. Specific goals included:

1. Identify real-world semantic graph datasets spanning multiple domains (e.g. bionetworks, social networks, e-science, and government data).
2. Define a set of challenging queries representative of graph search patterns common in use cases from those different domains.
3. Identify a set of domain-independent, mathematically abstract search patterns for which the domain-specific queries are instantiations.

27 researchers attended, representing academic (Indiana U, U Washington, UCSD, UT Austin, Johns Hopkins U, Rennselaer Polytechnic), government (Pacific Northwest National Lab, Sandia National Lab), and industry (Oracle, Mayo Clinic, Microsoft, Noblis, Hayden, Cycorp) research labs. Multiple kinds of researchers attended, including:

– Those who own, represent, or are otherwise deeply connected to graph data in particular domains (e.g. bioinformatics, social science)
– Theoreticians working with the underlying formalisms and methodologies for graph databases
– Engineers and developers whose mission is to build robust SGD platforms

Attendees were organized into four groups, and charged to pursue items (1)–(3) above. In addition, we identified a number of attendent technical questions, which, while not the primary focus of the workshop, we want to be attentive to, including:

- What current "benchmarks" did groups use or know of, e.g. the Billion Triple Challenge[1], SP2Bench [2], data.gov[2], LUBM [1][3], Bio2RDF[4], and Uniprot[5]?
- What is the future role for hybrid graph/relational DBs?
- How do we prepare for dynamic optimization of graph queries?
- How enabling or limiting is the OWL/RDF(S)/SPARQL paradigm?
- How does the expected evolution of graph query languages impact the field (e.g. SPARQL 1.1)?
- How do we accommodate the need for scalar and vector quantities and operations in graph databases, e.g. to represent spatial proximity, temporal sequence, or other quantitative attributes?
- What should be the role of model-generated test or benchmark triplebases (e.g. LUBM) versus those of other sources (e.g. webcrawls, linked open data[6])?
- What is the significance and role of ontological typing information, and its presence or absence in benchmark triplebases?
- How significant is the use of such ontological information and the necessity for inference (materialized or unmaterialized) in queries?
- What is the potential role for approximate inference and query? How are temporal dynamics accommodated?

In the sequel we present a combined report on some of the specific findings the groups came up with, in terms of:

- Identifying and characterizing real-world data sets, including:
  - Their domain and coverage;
  - Use cases;
  - Their formal nature as graphs, relations, arrays, transactions, etc.;
  - Their typing complexity in terms of schema columns and tables or graph ontologies;
  - Their size in terms of instances and physical sizes; and
  - Their temporal dynamics and update rates.
- Identifying classes of queries, including:
  - Identifying natural queries beyond current capabilities;
  - Identifying the extent of specifically graph database challenges;
  - Qualifying their size and complexity;
  - Identifying the presence of non-graphical elements such as quantitative attributes (e.g. space and time); and
  - Identifying the presence of aggregation, indirection, and inference.
- Attempt a first cut at a mathematical generalization.

[1] http://challenge.semanticweb.org
[2] http://data-gov.tw.rpi.edu/wiki
[3] http://www.cse.lehigh.edu/~yug2/ssws10/home.htm
[4] http://bio2rdf.org/
[5] http://dev.isb-sib.ch/projects/uniprot-rdf
[6] http://linkeddata.org

3 Initial Data Sets Considered

Some data sets were considered, but not developed further. These included:

Joint publications and biomedical data: Considering a loosely bipartate structure of biomedical publications pairs against the gene products they reference in a many-many relation, for example Medline[7] against Uniprot or the Gene Ontology[8]. A sample query could be to find all authors at the Mayo clinic who have published on colon cancer.

Linked Life Data: [9] A systems biology repository for causal pathway reasoning, combining OpenCyc plus lexicon for decoding paper titles.

Foundational Model of Anatomy: [10] 1.7 M RDF triples, much of it ontological (subClass, subProperty, etc.) data with a tree-like structure. They do not currently do any inferencing or constraint-checking on the relations.

4 Instantiated Queries and Data Sets

A number of other data sets were developed, together with queries.

4.1 NSF Proposal Conflicts of Interest

Consider the set of proposals submitted to the US National Science Foundation, and whether or not there are conflicts of interest between the submitters and a collection of authors of papers who are being recruited to serve on a review panel. Then given a proposal, a topic, and a set of documents, a joint query and sub-query to determine whether any paper authors have conflicts with the investigators is as follows:

- Let a panel candidate be any author of a paper about the topic such that
  - the author resides in North America and
  - has no conflicts with any of the investigators of the proposal.
- A conflict occurs when either:
  - an author has co-authored a paper with an investigator published within the last 48 months; or
  - is an investigator; or
  - is married to an investigator.

The graphical form of this query is shown in Fig. 1, for an *ad hoc* graphical notation. Some features of this query include:

- The presence of negation
- The need for inference, to explore a geographic partonomy (fragment shown), to handle the query composition between the `Non-Conflicting Reviewers` query and the `Conflict` predicate, and potentially a topic hierarchy.
- Recursion, in the potential for sub-awards within main NSF awards

[7] http://www.nlm.nih.gov/databases/databases_medline.html
[8] http://geneontology.org
[9] http://linkedlifedata.com
[10] http://sig.biostr.washington.edu/projects/fm/AboutFM.html

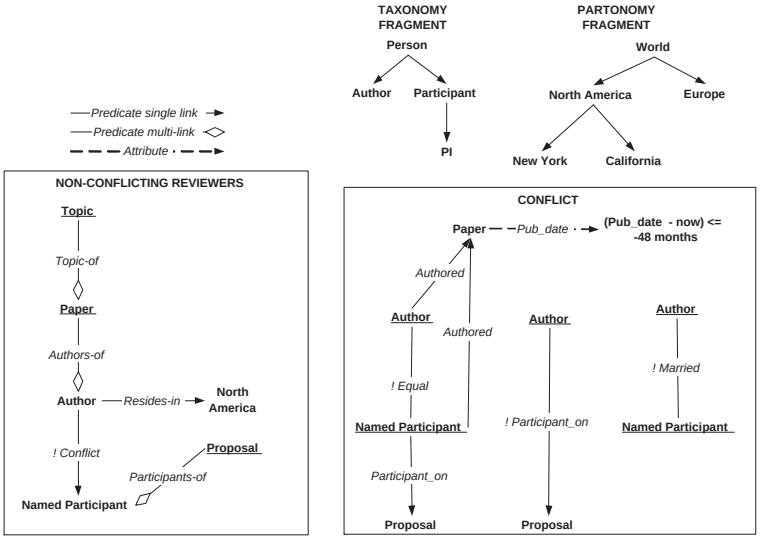

Fig. 1. Graphical form of the NSF query. Input objects are underlined, negated predicates use `!`. The left box shows the `Non-Conflicting Reviewers` query, which includes the `Conflict` predicate, itself expanded in the right box. Above is a relevant fragment of an ontology, including a taxonomy and a partonomy.

- Aggregation, as papers with more than 12 authors are discarded
- Disjunction, as shown in the `Conflict` predicate, and as indicated by the disconnected graph components
- The presence of directed and undirected links
- The presence of quantitative temporal attributes and arithmetic expressions
- The presence of units of measure (months)

4.2 Mayo Clinic Medical Records

Mayo is developing a dataset containing all of their hundreds of years of clinical records, including 7 M patients, of which 4 M are in electronic form. The electronic medical records (EMRs), composing 6 PB of data across 1200 sources, and a separate 6 PB of image data. There are 200–300 major sources, many of the others contain only a small amount of information. Some of the data is structured and/or in relational databases, and some is unstructured and/or free-form text, including 1.4 TB of clinical notes. The proposed aggregate dataset would aim to cover all of Mayo's important categories, including admissions, administrative data, diagnostics, lab results, and pathology reports. But there is substantial redundancy, and it is hypothesized that the entire dataset could be shrunk to about 1 PB of semantic relationships.

This data is patient-oriented, so that the links between patient and diagnostic or patient and hospital visit might be rich and clear. But other relationships

might be more difficult: while a physician's diganosis of a patient's condition as a particular ICD9[11] code (which could be unified across patients), a description from a pathology report about the symptoms of an ambiguous condition could be very difficult to create relationships from. Potential queries include:

Physician Experience: *Does a physician's years of experience help or harm a patient's outcome?* This is somewhat broad, but given EMRs of all patients in Mayo, identify similar conditions handled by different physicians, and then correlate the patient's outcome with the years of practice that the responsible physician had. There are multiple definitions of "similar" and "outcome", but the correlation and years of practice properties are straightforward. ICD codes could be used to identify similar conditions, but one could easily say that outcome is significantly affected by correct diagnosis, which limits the usefulness of specific diagnostic codes. Likewise, outcome might be longevity or time-to-cure, but both of these can be biased quite easily (longevity by secondary diseases, and time-to-cure by the patient's visitation record). One could rightly claim that this is a special case of evidence-based medicine, but because it was brought up independently and had a clearly-defined use case, it seemed worth breaking out separately.

Physician Training: *How significant are trends corresponding to a physician's particular school of training to a patient's outcome or treatment?* Like the last question, but instead of comparing years of experience, compare the location where the physician had their formal training. Different medical schools teach different techniques and emphasize different lessons, and the goal is to identify which are *actually* more effective in treating disease in the real world. Again, this is a special case of evidence-based medicine, but there are cases for which the answer is actually known (one attendee observed that certain schools teach procedures while others which do not, this could serve as a reference answer for the problem).

An example of an abstraction concerns the connection between two people. The motivation for this type of query came from the scenario of two people with similar symptoms: what is the strongest connection between them other than the symptoms? We invented some SPARQL syntax to represent this.

```
SELECT DISTINCT * WHERE {
  ?patient1 rdf:type :Patient .
  ?patient2 rdf:type :Patient .
  ?patient1 (?)* ?factor .
  ?patient2 (?)* ?factor .
  ?factor rdf:type (:Address | :Person | :Location | :Drug) .
}
```

Also in the Mayo clinical data, a query for medication side effects was considered, together with its abstraction. This query is a matter of finding the

[11] http://www.cdc.gov/nchs/icd/icd9cm.htm

fixed path through the data from a drug through prescription through person to symptom. Aggregating these by, and presenting them in order by highest frequency finishes out this query. There may be some temporal constraints needed such as symptom is diagnosed some minimum time after the drug is prescribed. Abstactly, this becomes searching for two internally disjoint paths between endpoints. Another perspective is that this is a cycle with two specific vertex types. In the following abstraction, ?X represents the symptom and ?D represents the medication.

```
SELECT DISTINCT * WHERE {
  ?X rdf:type "X_type" .
  ?Y1 abstract:relatesTo_1 ?X .
  ?Y1 rdf:type "Y_type" .
  ?Y2 abstract:relatesTo_1 ?X .
  ?Y2 rdf:type "Y_type" .
  ?Z1 abstract:relatesTo_2 ?Y1 .
  ?Z1 rdf:type "Z_type" .
  ?Z2 abstract:relatesTo_2 ?Y2 .
  ?Z2 rdf:type "Z_type" .
  ?D abstract:relatesTo_3 ?Z1 .
  ?D abstract:relatesTo_3 ?Z2 .
  ?D rdf:type "D_type"
}
```

4.3 Mayo Clinic Genetic Data

One research area at Mayo is correlation of genes with adverse conditions. First, one thousand patients' genomes are sequenced with a high-throughput gene sequencing machine. The idea is to use known reference models of biological interactions to compare against the genomes from the diseased patients. The high-throughput sequencers produce data on the order of 50–5000 nodes (genes), mostly because these are down-selected to just the mutations or interesting areas. The reference models are generally much larger: 25K–500K interactions (edges). The "semantic" aspect of the data is the typed nature of the reference models. A gene interaction network alone may be undirected and untyped, but combine it with a gene expression network (directed, bipartite, and with hyper-edges) and a regulatory network (directed, typed), and you start to have a very rich dataset of many millions of edges. Effectively, this is a multi-modal dataset, and unification *is* a problem: some nodes (genes, proteins, etc.) are named similarly between reference networks, but not universally. Moreover, the questions in this area generally need cross-correlation; you are not working with just one patient's genome, but maybe 1000 patients' data at the same time, which grows the problem. Currently, they are using the Gene Ontology[12], but in a very coarse manner. One example is grouping genes by function ('function' is a defined relationship

[12] http://www.geneontology.org/

in the GO). Finally, gene sequencing today is done on a "typical" cell. In reality, there are thousands of different types of cells, each with different genetic codes. In the ideal case, one would want to sequence all of them, expanding the patient's dataset from 50–5000 to something much larger.

Potential queries in personalized medicine and genetic network correlation could then include:

- **Find caucasian males, aged 20–30, with a specific disease:** All the constraints are specified to exact values (caucasian, male, 20–30, ICD code for disease). It should be a simple lookup problem, albeit with a numeric interval.
- **Find people that look like caucasian males, age 20–30, with a type of disease:** This query should return people who fit *most* of the criteria, if not all (i.e.- an caucasian male, aged 35 with the disease might be returned). Also, the constraints are no longer exact: the disease encompasses an entire class of diagnostic codes, and will require ontological inference.
- **Find people that look like caucasian males, age 20–30, with a type of disease, then identify other correlated characteristics and find people with those characteristics.** Like the previous query, but this time adding in correlation of additional properties across the return set and then expanding the set again to match those additional properties. An example might be identifying smoking as a correlated characteristic in the answer set (and not as strongly correlated in the overall data) and then returning a cohort including a 35-year-old hispanic female with the disease who smokes.
- **Find all correlated conditions related to a certain disease.** The logical conclusion of the previous queries. Now all characteristics except the disease type are left open. The system must identify all of the correlated properties across people with the disease and automatically build the result set (cohort). In this type of query, there is almost no emphasis on specifying data constraints but a significant amount of effort to constrain the algorithm's behavior (expected cohort size, correlation threshold, etc.). Mayo would like to run this type of query for every type of disease (though one would expect to have to have different parameters to generate good result sets for different diseases).
- **Which genes are bottlenecks for signaling known to be associated with a condition?** Given two known sites A and B (although we originally described the known set to be "bad", it needn't strictly be; in this case, a disease might be caused by the *absence* of a genetic pathway), find all the distinct pathways which link the two sites, passing through genes in the patient's list. In one case, this might just be a constrained path search between A and B, crossing certain links in a certain order with a certain length. It could be more complex, however. One possibility is that the patient's list of genetic differences are actually interpreted as modifications to the reference graph (i.e.- edges removed or added). The goal might then be to identify paths related to A and B which are affected by the absence or inclusion of some gene.

- **Given a multi-modal genetic reference network N and a set of bad genes G_{Bad}, find the value of a centrality metric M for a gene G_A with respect to G_{Bad}.** This (and the following set of queries) is a more exact specification of the previous problem. Here we are given an algorithm or expression for evaluating the effect of a gene on a path, and we just need to execute it. This algorithm would almost certainly exploit the semantics of the network, calculating only the paths that have some pattern or satisfy some constraint, but it is still a known routine with some set of parameters. An example M could be any of the common centrality algorithms in graph theory, modified to operate on the subgraph which is produced by applying a semantic filter.
- **Given N, G_{Bad}, and G_A, find other genes that are like G_A, based on M.** Like the previous problem, except now we must define some similarity expression to apply to the result of running $M(N, G_{Bad}, G_A)$ which will allow use to create a result set where all of the values for $M(N, G_{Bad}, G_i)$ are "similar". This similarity could simply be a range (find all G_i such that the value of M is within some tolerance of the value of M for G_A) or it could be something more complicated (find all G_i such that at least 50% of the graph nodes used to compute M for G_i were also used to compute M for G_A).
- **Given N, G_{Bad}, G_A, and M, find other genes that are like G_A based on metrics that are like M.** Like the previous problem, except now the metric is not explicitly defined. There are several ways to phrase this problem. Perhaps there is an ontology of centrality metrics M_i of which our M is only one; in this case, we could use ontological similarity to identify other candidate metrics to use. Perhaps we want to identify other metrics which are correlated to our M, in which case we could run many of them and examine their result sets. (The goal here would be to use a set of metrics defined by an exemplar M in order to capture corner cases which our M may have missed alone.)
- **Given N and M, find genes which are like each other based on metrics like M.** The logical conclusion, where we're seeking relationships within the gene network based on a family of metrics defined by the exemplar M. The constraints are now likely parameters for M or for finding metrics like M or for constraining the set of genes considered (but not to the degree of providing an example gene, as we did in the earlier queries with G_A).

4.4 Law Enforcement

This data has $\approx$ 100K elements/records about people, places, and events, and on the order of millions of relationships between records. It is in relational form, but with a rudimentary triple-based ontology.

Major features of the dataset include:

- gang activity, sex offenders, burglary, etc.
- Roles: witness, victim, suspect, perpetrator

- Events are temporal. Associations between people and addresses/phone number are temporal. Addresses are spatial, linked to people and organizations.
- Example of events: car prowl at a certain time, time of report of shots fired, not so much booking or jail-related, more focused on officers. However, history and records are important, like criminal records and people who were in jail together or shared a cell together.

Characteristics of useful queries include:

1. Shortest path queries ("Show how two people are connected") are very useful, but tend to blow up for time or memory.
2. Who is associated with crimes over a specific geographic region? The spacial component becomes very difficult for boundary cases such as a car accident in the middle of a street that serves as a boundary for the law enforcement zones.
3. When a crime occurs, queries are requested that perform the task of *landscaping*, or trying to find anything related to the new crime, such as similar victims, locations, etc.

Finally, it would be useful to be able to search for probable crime spree patterns. Here a crime spree is loosely defined as a pattern of similar crimes following a feasible spatio-temporal sequence. For example, give all crimes in the "neighborhood" (defined) of a given space+time, and do so iteratively for all crimes in the neighborhood, until the neighborhood can no longer be expanded. The sequence can no longer be expanded when the time between two events are such that there was not sufficient time to move from the previous location to the next. Since the "time" of the crime is a fuzzy term—when did a break-in happen given when it was first noticed?—this constraint may come down to a confidence factor dipping below some threshold.

5 Other Issues

Pursuant to the questions in Sec. 2 above, a number of other points of discussion ensued.

5.1 What makes a query hard?

To develop benchmarks it is necessary to judge the difficulty of specific queries against specific datasets. A number of observations were offered as to specifically what makes a graph query "hard":

- The expressivity of the query language is a barrier.
- Spatial-temporal queries: for example, "how has gang X transitioned over the last 10 years?"
- A problem that has an indeterminate (probably bounded) number of "joins".

– A complicated inference requirement. For example, there is a Mayo problem of infering familial relations, which can be relegated to inference rules.
– The number of "dimensions" (orthogonal characteristics). Dimensions may correspond to different data modalities.
– Joins with low selectivity of data and/or high cardinality.

5.2 Query Complexity Scale

A description of an abstract complexity scale was offered, which one might use to increase or decrease the difficulty of other queries.

1. **All of the constraints, all of the values.** A basic graph lookup problem. You know exactly what you're looking for, even if the constraints are hard to match or expensive to compute.
2. **All of the constraints, exemplar values.** Instead of specifying what you're looking for, you give an example of what you're looking for and expect the system to use (specified) approximation metrics to expand the result set.
3. **Exemplar constraints, exemplar values.** You specify both a model for what you're looking for and how you're looking for it, but you expect the system to use (specified) approximation metrics to expand both the set of values and constraints to produce a result set.
4. **Exemplar constraints, no values.** You are looking for all things related by constraints defined by an instance of a constraint. This is more like clustering and less like graph pattern lookup—heavy on the algorithm parameterization and light on the specifics.

5.3 Benchmark Criteria

There was discussion about what makes a valuable benchmark. It was observed that benchmarks are useful to everyone to make rationale decisions, but can and will be abused. In order to mitigate abuse and maximize utility, the following suggestions were made:

– **Identify a target.** An analyst will care about very different metrics than a developer, so make sure the benchmark's features address the expected user.
– **Address performance and expressivity.** A benchmark is useful not only in determining what a tool can do quickly, but what it can do *at all*.
– **Provide tuning knobs.** Benchmarks are meant to be used across many different environments and for many different users, so they must be adjustable depending on the situation. The results from a simple lookup on a 1K-node graph running on a workstation should be on the same (multi-dimensional) sliding scale as a path query on a 1B-node graph running on a supercomputer, not because the two are necessarily meant to be compared, but because a user needs to be able to make a close approximation to their environment. Results from a benchmark which is run with many different parameter settings allows a user to identify which settings they care most about and then rationalize about the data at that point.

– **Data should be real or synthetically generated based on measured statistics.** Real data is noisy and usually much more diverse than purely synthetic data. As one participant pointed out, "when you use [purely] synthetic data, it becomes too easy to structure and cheat." Another followed up with, "in the real world, the 'right' answer is not easy to come by." His meaning is that sometimes it is impossible to actually know the correct answer to a question; queries are merely approximations that might (or might not) accurately capture some aspect of the solution.

5.4 Probabilistic Models

Probability came up a number of times with respect to both data and inference. Mayo and Oracle both deal with statistical datasets, and both would like to allow rules (e.g.-Bayesian inference). Neither believes that it is sufficient to "round up" and use boolean truth values for probabilistic quantities.

5.5 Interaction with Relational Data

Both users and developers agreed that tightly-coupled interaction with relational data is necessary for real-world applications. Unfortunately, nobody could agree on how that interaction should be built. Some thought that semantic and relational data should be mixed; queries should be able to access either seamlessly. But that can come with a price (treating sparse, semantic data as relational can kill performance). Others believed the two should be complementary; the data should exist in both locations, and queries should be asked of whichever set is most appropriate.

Acknowledgments

We would like to thank Cray Inc. and Edie Birk from Cray for their support of this workshop. Debbie Scherr from PNNL also provided invaluable assistance.

We would additionally like to recognize our organizing committee: Jon Berry and Eric Goodman (Sandia National Laboratories); David Mizell (Cray Inc.); David Silberberg (Johns Hopkins University/Advanced Physics Lab); and Stefan Weitz (Microsoft).

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.

References

1. Yuanbo Guo, Zhengxiang Pan, Jeff Heflin: (2005) "LUBM: A benchmark for OWL knowledge base systems", *Web Semantics: Science, Services and Agents on the World Wide Web*, 3:2-3
2. Schmidt, Michael; Hornung, Thomas; Küchlin, Norbert; Lausen, G; and Christoph Pinkel: (2008) "An Experimental Comparison of RDF Data Management Approaches in a SPARQL Benchmark Scenario", in: *Proc. 7th Int. Conf. Semantic Web (ISWC 08)*, pp. 82-97, doi>10.1007/978-3-540-88564-1_6