Indexing Social Semantic Data

Poster Abstract

George H. L. Fletcher Peter W. Beck

School of Engineering and Computer Science
Washington State University, Vancouver, USA
{fletcher, pwbeck}@wsu.edu

1. INTRODUCTION

Personal information management (PIM) is concerned with automating the processes of collecting, organizing, and securely storing personal data such as music, text documents, spreadsheets, email, financial and medical records, bookmarked webpages, calendars, notes, address books, pictures, chat logs, etc. [6, 7]. PIM systems must support non-technical casual users in capturing, querying, and exploration of this wide variety of semantically rich personal data. PIM systems must also facilitate seamless sharing of personal data in web-scale social collaboration networks. PIM is an area of intense interdisciplinary investigation and is a vital facet of the social semantic web vision [7].

To be successful, PIM solutions must be built on top of a robust data management infrastructure. Such an infrastructure must efficiently and unobtrusively support the requirements of PIM. At present, the design of data management infrastructures for PIM is in its infancy [6]. In particular, indexing, a fundamental data management technology, is still not well understood in this domain. Indexes are necessary for efficient querying and exploration of data. This poster will describe our ongoing efforts to design index structures specifically tailored to the social semantic data managed by PIM systems.

2. RDF AND BASIC GRAPH PATTERNS

The W3C data model RDF and its query language SPARQL have emerged as the standards for representing and querying social semantic data [4, 6]. In RDF, information resources are represented by URIs and relationships between resources are captured as triple statements. Figure 1 illustrates a small subset of a personal triple store. In SPARQL, queries over triple stores are posed in an SQL-like syntax.

EXAMPLE 1. *Consider the query "What are the dates and types of documents on which McShea was a performer?" In SPARQL, where variables are identified by a leading* `?`*, this query can be posed against the triple store in Figure 1 as follows:*

```
SELECT ?date ?type
WHERE { McShea performed  ?doc  .
        ?doc   created_on ?date .
        ?doc   type       ?type  }
```

The `WHERE` *clause of the query specifies a* basic graph pattern *(BGP), via a set of simple access patterns. BGP's, which are at the heart of all SPARQL queries, identify a subset of related resources to be extracted from the RDF graph, which is then returned as a set of variable mappings. In this case, we have only one set of valid bindings for the output variables specified in the* `SELECT` *clause:* {?date : `26.10.08`, ?type : `MP3`}.

Intense research efforts are currently focused on BGP query evaluation techniques (e.g., [8]). Complementing this ongoing research, we are interested in designing *native* RDF index data structures to accelerate BGP query evaluation.

3. NATIVE RDF INDEXING

A BGP can be represented as a "join" graph, wherein each simple access pattern in the BGP is a node and an edge exists between two nodes if they share a variable. For example, the BGP of Example 1 can be visualized as in Figure 2. The width k of a join graph is the length of the longest direct path in the graph. In Figure 2, the graph has width $k = 1$.

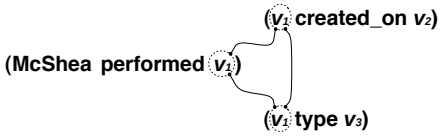


Figure 2: Join graph of BGP of Example 1.

Figure 3 illustrates a slightly richer example join graph for the query "Who has authored a document performed by someone (socially) related to McShea?"; here, width $k = 3$.

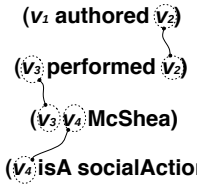
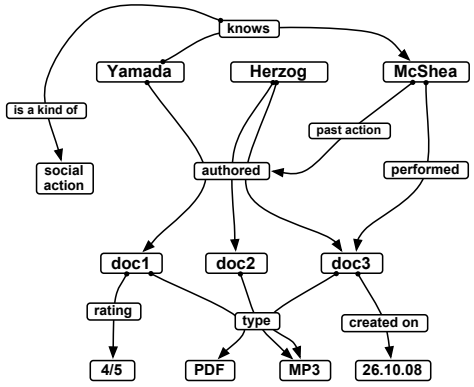


Figure 3: A wider BGP join graph.

Clearly, with massive RDF databases, disk-based index data structures are necessary to efficiently process "wide" BGP's (i.e., where $k > 0$) such as those in Figures 2 and 3. Currently, many RDF data management systems utilize index structures which facilitate efficient look-up of individual simple access patterns. For example, [5, 11] use the classic B+tree data structure for such look-ups. These "$k = 0$" approaches only support evaluation of BGP's with join graphs containing no edges. In this sense, these are not "native" indexes since they do not reflect the inherent graph structure of BGP's and RDF data. Recently, there have been proposals for native $k = 1$ index data structures, e.g., [1, 10];

```
{⟨Yamada,  authored,      doc1⟩,
 ⟨Yamada,  knows,         McShea⟩,
 ⟨knows,   is a kind of,  social action⟩,
 ⟨Herzog,  authored,      doc2⟩,
 ⟨Herzog,  authored,      doc3⟩,
 ⟨McShea,  performed,     doc3⟩,
 ⟨McShea,  past action,   authored⟩,
 ⟨doc1,    type,          PDF⟩,
 ⟨doc1,    rating,        4/5⟩,
 ⟨doc2,    type,          MP3⟩,
 ⟨doc3,    type,          MP3⟩,
 ⟨doc3,    created on,    26.10.08⟩}
```

(a) A triple graph



(b) A visualization of this graph

Figure 1: A small subset of a personal triple store.

however, these have either focused on specific join patterns or are limited to main-memory data structures. Indexing techniques have also been developed for special classes of larger patterns, e.g., [9]; such techniques, however, do not support processing of the full variety of BGP join patterns.

The development of native disk-based index data structures for wide BGP's is crucial. Recently, a robust generic methodology for designing indexes has been developed for XML data [3]. This approach hinges on coupling query language induced partitions of the database with a structural partitioning of the database. Such partitions are the basis of engineering index data structures which are ideally suited for efficient query processing. Through a theoretical analysis, we have shown that such an approach can also be successfully leveraged in the development of native indexes for RDF data [2]. Indeed, we have characterized the partition on resources induced by various k-width fragments of BGP, for $k \geqslant 0$. Evaluation of k-width bounded BGP's with arbitrary join patterns can be directly computed on these partitions.

4. POSTER PRESENTATION

We are currently investigating the theoretical foundations and practical engineering of disk-based native indexes for efficient evaluation of wide BGP's over massive collections of social semantic triple data.[1] We have successfully designed, implemented, and empirically evaluated an efficient disk-based $k = 1$ index data structure, thus demonstrating the feasibility of native indexing of RDF for an important fragment of BGP. Will we discuss how this data structure efficiently supports the full range of $k = 0$ and 1 join patterns in our poster. Based on this empirical investigation and on the theoretical foundations established in [2], we are now designing and evaluating a robust disk-based index data structure to accelerate processing of $k > 1$ BGP's. The balance of the poster will highlight our progress on this investigation.

The development of native indexes significantly advances the state of the art in RDF data management. Such indexes will serve as a key component in the engineering of successful PIM (and, more broadly, semantic web) systems. The poster will present the current results of our ongoing research into the development of efficient disk-based RDF indexes. We hope to receive critical feedback from the community during our presentation.

[1]We are using the DBPedia (`http://wiki.dbpedia.org`) and UniprotRDF (`http://dev.isb-sib.ch/projects/uniprot-rdf`) datasets in our experiments, each of which is on the order of 10^8 triples.

5. REFERENCES

[1] D. J. Abadi, A. Marcus, S. Madden, and K. J. Hollenbach. Scalable Semantic Web Data Management Using Vertical Partitioning. In *VLDB*, pages 411–422, Vienna, 2007.

[2] G. H. L. Fletcher. An Algebra for Basic Graph Patterns. In *LID*, Rome, 2008.

[3] G. H. L. Fletcher, D. Van Gucht, Y. Wu, M. Gyssens, S. Brenes, and J. Paredaens. A Methodology for Coupling Fragments of XPath with Structural Indexes for XML Documents. In *DBPL*, pages 48–65, Vienna, 2007.

[4] C. Gutiérrez, C. A. Hurtado, and A. O. Mendelzon. Foundations of Semantic Web Databases. In *ACM PODS*, pages 95–106, Paris, 2004.

[5] A. Harth, J. Umbrich, A. Hogan, and S. Decker. YARS2: A Federated Repository for Querying Graph Structured Data from the Web. In *ISWC*, Busan, Korea, 2007.

[6] W. Jones and J. Teevan, editors. *Personal Information Management*. University of Washington Press, Seattle, 2007.

[7] m. c. schraefel. What is an Analogue for the Semantic Web and Why is Having One Important? In *ACM Hypertext*, pages 123–132, Manchester, UK, 2007.

[8] M. Stocker, A. Seaborne, A. Bernstein, C. Kiefer, and D. Reynolds. SPARQL Basic Graph Pattern Optimization Using Selectivity Estimation. In *ACM WWW*, pages 595–604, Beijing, 2008.

[9] O. Udrea, A. Pugliese, and V. S. Subrahmanian. GRIN: A Graph Based RDF Index. In *AAAI*, pages 1465–1470, Vancouver, B.C., 2007.

[10] C. Weiss, P. Karras, and A. Bernstein. Hexastore: Sextuple Indexing for Semantic Web Data Management. In *VLDB*, Auckland, New Zealand, 2008.

[11] G. Wu, J. Li, and K. Wang. System II: a Hypergraph Based Native RDF Repository. In *WWW*, pages 1035–1036, Beijing, 2008.