Ontology-based Adaptive Reasoning Service for ScienceWeb

Hui Shi, Kurt J. Maly, and Steven J. Zeil
Department of Computer Science
Old Dominion University
Norfolk, VA
hshi/maly/zeil@cs.odu.edu

***Abstract*—ScienceWeb is a system that provides answers to qualitative and quantitative queries of a large knowledge base covering science research. The system will support the community joining together, sharing the insights of its members, to evolve the large knowledge base. ScienceWeb will need to scale to accommodate the substantial corpus of information about researchers, their projects and their publications. It will need to accommodate the inherent heterogeneity of both its information sources and of its user community. A reasoning system supports the queries and scalability becomes a serious challenge. In this paper we describe experiments and the resulting reasoning architecture and services whose scalability and efficiency are able to meet the requirements of query and answering in ScienceWeb. One key element of the services is an adaptive combination of both Query-invoked Inference and Materialization Inference together with incremental inferencing for more efficient query and answering. Second, we introduce new ways of storing, grouping and indexing objects in the knowledge base for faster searching and reasoning as the size of the triple set scales to the millions and the complexity of the Abox increases. The adaptive reasoning architecture and resulting adaptive reasoning service should provide efficient reasoning based on a scalable knowledge base.**

***Keywords*-knowledge bases; ontologies; reasoning**

I. INTRODUCTION

Consider a potential chemistry Ph.D. student who is trying to find out what the emerging areas are that have good academic job prospects. What are the schools and who are the professors doing groundbreaking research in this area? What are the good funded research projects in this area? Consider a faculty member who might ask, "Is my record good enough to be tenured at my school? At another school?" Similarly consider a National Science Foundation (NSF) program manager who would like to identify emerging research areas in mathematics that are not being currently supported by NSF. It is possible for these people each to mine this information from the Web. However, it may take a considerable effort and time, and even then the information may not be complete, may be partially incorrect, and would reflect an individual perspective for qualitative judgements. Thus, the efforts of the individuals neither take advantage of nor contribute to others' efforts to reuse the data, the queries, and the methods used to find the data.

A number of projects (e.g., Arnetminer [1]) have built systems to capture limited aspects of community knowledge and to respond to semantic queries. However, these lack the level of community collaboration support that is required to build a knowledge base system that can evolve over time, both in terms of the knowledge it represents as well as the semantics involved in responding to qualitative questions. These systems are also homogeneous, in the sense that they harvest data from one type of resources. A team at ODU is working on ScienceWeb [2] [3], which will combine diverse resources such as digital libraries for published papers, curricula vitae from the web, and agency data bases such as NSF's research grant data base and that will use collaboration as the fundamental approach to evolve its knowledge base.

Collaboration is at the heart of the approach to build ScienceWeb. Such collaboration includes building and evolving the knowledge base, building, evolving and reusing queries and identifying, specifying methods and harvesting raw information. The interaction between the system and people must be effective enough to allow for collaborative development.

Reasoning over the knowledge base provides support for answering qualitative questions and quantitative questions, whose scalability and efficiency influence greatly the response time of the system. ScienceWeb is a system that collects various research related information. The more complete the knowledge base is, the more helpful answers the system will provide. As the size of knowledge base increases, scalability becomes a challenge for the reasoning system. It may handle millions units of reasoning items in the knowledge base. As users make changes to the basic descriptors of the knowledge base, fast enough response time (ranges from seconds to a few minutes) in the face of changes is one of the core challenges for the reasoning system.

In this paper, we describe an adaptive reasoning architecture and an adaptive reasoning service whose scalability and efficiency are able to meet the interaction requirements in ScienceWeb system when facing a large and evolving knowledge base. The remainder of this paper is organized as follows: Section II describes the prior research relevant to this paper. Section III describes the work already done to implement parts of the ScienceWeb system. Section IV describes the adaptive reasoning service for ScienceWeb.

 ISBN: 978-1-61208-152-6

Section V presents the conclusions.

II. BACKGROUND

There are a large number of knowledge bases for a variety of domains.

An example of a sophisticated knowledge base in Computer Science is Arnetminer [1], which provides profiles of researchers, associations between researchers, publications, co-author relationships, courses, and topic browsing. It has the capability to rank research and papers. It is a centrally developed system with fixed queries and schemas for data, but the knowledge base is continually growing as new data become available.

A. Ontologies

There has been increasing effort in organizing web information within knowledge bases using ontologies. Ontologies can model real world situations, can incorporate semantics that can be used to detect conflicts and resolve inconsistencies, and can be used together with a reasoning engine to infer new relations or proof statements. For example, the DBpedia [4] project focuses on converting Wikipedia [5] content into structured knowledge.

A number of tools exist for collaboratively designing and developing ontologies: for example, CO-Protégé [6], and Kaon2 [7].

Research in the field of knowledge representation and reasoning usually focused on methods for providing high-level descriptions of the world that can be effectively used to build knowledge-based systems. These knowledge-based systems are able to get implicit consequences of its explicitly represented knowledge. Thus, approaches to knowledge representation are crucial to the ability of finding inferred consequences [8]. Early knowledge representation methods such as frames [9] and semantic networks [10] lack well-defined syntax and a formal, unambiguous semantics, which are elements of qualified knowledge representation. Description Logic (DL) was therefore introduced into knowledge representation systems to improve the expressive power. Description Logic [11] is a family of logic-based knowledge representation formalisms, which is designed to represent the terminological knowledge from an application domain [12].

A DL knowledge base analogously consists of two parts: *intentional knowledge* (TBox), which represents general knowledge regarding a domain, and *extensional knowledge* (ABox), which represents a specific state of affairs. The "T" in the term "TBox" denotes terminology or taxonomy, which is built based on the properties of concepts and the subsumption relationships among the concepts in the knowledge. The "A" in the term "ABox" denotes assertional knowledge that includes individuals of the specific domain. [8] [13]

B. Inference Methods

The main reasoning tasks for DL reasoners are verifying KB consistency, checking concept satisfiability, concept subsumption and concept instances [12]. Algorithms for reasoning in DLs are: structural subsumption algorithms [14], the resolution-based approach [15], the automata-based approach [16] [17], and the tableau-based approach [18], which is currently the most widely used reasoning algorithm for DLs. There are three kinds of inference methods for First Order Logic (FOL): Forward chaining, Backward chaining and Resolution [19].

Materialization and Query-rewriting are the most popular inference strategies adopted by almost all of the state of the art ontology reasoning systems. *Materialization* means pre-computation and storage of inferred truths in a knowledge base, which is always executed during loading the data and combined with forward-chaining techniques. *Query-rewriting* means expanding the queries, which is always executed during answering the queries and combine with backward-chaining techniques.

Materialization and forward-chaining are suitable for frequent, expensive computation of answers with data that are relatively static. Owlim [20] [21], Oracle 11g [22], Minerva [23] and DLDB-OWL [24] all implement materialization during loading of the data. Materialization permits rapid answer to queries because all possible inferences have already been carried out. But any change in the ontology, instances, or custom rules requires complete re-processing before responding to any new queries. Furthermore, a large amount of redundant data may be produced by materialization of a large knowledge base, which may slow the subsequent loading and querying.

Query-rewriting and backward-chaining are suitable for efficient computation of answers with data that are dynamic and infrequent queries. Virtuoso [25] is one system that implements dynamical reasoning when it is necessary. This approach improves the performance of answering new queries after data changes and simplifies the maintenance of storage. But frequent repeated queries in query-rewriting will require repeated reasoning, which is time-consuming compared to pure search in materialization. Hstar [26] attempts to improve performance by adopting a strategy of partially materializing inference data instead of complete materializing.

A hybrid approach may give the best of both worlds. Jena [27] supports three ways of inferencing: forward-chaining, backward-chaining and a hybrid of these two methods. An adaptive hybrid approach would combine the strong points of both patterns for better performance under changing circumstances.

C. Storage Scheme

Semantic web applications can contain large amounts of data conformed to the ontology. How to store these large

 ISBN: 978-1-61208-152-6

amounts of data and how to reason with them becomes a challenging.

Generally, there are two main kinds of ontology stores, native stores (disk-based stores) and database-based stores. Native stores are built on the file system, while database-based stores use relational or object relational databases as the back-end store.

Examples of native stores are OWLIM [20] [21], Allegro-Graph [28], Sesame Native [29], Jena TDB [27], Hstar [26] and Virtuoso [25].

Representative database-based stores are: Jena SDB [27], Oracle 11g R2 [22], Minerva [23], (Sesame + MySQL) [29], DLDB-OWL [24] and (Sesame+ PostgreSQL) [29]. They all take advantage of existing mature database technologies for persistent storage.

The advantage of native stores is that they reduce the time for loading and updating data. However, a disadvantage of native stores is they are not able to make direct use of the query optimization features in database systems. Native stores need to implement the functionality of a relational database from the beginning, such as indexing, query optimization, and access control. As for database-based stores, the advantage is that they are able to make full use of mature database technologies, especially query optimization while the disadvantage is that they may be slower in loading and updating data,

D. Reasoning over Large ABoxes

Due to the large size of instance data conformed to corresponding ontology in many knowledge bases, reasoning over large ABoxes has become an issue in the fields of Semantic Web and Description Logics. There are two main kinds of approaches to dealing with this issue. The first approach includes designing novel algorithms, schemes and mechanisms that enhance the reasoning ability on large and expressive knowledge bases. Compared with some used state-of-the-art DL reasoners such as Racer, FaCT++, and Pellet, Kaon2 has been shown to have better performance on knowledge bases with large ABoxes but with simple TBoxes [30]. The second approach adopts simplification by reducing the expressive power of TBoxes describing large ABoxes. Calvanese et al. [31] have proposed a new Description Logic, called DL-Lite, which is not only rich enough to capture basic ontology languages, but also requires low complexity of reasoning.

We summarize the different ontology based reasoning systems along with the features they support in Table I.

III. SCIENCEWEB

This section describes the work we have already done to implement parts of the ScienceWeb system that includes the design of the architecture of ScienceWeb, the ontology in ScienceWeb and a synthetic data generator, the comparison of ontology reasoning systems, the study on a selected benchmarks and a formative study in Virginia.

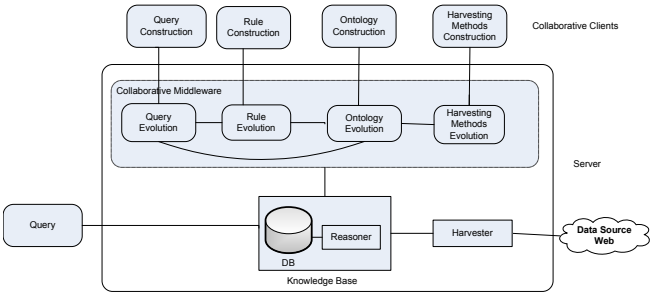

Figure 1. Architecture of ScienceWeb

A. Architecture

ScienceWeb is a platform where researchers including faculty, Ph.D. students and program managers can collaboratively work together to get answers of their queries from a consensus point of view or from their specific point of view. The collaborative aspect is not only in the construction of queries but in the construction of the underlying ontology, rules and instance data. The proposed architecture of the ScienceWeb is shown in Figure 1.

A traditional data mining architecture involves harvesting from data sources to populate a knowledge base, which in turn can then answer queries about the harvested content. We propose to enhance this architecture by adding a layer of collaborative clients for construction of queries, rules, ontological concepts, and harvesting methods, mediated by a layer of server functions that oversee and support the evolution of each of those functional groups within the knowledge base.

The system is built, developed and evolved based upon users' collaborative contributions. Users contribute during querying & answering, harvesting and ontology evolution. Querying is not an ordinary job of posting, parsing and retrieving as in a conventional database. Instead, it becomes an interactive, collaborative process. Harvesting and ontology evolution also benefit from the information provided by the users. Thus, collaboration is critical and widely spread throughout the system.

B. Performance of Existing Reasoning Systems

As described in Section II, there have been a number of studies on reasoning systems using only their native logic. To provide credibility for our context, we used benchmark data from these studies, replicate their results with native logic, and then extend them by adding customized rules. We used LUBM [32] for comparing our results with earlier studies. The second set of data will emulate a future ScienceWeb. Figure 2 shows the limited ontology class tree we deem sufficient to explore the scalability issues.

Both the LUBM and the ScienceWeb ontologies are about concepts and relationships in a research community. For instance, concepts such as `Faculty`, `Publication`, and

 ISBN: 978-1-61208-152-6

Table I
GENERAL COMPARISON AMONG ONTOLOGY REASONING SYSTEMS

	Supports RDF(S)?	Supports OWL?	Rule Language	Supports SPARQL Queries?	Repository: Persistent(P)/ In-Memory (M)
Jena	yes	yes	Jena Rules	yes	M
Pellet	yes	yes	SWRL	no	M
Kaon2	yes	yes	SWRL	yes	M
Oracle 11g	yes	yes	Owl Prime	no	P
OWLIM	yes	yes	Owl Horst	yes	P

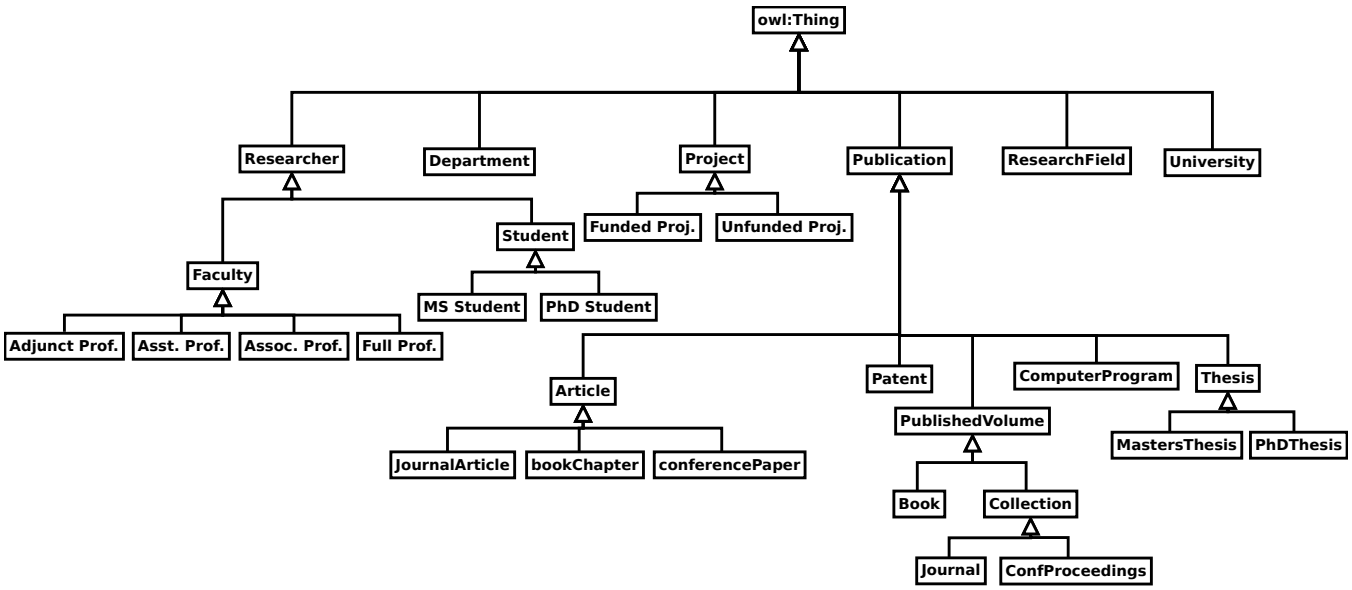

Figure 2. Class tree of research community ontology

`Organization` are included in both ontologies, as are properties such as `advisor`, `publicationAuthor`, and `worksFor`. All the concepts of LUBM can be found in the ScienceWeb ontology, albeit the exact name for classes and properties may not be same. ScienceWeb will provide more detail for some classes. For example, the ScienceWeb ontology has a finer granularity when it describes the classification and properties of Publication. The ontology shown in Figure 2 represents only a small subset of the one to be used for ScienceWeb. It was derived to be a minimal subset that is sufficient to answer a few select qualitative queries. The queries were selected to test the full capabilities of a reasoning system and to necessitate the addition of customized rules.

In support of the performance analysis described above, a flexible system for generating benchmark knowledge base instances of varying sizes has been developed [3]. The major challenge for this generator was to not only produce ontology-conformant data sets of the desired size, but to guarantee a plausible distribution for the many properties that relate objects across the knowledge base.

We performed a study to compare the scalability of existing reasoning systems when answering queries with customized rules in addition to native logic. We selected both LUBM and our own UnivGenerator [3] to provide the sample data and defined 5 custom rule sets for this experiment. Based on the comparison among these state-of-the-art ontology reasoning systems on full rule sets and transitive rule, we found [2] that OWLIM and Oracle offer the best scalability for the kinds of datasets anticipated for ScienceWeb, but they both have heavy loading and negative implications for evolving systems.

C. The Need for Agile Reasoning

The authors conducted a survey of faculty and PhD students at several universities in Virginia, aimed at determining both the demand for a ScienceWeb-like facility and the demands likely to be placed upon such a system [33]. Due to the similar faculty and student structure among universities in the USA, we believe this survey is likely representative of university communities in the USA. We have not yet explored differences likely to arise in a more international

 ISBN: 978-1-61208-152-6

context. Notable results include:

- The perceived utility of a knowledge base with the proposed scope of ScienceWeb is high. Respondents indicated an interest in use of such information for both research and hiring/promotion purposes (including, apparently, job hunting by those expecting to acquire new PhDs). More than 90% of respondents indicated that they would have used such a system, had it been available, in the past year, to find information regarding hiring and promotion, with more than half believing they would have used such a system multiple times. Information on publications and projects had even higher perceived value, with 100% of respondents indicating that they would have used such a system multiple times in the past year.
- There is rather lower willingness to invest time in collaborative construction of such a system than might have been expected. Slightly over 40% of respondents would want to devote no more than a few minutes at a time to such activities, with another 30% willing to spend more than a few minutes but less than an hour.
  A notable exception to that rule is in updating and making corrections to one's own personal information in such a knowledge base, where more than 25% of respondents indicated they would likely devote multiple hours to validating and correcting information. This suggests that identification of low-effort collaborative techniques will be essential to the success of a ScienceWeb.
- Respondents generally believed themselves to share a consensus with both research peers and departmental colleagues on quality judgments pertaining to research. For example, more than 70% agreed with the statement "I share a consensus with my research peers on the criteria that define a groundbreaking researcher in my field."
  By contrast, they did not believe that they shared such a consensus on questions such as who might be good candidates for hiring or awarding of tenure, even with their Departmental colleagues. Less than 20% agreed with the statement "I share a consensus with the majority of my Departmental colleagues on the criteria for hiring a new assistant professor".
  This is an interesting distinction because one of the original motivations for ScienceWeb came from a desire for a trusted source of information to address disputes in the latter area.

One implication of these results is a system such as ScienceWeb must place a high premium on quick exploration of potential results. Users appear to be far more willing to wait for results to a "final" query than to engage in multiple iterations of time-consuming steps while composing a new query.

IV. Adaptive Reasoning Service for ScienceWeb

Although the technology for storing large knowledge bases is reasonably mature, the introduction of reasoning components into the query process poses a significant challenge. Our own preliminary study has shown that extant systems can provide fast responses to queries with substantial reasoning components, but only at a cost of extensive pre-processing, that is required to integrate new knowledge instances and new rules. In essence, there is a trade-off between scalability of query processing and the agility of the system in responding to changes in the supporting rules and evolution of the data.

A. Architecture

There are two competing strategies that can be used by this reasoning system: materialized inference, where all the rules in the system are fired at once and all inferred triples are generated, and query-invoked inference where relevant rules (TBox inference) in the system are fired after a query is accepted and partial inferencing (ABox inference) is done. To achieve the goal of improving the performance and scalability of reasoning, we propose an adaptive reasoning architecture that exploits both of these strategies. So far Jena is the only one system we found that contains both of these two strategies, but Jena's approach does not scale well to the size of knowledge base expected for ScienceWeb. The adaptive mechanism needs to determine what part of TBox inferencing is stable and what part of inferred instances should be temporary for incremental inferencing. For query-invoked inference the mechanism needs to determine what part of Tbox inferences should be pre-computed. As appropriate, the adaptive mechanism will switch between query-invoked inference and materialized inference. In addition, we introduce a long-term storage that will contain stable inferred instances from an incremental inference in materialized inference and pre-computed TBox inferences for a query-invoked inference. Any Abox results of a query-invoked inference will be saved in short-term storage. A query unaffected by changes to ontology, custom rules or instances can be answered by searching in the long-term storage. A query affected by changes to ontology, custom rules or instances can be answered by performing a query-invoked inference and searching the short-term storage.

The resulting architecture of an adaptive reasoning system is presented in Figure 3. The major modules comprising this architecture are as follows:

1) Input from users: A *query* is the basic way for users to search and retrieve information that they are interested. For example, "Who are the ground breaking researchers in Digital Libraries?" Here "ground breaking" is a qualitative descriptor that has been evolved by one (or more) user(s) by developing custom rules and researcher and digital library are classes in the ontology.

 ISBN: 978-1-61208-152-6

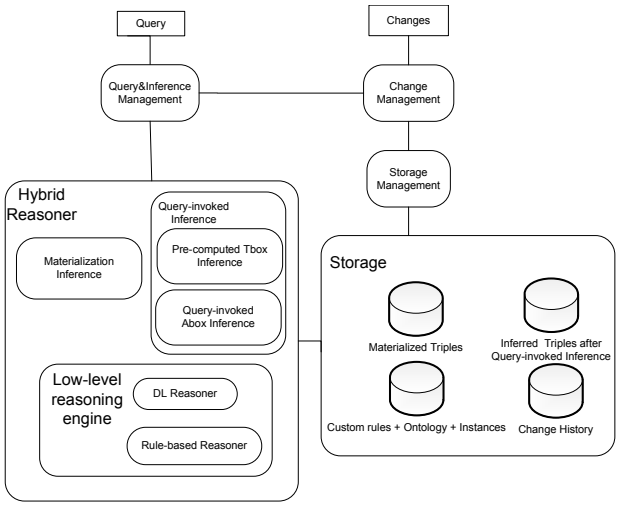

Figure 3. Architecture of an Adaptive Reasoning System

Changes refer to changes of ontology, custom rule set and instances as originally defined by users or harvested from the web. Generally, people might not agree on the ontology as it originally was designed and they will make changes according to their own beliefs. Similarly, custom rules represent a personal understanding of qualitative descriptors and as more people add their own opinion, they will change and, it is to be hoped, these qualitative descriptors will evolve to a consensus. The collection of instances will be enriched gradually with the discovery of new sources of information by individuals and the subsequent update of the methods of harvesting the information. Thus, changes of ontology, custom rule set and instances may occur at random or periodic times with varying degree of frequency. Changes have a significant influence on the process of storage and performance of query no matter whether the query involves inferencing or not.

2) Query & Inference Management: Query & Inference Management is the component that makes the choice between materialized inference and query-invoked inference. After a user submits a query, this component will determine if the query has been affected by the changes to the ontology, custom rules or instances by communicating with component of Change Management.

3) Change Management: Change Management maintains all the changes and arrangements in the Storage and records the change history after changes to the ontology, custom rules or instances. It not only provides change records to Query & Inference Management for the adaptive reasoning mechanism, but also communicates with the Storage Management module to realize the actual changes and operations in the storage.

4) Hybrid Reasoner: As a central component in the adaptive reasoning system, the Hybrid reasoner is a combination of Materialization Inference and Query-invoked Inference, and is responsible for the reasoning task with the assistance of a DL reasoner and a Rule-based reasoner.

Materialization Inference is one component that fires all of the rules in the system and generates all inferred triples at once. It implements all of the inference in advance following by various queries about the knowledge base from users. After Materialization Inference, answering query does not involve any reasoning but simple parsing and searching.

Query-invoked Inference is another component that invokes partial inference (ABox inference) when query is accepted after firing part of the rules (TBox inference) in the system in advance. Pre-computed TBox Inference is responsible for the TBox inference before queries while Query-invoked ABox Inference is responsible for the ABox inference during the query and answering.

5) Storage Management: The Storage Management module aims to update and to arrange the storage in an organized way to improve the scalability and performance of inferencing, especially the ABox inferencing. This component provides mechanisms to group and index base triples obtained from the users and the harvester module and triples that have been inferred such that search and updates can be done efficiently.

6) Storage: There are four separate storage areas of data in the system: Materialized Triples that are generated by Materialization Inference, Inferred Triples after Query-invoked Inference that are generated by Query-invoked Inference, Change History that is generated by Change Management and base storage including custom rules, ontology and instances as they were defined at startup of the system.

B. Adaptive Reasoning Mechanism

ScienceWeb is a collaborative platform that integrates efforts from users to define what data are to be obtained in what way and how the data are to be organized and what forms the queries will be. After a bootstrapping process has generated an initial knowledge base, we expect a period of frequent changes to all aspects of the knowledge base: ontology, rule set, harvesting methods, and instance data. It remains to be seen whether the ontology and rule set will stabilize over time. Instance data, however, will be continue to be changed via periodic harvesting.

The Adaptive Reasoning Mechanism is designed to select the appropriate reasoning method depending partially on the degree of change. Materialization inference is preferred in situations with infrequent or no updates. Any update of the ontology, custom rules or instances, however, would require re-loading of the data and re-materialization.

Query-invoked inference is preferred, therefore, in situations of rapid changes. As only related rules and data are involved, answers can be returned within an acceptable time period.

 ISBN: 978-1-61208-152-6

C. Knowledge base

Native storage of the knowledge base may help to improve the scalability of ABox inferencing. Native storage will speed up inferencing as well as search since it reducing the time for loading and updating data. Materialized Triples, Inferred Triples after Query-invoked Inference are stored separately from base storage (that including custom rules, ontology and instances) but we create a correlation index.

When we store triples, we group and index them to improve the inferencing performance. For example, triples can be grouped by each property, e.g., "publicationAuthor" or "advisor". Triples with the same property can be stored in the same file. We will use Indexing as a way to store relationship and enhance the search performance.

D. Combination of DL and Rule-based Reasoners

DL reasoners have sufficient performance on complex TBox reasoning, but they do not have scalable query answering capabilities that are necessary in applications with large ABoxes. Rule-based OWL reasoners are based on the implementation of entailment rules in a rule engine. They have limited TBox reasoning completeness because they may not implement each entailment rule or they choose the performance instead of the completeness. Hybrid reasoners that integrate a DL reasoner and a rule engine can combine the strong points of both sides.

ScienceWeb introduces custom rules for description of qualitative and quantitative descriptors. Thus, besides the traditional Tbox and Abox reasonings, we introduce an Rbox reasoner for custom rules inferencing. The DL reasoner is responsible for TBox reasoning while the Rule-based Reasoner is responsible for both Rbox reasoning and ABox reasoning. These three kinds of reasoning are separable in the reasoning system for independent update of ontology, custom rules and instances.

These three kinds of reasoning are not independent. The DL reasoner generates specific entailment rules for the Rule-based Reasoner and ABox reasoning or Rbox reasoning is connected with TBox reasoning. We need to maintain correlations between individual objects.

V. Conclusion and Future Work

Our contributions in this paper are mainly in designing an adaptive reasoning architecture whose scalability and efficiency are able to meet the interaction requirements in ScienceWeb system, and in introducing the preliminary study that we have done for ScienceWeb system. With this architecture we are developing new technologies and an adaptive reasoning system for ScienceWeb that allows users to: (a) get answers effectively in real-time after posting existing qualitative queries, (b) get answers effectively in real-time after changing the ontology, custom rules, or instances and posting qualitative queries, and (c) get answers effectively in real-time when the size of knowledge base scales from thousands to millions.

Because reasoning is required for many large-scale semantic applications, we think it plausible that this architecture could be applied to a variety of different domains.

Sufficient reasoning support is a cornerstone for any realization of ScienceWeb. Our architecture and resulting system will, it is hoped, provide efficient reasoning based on a scalable knowledge base. We have completed the design, have performed a number of formative performance studies using a novel synthetic data generator, and have identified major issues we still need to address as:

Materialization inference - When can incremental inference be invoked? What part of TBox inference and inferred triples are stable for incremental inference?

Query-invoked inference - When can pre-computed TBox inferences be invoked? What pre-computed TBox inferences are? How to optimize query-invoked inference with large size of ABox?

Adaptive mechanism - When can the system switch between materialization inference and query-invoked inference? What are trivial changes and normal changes? Does the system need to differentiate the trivial changes and normal changes?

References

[1] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su, "Arnetminer: Extraction and mining of academic social networks," in *Proceedings of the Fourteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD'2008)*. ACM, 2008, pp. 990–998.

[2] H. Shi, K. Maly, S. Zeil, and M. Zubair, "Comparison of ontology reasoning systems using custom rules," in *International Conference on Web Intelligence, Mining and Semantics*, Sogndal, Norway, 2011.

[3] A. Yaseen, K. J. Maly, S. J. Zeil, and M. Zubair, "Performance evaluation of Oracle semantic technologies with respect to user defined rules," in *10th International Workshop on Web Semantics*, Toulouse, France, 2011.

[4] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, and S. Hellmann, "DBpedia-A crystallization point for the web of data," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 7, no. 3, pp. 154–165, 2009.

[5] Wikipedia, *About Wikipedia — Wikipedia, The Free Encyclopedia*, 2010, http://en.wikipedia.org/wiki/Wikipedia:About [June 18, 2011].

[6] A. Diaz and G. Baldo, "Co-Protégé: A groupware tool for supporting collaborative ontology design with divergence," in *The Eighth International Protégé Conference*, 2005, pp. 32–32.

 ISBN: 978-1-61208-152-6

[7] R. Volz, D. Oberle, S. Staab, and B. Motik, "Kaon server - a semantic web management system," in *Alternate Track Proceedings of the Twelfth International World Wide Web Conference, WWW2003*, Budapest, Hungary, 2003, pp. 20–24.

[8] D. Nardi and R. Brachman, "An introduction to description logics," in *The description logic handbook: theory, implementation, and applications*, 2002, pp. 1–40.

[9] M. Minsky, "A framework for representing knowledge," *The Psychology of Computer Vision*, 1975.

[10] M. Quillian, "Semantic memory," *Semantic Information Processing*, pp. 227–270, 1968.

[11] F. Baader, *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge Univ Pr, 2003.

[12] F. Baader, I. Horrocks, and U. Sattler, "Description logics," *Foundations of Artificial Intelligence*, vol. 3, pp. 135–179, 2008.

[13] G. Stoilos, *An Introduction to Description Logics*, 2005.

[14] E. Mays, R. Dionne, and R. Weida, "K-Rep system overview," *ACM SIGART Bulletin*, vol. 2, no. 3, p. 97, 1991.

[15] Y. Kazakov and B. Motik, "A resolution-based decision procedure for SHOIQ," *Journal of Automated Reasoning*, vol. 40, no. 2, pp. 89–116, 2008.

[16] F. Baader, J. Hladik, C. Lutz, and F. Wolter, "From tableaux to automata for description logics," *Fundamenta Informaticae*, vol. 57, no. 2, pp. 247–279, 2003.

[17] C. Lutz, "Interval-based temporal reasoning with general TBoxes," in *Proc.of the 17th Int. Joint Conf. on Artificial Intelligence (IJCAI 2001)*, 2001, pp. 85–96.

[18] F. Baader and U. Sattler, "An overview of tableau algorithms for description logics," *Studia Logica*, vol. 69, no. 1, pp. 5–40, 2001.

[19] S. Russell and P. Norvig, *Artificial Intelligence: a Modern Approach*. Prentice hall, 2009.

[20] A. Kiryakov, D. Ognyanov, and D. Manov, "OWLIM - a pragmatic semantic repository for OWL," *Web Information Systems Engineering*, vol. 3807/2005, pp. 182–192, 2005.

[21] Ontotext, *OWLIM-OWL Semantic Repository*, 2011, http://www.ontotext.com/owlim/ [June 18, 2011].

[22] Oracle Corporation, *Oracle Database 11g R2*, 2011, http://www.oracledatabase11g.com [June 18, 2011].

[23] J. Zhou, L. Ma, Q. Liu, L. Zhang, Y. Yu, and Y. Pan, "Minerva: A scalable OWL ontology storage and inference system," *The Semantic Web-ASWC 2006*, pp. 429–443, 2006.

[24] J. Heflin and Z. Pan, "DLDB: Extending relational databases to support semantic web queries," in *Proceedings of Workshop on Practical and Scaleable Semantic Web Systems*, 2003, pp. 109–113.

[25] O. Erling, *Advances in Virtuoso RDF Triple Storage (Bitmap Indexing)*, June 2006, http://virtuoso.openlinksw.com-/dataspace/dav/wiki/Main/VOSBitmapIndexing [June 18, 2011].

[26] Y. Chen, J. Ou, Y. Jiang, and X. Meng, "HStar-a semantic repository for large scale OWL documents," *The Semantic Web-ASWC 2006*, pp. 415–428, 2006.

[27] Epimorphics Ltd, *Jena - a Semantic Web Framework for Java*, 2010, http://jena.sourceforge.net/ [June 18, 2011].

[28] Franz Inc, *AllegroGraph RDFStore 4.2.1*, 2011, http://www.franz.com/agraph/allegrograph/ [June 18, 2011].

[29] J. Broekstra, A. Kampman, and F. Van Harmelen, "Sesame: A generic architecture for storing and querying RDF and RDF schema," *The Semantic Web-ISWC 2002*, pp. 54–68, 2002.

[30] B. Motik and U. Sattler, "A comparison of reasoning techniques for querying large description logic aboxes," *Logic for Programming, Artificial Intelligence, and Reasoning*, vol. 4246/2006, pp. 227–241, 2006.

[31] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati, "Data complexity of query answering in description logics," in *Proceedings of the 10th International Conference on the Principles of Knowledge Representation and Reasoning*, 2006, pp. 260–270.

[32] Y. Guo, Z. Pan, and J. Heflin, "LUBM: A benchmark for OWL knowledge base systems," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 3, no. 2-3, pp. 158–182, 2005.

[33] K. Maly, S. Zeil, and M. Zubair, *ScienceWeb pre-survey*, 2010, http://www.cs.odu.edu/∼zeil/scienceWeb/survey2010/ [May 15, 2011].

 ISBN: 978-1-61208-152-6