

Managing Large Scale Native RDF Semantic Repository from the Graph Model Perspective*

Gang Wu
wug03@mails.thu.edu.cn

Juanzi Li
ljz@keg.cs.tsinghua.edu.cn

Department of Computer Science, Tsinghua University, Beijing, P.R.China 100084

ABSTRACT

We propose a set of solutions for managing a large scale RDF semantic repository from the perspective of RDF graph model. A native storage instead of relational database is used to hold RDF. Indices supporting regular path expression, full-text retrieval and partial OWL Lite inference are built above the storage model. Semantic ranking for resources are provided as well.

Categories and Subject Descriptors: H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval

General Terms: Algorithm, Experimentation

Keywords: Semantic Web, RDF, Index, Rank, Reasoning

1. INTRODUCTION

The RDF graph model is the foundation for representing Semantic Web documents formatted in RDF/S and OWL. Since an increasing amount of information is being represented as Semantic Web documents, it is urgent to provide an efficient semantic repository to manage large scale information over the RDF graph model. This challenges both database and information system researchers in the efficiency, and the quality of storing, querying and reasoning.

Though some RDF repositories have been developed toward this aim, e.g. memory based, relational DB based, and native systems, none is superior to the others[2]. There exists room for improvement, especially in the following topics. **i) Storage Scheme.** Since the RDF model is a directed labeled graph, it requires techniques for accelerating data access and reducing data redundancy over graph structure. **ii) Special Indices.** In order to satisfy manifold requirements in regular path expression (RPE) querying, full-text retrieval, and OWL-Lite inference, a RDF repository should provide efficient index structures correspondingly. **iii) Ranking scheme.** Though high quality and

*This work is supported by the National Natural Science Foundation of China under Grant No. 90604025.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Proceedings of SIGMOD2007 Ph.D. Workshop on Innovative Database Research 2007(IDAR2007), June 10, 2007, Beijing, China.

convincing results for querying and reasoning are important, no ranking scheme is included in the existing RDF repositories.

In this paper, we propose a set of solutions for managing a large scale RDF repository according to the graph model. The output of this research is a scalable, efficient and quality native RDF repository prototype.

1.1 Preliminaries

A RDF graph consists of a set of RDF triples, each of which represents a statement with the form of (*subject*, *predicate*, *object*). Subject, predicate (also know as property), and object are three important components of RDF. Let RDF graph $G = (V, E, l_V, l_E)$, where $V = \{v_x | x \in \text{subject}(T) \cup \text{object}(T)\}$, $E = \{e_{s,p,o} | (s, p, o) \in T\}$,

$$l_V(v_x) = \begin{cases} (x, d_x) & \text{if } x \text{ is literal } (d_x \text{ is datatype}) \\ x & \text{else;} \end{cases}$$

, and $l_E(e_{s,p,o}) = p$. V is the set of vertices in the RDF graph, and E is the edge set. Label functions on V and E are represented as l_V and l_E respectively. T represents the set of triples. Advanced Semantic Web languages such as RDFS and OWL are semantic extensions of RDF.

2. NATIVE STORAGE SCHEME

Some researchers show that the directed labeled graph representation of RDF is not suitable for persistency because it is possible for a resource being both the predicate of one statement and the subject or object of another statement. Fortunately, a RDF graph could be transformed to an ordered *hypergraph*[3], where $\mathcal{V} = \{v_x | x \in \text{subject}(T) \cup \text{object}(T) \cup \text{predicate}(T)\}$, and $\mathcal{E} = \{(v_s, v_p, v_o) | (s, p, o) \in T\}$. As illustrated in Figure 1, we use two kinds of persistent objects, “GraphVertex” and “GraphEdge”, for storing elements in \mathcal{V} and \mathcal{E} respectively. The “outEdges” of a “GraphVertex” is the container of all the references to the triples taking it as the subject, and the “inEdges” is for the object. The “srcVertex”, the “typeVertex”, and the “tarVertex” of a “GraphEdge” are the references to the subject, the predicate, and the object of the triple. We implement such

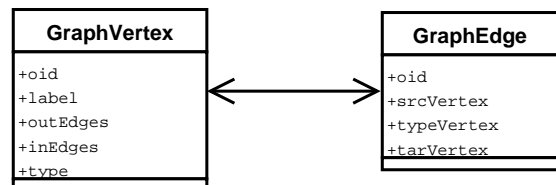


Figure 1: Classes Diagram for Hypergraph a native storage scheme in Berkeley DB. Each resource and

each statement appear only once as a “GraphVertex” and a “GraphEdge” respectively with a unique “oid”, but could be referenced many times. In this way, it greatly reduces the storage redundancy. We also find it convenient to cluster objects and support graph navigation.

3. INDICES DESIGN

3.1 Index for Regular Path Expression

Similar to the techniques for RPE query processing used in XML DB[5], we provide a value index and a labeling based index to replace the time-consuming naive graph traversal on the storage. The *Resource Value Index* is used to locate a resource with a given URI value. This index is simple but effective. The reason is that any resource is identified uniquely by URI. In term of our physical storage model, a *Hash* index is built by taking URI as key and oid as value. Furthermore, we employ a sequence labeling scheme to improve the performance in matching RPE. It avoids costly join operations inhabiting in the RDF repositories that only store RDF as the set of loose coupling statements. We manage to apply the Prüfer[4] sequence in graph labeling for its ability in one-to-one mapping a tree to the generated sequence. However, the Prüfer’s method cannot be employed directly. The RDF graph may contain cycles, and have labels in both nodes and edges. The former problem is solved by giving a order to all the resources in the cycle according to oid. We solve the latter problem by involving the label of the edge into Prüfer’s method. The evaluation to RPE queries are converted to subsequence matchings between the sequence of the graph and the sequence of the query.

3.2 Index for Partial OWL Lite

Besides retrieving explicit information, inference of implicit information is also necessary for a RDF repository. Inspired by the idea of HStar[1], we provide index structures to improve the scalability and the completeness of the OWL Lite inference. We find that all inference rules depend on subsumption relations in the Classes and Properties hierarchies together with the characters of the Properties. Hence, we developed a prime number labeling scheme for DAG[7] to index those subsumption relations. The scheme invests intrinsic mapping between the integer divisibility and the subsumption hierarchy. It diminishes the transitive closure computations and the storage redundancy. We give each resource vertex on the hierarchy subgraph an integer label whose value equals to the arithmetic product of a group of prime numbers associating with the vertex and all its ancestors respectively. By checking the divisibility between the labels, the inheritance and equivalence relation required in the OWL Lite inference could be determined easily.

3.3 Index for Full-text Retrieval

In practice, keyword based queries are more convenient than structural expressions for common users to exert in order to discover information in a RDF repository where a large proportion of information is the text value of literal properties. Hence, we build a full-text index in our system. Traditional full-text index structures take the text content in a Web page as a document for indexing, while we analogize a resource to a document virtually. The virtual document consists of all the keywords contained in the values of its literal properties. For each term, we index the “oid”s of

those “GraphEdge”s where the term appears in the “label” of their “tarVertex” field. Given a term, we can locate the statements containing it with such a fine-grained inverted list index and then expand the result set by navigating on the RDF graph starting from there.

4. SEMANTIC RANKING

We propose SWRank[6] as a semantic web ranking approach which applies an object-level link analysis reversely to the direction of relations and consistently across the schema part of the graph and the data part of the graph. A set of resources are returned according to the overall ranking score.

Our observations show that SWRank demonstrates typical hub score characteristics. Therefore, we reverse the direction of all the edges in a RDF graph and apply PageRank on the reversed RDF graph. Let SWRank $R(v_x)$ be the global importance of an entity $v_x \in V$ in a RDF graph.

$$R(v_x) = (1-\alpha)E(v_x) + \alpha \sum_{v_y \in F(v_x)} \frac{w(e_{v_x,p,v_y})R(v_y)}{\sum_{v \in B(v_y)} w(e_{v,p',v_y})} \quad (1)$$

where α is a constant between 0 and 1; $w(e_{v_x,p,v_y})$ is the weight of the labeled edge representing the statement (v_x, p, v_y) ; and $E(v_x)$ is a population containing entities corresponding to a source of rank with uniform probability distribution. Keyword-based ranking is implemented by varying traditional vector space model. For a query Q , the similarity equals the *cosin* value of the angle between vector \vec{Q} and vector $\vec{v_j}$. When considering both the global SWRank score and the similarity from keyword-based ranking, we have $R(v_j, Q) = \text{sim}(v_j, Q)(R(v_j))^g$ where g is the *global weight* which determines the importance of SWRank among the overall rank score.

5. CONCLUSION

We discuss topics related to the efficiency and quality of large scale RDF repository management, including the storage scheme, indexing techniques, and the ranking scheme. All of our solutions toward these topics consider intrinsic characters of the RDF graph model for improvements.

6. REFERENCES

- [1] Y. Chen, J. Ou, Y. Jiang, and X. Meng. Hstar - a semantic repository for large scale owl documents. In *ASWC 2006*, pages 415–428, 2006.
- [2] Y. Guo, Z. Pan, and J. Heflin. Lubm: A benchmark for owl knowledge base systems. *Journal of Web Semantics*, 3(2-3):158–182, October 2005.
- [3] J. Hayes. A graph model for rdf. Master’s thesis, August 2004.
- [4] P. Rao and B. Moon. Sequencing xml data and query twigs for fast pattern matching. *ACM Trans. Database Syst.*, 31(1):299–345, 2006.
- [5] G. Wang, B. Sun, J. Lv, and G. Yu. Rpe query processing and optimization techniques for xml databases. *J. Comput. Sci. Technol.*, 19(2):224–238, 2004.
- [6] G. Wu, K. Zhang, and J. Li. Swrank: Ranking semantic web reversely and consistently. In *ODDIS: VLDB Workshop*, 2006.
- [7] G. Wu, K. Zhang, C. Liu, and J.-Z. Li. Adapting prime number labeling scheme for directed acyclic graphs. In *DASFAA*, pages 787–796, 2006.