An Investigation into the Feasibility of the Semantic Web

Zhengxiang Pan, Abir Qasem, and Jeff Heflin
Department of Computer Science and Engineering, Lehigh University
19 Memorial Dr. West, Bethlehem, PA 18015, U.S.A.
{zhp2,abq2,heflin}@cse.lehigh.edu

Abstract

We investigate the challenges that must be addressed for the Semantic Web to become a feasible enterprise. Specifically we focus on the query answering capability of the Semantic Web. We put forward that two key challenges we face are heterogeneity and scalability. We propose a flexible and decentralized framework for addressing the heterogeneity problem and demonstrate that sufficient reasoning is possible over a large dataset by taking advantage of database technologies and making some tradeoff decisions. As a proof of concept, we collect a significant portion of the available Semantic Web data; use our framework to resolve some heterogeneity and reason over the data as one big knowledge base. In addition to demonstrating the feasibility of a "real" Semantic Web, our experiments have provided us with some interesting insights into how it is evolving and the type of queries that can be answered.

Introduction

The Semantic Web envisions a web of ontologies and associated data, which will augment the present Web by adding formal semantics to its documents. By transforming the Web from a collection of documents to a collection of semantically rich data sources, the Semantic Web promises an unprecedented benefit of a global knowledge infrastructure. In addition, the idea of using knowledge representation in a Web friendly manner makes the emerging Semantic Web an ideal test bed for AI applications. However, there are three key questions that have to be addressed before the Semantic Web becomes a feasible enterprise. They are:

1. Who will annotate the data?
2. Without any centralized control how will all this data be connected to one another?
3. Will the existing AI techniques be sufficient to process this huge amount of data?

The first question has received significant attention in recent years. The proliferation of tools and services that can automatically generate Semantic Web data and advances in ontology learning techniques (Maedche & Staab 2001), are indicators that we are making good progress towards overcoming the annotation problem of the Semantic Web. However, the latter two questions in our opinion have not received the attention they deserve.

The second question stems from the inherent autonomous nature of the Web. While this autonomy definitely fuels the Web's phenomenal growth, it also increases the heterogeneity of the information sources. This heterogeneity is equally likely to occur in the Semantic Web. Without a simple (hence easily adaptable) mechanism to address this heterogeneity, it will be difficult to realize the full potential of the Semantic Web.

The third question revolves around the scalability issue of AI systems. At the time of writing of this paper, the number of Resource Description Framework (RDF) documents, cached by Google was about 2.8 million. The Semantic Web search engine Swoogle (Ding *et al.* 2005) currently boasts an index of 850K of Semantic Web documents and more than doubled its index in less than a year. Even in its infancy, Semantic Web data (45 million triples as per Swoogles index of November 2005) is too large for typical reasoning systems. Most of these systems implement the reasoning process in main memory. A size of more than 45M triples places the existing Semantic Web well beyond the reach of many advanced AI systems. Given that the Semantic Web is starting to reach critical mass, we believe the time is right to begin exploring the last two questions.

In this paper we try to do reasoning on this heterogeneous and massive Semantic Web. Specifically we focus on the query answering capability of the Semantic Web. We should note here that there are several other possible applications of the Semantic Web. In recent years the Semantic Web has been touted as an infrastructure for automatic web service composition and autonomous web agents that perform transactions on behalf of a user. However, we believe answering a query by means of logical reasoning lies in the core of original Semantic Web vision. Furthermore, query answering complements the other applications. For example, an agent working for a user will perform better if it has access to richer query answering facilities. To the best of our knowledge this is the first attempt to reason with a substantial amount of authentic Semantic Web data. Specifically we make three contributions. First, we provide a decentralized framework to integrate data sources that use different

vocabularies. Second, we propose a mechanism to realize this framework. And finally we describe a system that uses this mechanism to integrate the existing Semantic Web data and demonstrate that our system is capable of performing scalable reasoning on the real Semantic Web.

The following is the outline of our paper. First, we discuss the heterogeneity problem on the Semantic Web and introduce our framework for addressing this issue. Then we describe a mechanism that allows us to work in our framework. After that we describe our Semantic Web Knowledge Base system. We then describe our experience in working with the real Semantic Web. Specifically we describe the challenges faced in loading, aligning and querying this heterogeneous and massive data set and how we addressed some of them. We finally discuss some related work, make some conclusions and provide some future research direction.

Heterogeneity and the Semantic Web

An initial inspection of the data revealed that many terms with different URIs [1] have identical local names. In fact, this type of synonymy amounts to about 50% of the terms in our data. Among those terms, each local name on average is defined by 4.2 different ontologies. Our initial investigation supports our hypothesis that the Semantic Web is and will be a heterogeneous medium, however a more thorough analysis is still needed.

Another trend in the Semantic Web, is the proliferation of data that commits to simple ontologies like FOAF. FOAF data currently makes the bulk of the Semantic Web documents. The simplicity of the ontology facilitates automatic generation of data and is the main reason for such proliferation. Although a large collection of data that has relatively simple semantics can still make for some very interesting applications (Mika 2005), much more can be achieved if we can somehow add a layer of semantics on the FOAF ontology and then exploit the vast FOAF data with more advanced reasoning.

The trends described above require us to establish alignments between terms of different ontologies. Our approach is to use axioms of OWL, the de facto Semantic Web language, to describe a map for a set of ontologies. The axioms will relate concepts from one ontology to the other. An application can consult this map to retrieve information from sources that may have otherwise appeared to be providing differing information. In our framework, a map is an ontology that imports these ontologies and aligns their terms. It is obvious that we will not have direct mapping between all ontologies, however it should be possible to compose a map from existing maps by traversing through a "semantic connection" of maps. In the sense we now have a web of ontologies as opposed to a web of documents.

We would like to note several points about our mapping framework. First, there is a well-established body of research in the area of automated ontology alignment. This is not our focus. Instead we investigate the application of these alignments to provide an integrated view of the Semantic Web data. Second, in our framework anyone can create a mapping ontology between two or more ontologies, and publish that ontology for use by all. The ontology perspective mechanism that we describe in next section allows a user to select a mapping (or have one selected for them) in order to integrate data sources.

[1] A URI consists of a namespace and a local name.

Ontology Perspectives

Heflin (2001) has observed that there may not be a universal model of all the data sources and ontologies on the Web. In fact, it is extremely unlikely that one could even exist. Instead, we must allow for different viewpoints and contexts, which are supported by different ontologies. He defines perspectives which then allow the same set of data sources to be viewed from different contexts, using different assumptions and background information. A model theoretic description of perspectives is presented in (Heflin & Pan 2004). In this section, we set aside the versioning issues in that paper and show how these simplified perspectives would work for our framework by introducing some essential definitions.

Each perspective will be based on an ontology, hereafter called the basis ontology or base of the perspective. By providing a set of terms and a standard set of axioms, an ontology provides a shared context. Thus, data sources that commit to the same ontology have implicitly agreed to share a context. When it makes sense, we also want to maximize integration by including data sources that commit to different ontologies.

We now provide informal definitions to describe our model of the Semantic Web. A semantic web space $\mathcal{W}$ is two-tuple $\langle \mathcal{O}, R \rangle$, where $\mathcal{O}$ is a set of ontologies and R is a set of resources. We then define an ontology perspective model of a Semantic Web space:

Definition 1 (Ontology Perspective Model) *An interpretation $\mathcal{I}$ is an ontology perspective model of a semantic web space $\mathcal{W} = \langle \mathcal{O}, R \rangle$ based on $O \in \mathcal{O}$ (written $\mathcal{I} \models_O W$) iff:*

1. *$\mathcal{I}$ is a model of O*
2. *for each $r \in R$ such that r commits to O or an ancestor of O, $\mathcal{I}$ is a model of r.*

The definitions of ontology and resource models can be found in (Heflin & Pan 2004). Ontology perspective entailment follows the typical definition of entailment. We use $\mathcal{W} \models_O \phi$ to denote that the ontology perspective of $\mathcal{W}$ based on O entails ϕ.

Theoretically, each of these perspectives represents a single model of the world, and could be considered a knowledge base. Thus, the answer to a semantic web query must be relative to a specific perspective.

Definition 2 *A Semantic Web query is a pair $\langle O_i, \rho \rangle$ where O_i is the base ontology of the perspective and ρ is a formula with existentially quantified variables. An answer to the query $\langle O_i, \rho \rangle$ is θ iff $\mathcal{W} \models_O \theta\rho$ where θ is a substitution for the variables in ρ.*

In order to relate our framework to the perspectives, we now define the minimum mapping ontology between O_i and O_j is O_m, where O_m extends O_i and $O_j$and contains the axioms that map their vocabularies.

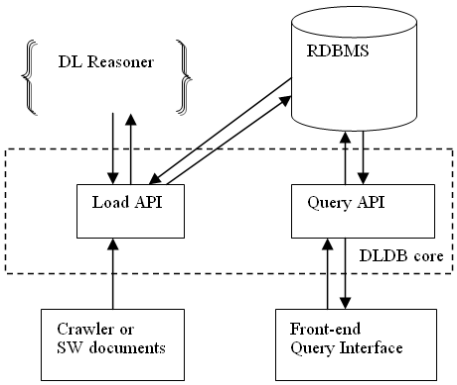

Figure 1: The Architecture of DLDB

Let's take an example to see how our definitions can be applied to the real ontologies. Imagine we have two simple ontologies. O_1 defines concept `Car` and O_2 defines concept `Automobile`. Two data sources commits to them respectively: r_1 states $\{ezz3290 \in O_1 : Car\}$ and r_2 states $\{dfg2134 \in O_2 : Automobile\}$. A map between O_1 and O_2 would be an ontology O_{m12} that imports O_1,O_2 and has the axiom $\{O_1 : Car \Leftrightarrow O_2 : Automobile\}$. We have a query formula $Car(x)$. If the query's perspective is based on either O_1 or O_2, there would be one or zero answers. However, by choosing O_{m12} as the perspective, the query will retrieve both instances.

We argue that our perspectives have at least two advantages over the traditional KR languages. First, the occurrence of inconsistency is reduced compared to the global views, since only a relevant subset of the Semantic Web is involved in processing a query. Even if two ontologies have conflicting axioms, inconsistency would not necessarily happen in most perspectives other than the ones that are based on their common descendants. Second, the integration of information resources is more flexible compared to local views, since any two data sources could be included in the same perspective as long as the ontologies they commit to are both being extended by a third ontology. This third ontology is likely to be the mapping ontology that serves as the perspective.

DLDB: A Semantic Web Query Answering Engine with Perspective Support

DLDB (Pan & Heflin 2003) is a knowledge base system that extends a relational database management system with additional capabilities for partial OWL reasoning. As shown in Figure 1, the components in DLDB are loosely coupled. The DLDB core consists of Load API and Query API implemented in Java. Any DL Implementation Group(DIG) compliant DL reasoner and any SQL compliant RDBMS with JDBC driver can be plugged into DLDB. This flexible architecture maximizes the customizability and allows reasoners and RDBMS run as services or even cluster on multiple machines.

Note we have extended DLDB to support perspectives. The following description is specific to this extended version of DLDB.

The basic table design of DLDB adopted an approach that is similar to the "Schema-aware" representation in (Theoharis, Christophides, & Karvounarakis 2005). According to this approach, creating tables corresponds to the definition of classes or properties in ontology. The classes or properties' ID should serve as the table names. Since each row in the class tables corresponds to an instance of the class, an 'ID' field is needed here to record the ID of the instances. The rest of the data about an instance is recorded using the table per property (aka decompositional) approach. Each instance of a property must have a subject and an object, which together identify the instance itself. Thus the 'subject' and 'object' fields are set in each table for property. Normally, the 'subject' and 'object' fields are foreign keys from the 'ID' field of the class tables that are the domain and range of the property, respectively. However, if the property's range is a declared data type, then the 'object' field is of corresponding datatype. This approach makes it easy to answer some queries with data constraints, such as finding individuals whose ages are under 21. In addition, to shrink the size of database and hence reduce the average query time, DLDB assigns each URI a unique ID number.

In DLDB class hierarchy information is stored through views. The view of a class is defined recursively. It is the union of its table and all of its direct subclasses' views. Hence, a class's view contains the instances that are explicitly typed, as well as those that can be inferred.

Using OWL's constraints for class description, a DL reasoner can compute class subsumption, i.e., the implicit `rdfs:subClassOf` relations. DLDB uses a DL reasoner to precompute subsumption, and then uses the inferred class taxonomy to create class views. Thus, DLDB only consults the DL reasoner once for each new ontology in the knowledge base. Whenever queries are issued concerning the instances of the ontology, the inferred hierarchy information can be automatically utilized. We think that as a query answering engine, DLDB's precomputation improves the computational efficiency which can save time as well as system resources.

Following the definitions of perspectives, every time DLDB loads a new ontology O, it actually sends O with all the ancestors of O to the reasoner for precomputation. Since different perspectives may have different axioms associated with a class / property and hence have different subsumptions returned by reasoner, DLDB also creates views on each ancestor's classes and properties but does not create tables for them. The views are virtual queries, so that the number of views has little impact on the size of the database. Meanwhile, the query based on different perspectives will be translated into corresponding database views.

However, without the separate tables, the instances that commit to different ontologies are indistinguishable. This is problematic as the perspective model does not include the data sources that commit to a descendant ontology. To solve this problem, we add a new field in every class table, namely the 'onto' field. Thus, each instance of a particular class is recorded by its ID, its source document's ID and the

ID of the ontology to which the data source commits. The example tables would be

```
O1:Student(ID, Source, Onto)
O1:takesCourse(Subject, Object, Source,
Onto)
```

Consider that in ontology $O1$ the class `Student` is defined as all the people who take a `Course`, the class `GradStudent` is defined as all the people who take a `GradCourse`, and the class `GradCourse` is a subclass of `Course`. Then, by OWL entailment we can conclude that `GradStudent` is a subclass of `Student`. Thus, if we call the class view creation algorithm after interaction with a DL reasoner, the view of `Student` will be defined as:

```
CREATE O1:Student:viewBy:O1 AS
SELECT * FROM O1:Student WHERE
O1:Student.Onto=O1
UNION SELECT * FROM
O1:UndergraduateStudent:viewBy:O1
UNION SELECT * FROM
O1:GradStudent:viewBy:O1;
```

Since the views are defined recursively, the indirect subsumptions can be automatically taken care of by the database system.

Note `<A owl:equivalentClass B>` is implemented as `<A rdfs:subClassOf B>` plus `<B rdfs:subClassof A>`. For properties, the `rdfs:subProperty` and `owl:equivalentProperty` relations are handled in the same way. In addition, `owl:inverseOf` relations are supported in the property view creation algorithm.

Currently, DLDB is known to be incomplete with respect to transitive properties and class membership realization through restrictions, but supports most other OWL DL features depending on the capability of underlying DL reasoner. Note the queries supported by current DLDB fall under extensional conjunctive queries.

Experiment

The broader goal of our experiment is to gather evidence regarding the feasibility of the Semantic Web. We wanted to see if our framework and the DLDB approach to scalable reasoning at least begin to address the last two of the three questions we posed at the beginning of the paper. The results are encouraging. DLDB has scaled well and the mapping framework with perspectives provided successful integration of disparate data.

The specific DLDB configuration we used in loading and querying the Semantic Web is as follows: the DLDB main program runs on a SUN workstation featuring dual 64-bit Opteron CPUs and 2GB main memory. The backend DBMS is PostgreSQL 8.0 running on the same machine. The DL reasoner is Racer Pro 1.9 running on another Windows powered machine. Both machines are connected via 100 Base-T Ethernet.

We have used Swoogle's 2005 index as our dataset. Of the 364,179 urls that we got from Swoogle, we successfully loaded 343,977 SW documents in 15 days and 15 hours. In total 41,741 among them are identified as ontologies by DLDB. It takes approximately 8 gigabytes disk space for the RDBMS to store the 45M triples. To the best of our knowledge this is largest load of a diverse real world Semantic Web data. This once again validates that DLDB approach scales fairly well.

Since our experiment is a pioneering attempt to treat the Semantic Web as a singular knowledge base and pose queries to it, naturally we did not have access to query patterns or query logs that would allow us to choose "typical" queries. For this experiment we have selected queries which can showcase the possibilities of an integrated Semantic Web using existing data. Note, limitations in available data sometimes placed severe constraints on the kinds of queries we could ask.

From the statistics about the identical local names in the second section, it is obvious that we need to map a large number of concepts with each other to have a fully integrated Semantic Web. However, the size of the Semantic Web dictates that we use some form of automated ontology mapping tool to address this problem. We note, that there are several ongoing research projects in this area (Noy & Musen 2002) but we chose to apply a heuristic described below to select our maps. We do this because in this paper our goal is not to present a comprehensive set of maps but rather to demonstrate the potential for integration in the Semantic Web.

In order to meet our objective, we have adopted the following strategy. We perform a breadth analysis where we try to come up with maps that will loosely align a large number of ontologies. We also perform a depth analysis where we try to come up with maps that will tightly align a few ontologies. In each case we examine the effect of integration on results retuned by various queries.

In breadth analysis we select mapping candidates that will have the highest "integratiblity". We loosely define the "integratiblity" of a concept as its potential to connect multiple ontologies. Specifically, in this experiment, we have considered integratiblity of a concept to be the frequency of its occurrence in different ontologies. We use a combination of manual inspection and semi automatic filtering to determine concepts with high integratiblity. The main idea behind this heuristic is that these concepts will give us the most integration with least amount of effort. We mapped concepts that have the highest frequency of occurrence, shown in the table below. Note, here we only considered the concepts that have instance data:

Concept	Frequency of occurrence
Person	10
Country	10
Project	7
City	5
University	4

We note two aspects of the above findings. First, a more careful analysis must be done to come up with a better integratiblity measure. For example, we need to identify synonyms or sub class / super class of a concept and identify those in different ontologies. Although they may not weigh as much as a direct match, they contribute to determining integratability. Second, we have noticed that there are several concepts that have an integratiblity of 3 or below. We do not

list them here for space constraints.

We select `Person` from the above list in order to investigate the effect of maps from our breadth analysis. We perform an experiment as follows:

We first query the knowledge base from the perspective of each of the 10 ontologies that define the concept `Person`. We now ask for all the instances of the concept `Person`. The results vary from 4 to 1,163,628. We then map the `Person` concept from all the ontologies to the `Person` concept defined in the FOAF ontology. We now issue the same query from the perspective of this map and we get 1,213,246 results. The results now encompass all the data sources that commit to these 10 ontologies. Note: a pair wise mapping would have taken 45 mapping axioms to establish this alignment instead of the 9 mapping axioms that we used. More importantly due to this network effect of the maps, by contributing just a single map, one will "automatically" get the benefit of all the data that is available in the network. Although in our experiment we have created this "super map" manually, it is conceivable that this can be automated. The response time for this query is about a minute. Although this is a very high number for an interactive system, we note that it is unlikely a user will ask a query that has such a low selectivity. In addition, our investigation suggests that this number may be high due to inefficiency in handling large result set found in the JDBC driver implementation we used.

In addition to developing the breadth analysis based maps, we also perform a depth analysis of some selected ontologies and mapped their concepts and roles in complete details.

The table below describes the prefixes, abbreviations and urls of the ontologies we use in our in depth map experiment.

Prefix	Abb.	url
foaf	O_f	http://xmlns.com/foaf/0.1/
aktors	O_a	http://www.aktors.org/ontology/portal
hack	O_h	http://www.hackcraft.net/bookrdf/vocab/0_1/
swrc	O_s	http://swrc.ontoware.org/ontology
dc	O_d	http://purl.org/dc/elements/1.1/

Our experiment considers maps of 1,500 concepts that span over 1,200 ontologies. As we have described through out this section some of these maps have been created by us specifically for this experiment. The rest of the maps are from ontologies that define their concepts in terms of ontologies they extend.

We describe the ρ formulas (see definition 2) of the queries below in FOL:

1. $\exists$ X,Y,Z, O_f :Person(X) $\wedge$ O_f :Document(Y) $\wedge$ O_f :maker(Z) $\wedge$ O_f :surname(X, ''Williams'')
2. $\exists$ X, O_d :Document(''rdf-sparql-query'') $\wedge$ O_f :isReferencedBy(''rdf-sparql-query'', X)
3. $\exists$ X, O_a :Activity(X)

In the first query we show how maps can dramatically improve retrieval in the Semantic Web. The query is about a list of publications by a certain individual using the FOAF ontology. There are several other ontologies that define these concepts and they have varying number of data that commit to them. However, a query posed in terms of FOAF ontology will miss the data that commits to the others. We can however map several ontologies with a mapping ontology and use that perspective to query the integrated view. We have mapped the relevant vocabularies of the following ontologies:(described using Description Logic)

`foaf:Person` $\equiv$ `swrc:Person` $\equiv$ `aktors:Person`
`swrc:Publication` $\sqsubseteq$ `foaf:Document`
`hack:Book` $\sqsubseteq$ `foaf:Document`
`aktors:Publication` $\sqsubseteq$ `foaf:Document`
`aktors:has-author` $\sqsubseteq$ `foaf:maker`

In the second query, we show that our flexible mapping framework can be used to add additional semantics to an existing ontology. For example, Dublin core(O_d) has two RDF properties: `isReferencedBy` and `references`. It is fairly obvious that they have an inverse relationship. However, RDF does not allow us to express this. We can create a map:

`dc:references`$^-$ $\equiv$ `dc:isRefenecedBy.`

The third query demonstrates the fact that reasoning across ontologies can discover implicit maps. In addition, unlike the first two queries, this one has low selectivity since it places no additional constraints on the individuals. In this query, we ask for a list of activities in terms of Aktors ontology. If our mapping ontology is as follows:

`swrc:Project` $\equiv \exists$ `swrc:isAbout.swrc:ResearchArea`
`aktors:Activity` $\equiv \exists$
`aktors:addresses-generic-area-of-interest`
`.aktors:Generic-Area-Of-Interest`
`swrc:ResearchProject` $\sqsubseteq$ `aktors:Generic-Area-Of-Interest`
`swrc:isAbout` $\sqsubseteq$ `aktors:addresses-generic-area-of-interest`

The reasoner will be able to infer `swrc:Project` to be a sub class of `aktors:Activity`. As a result, our query will return all the information about activities using aktors ontology and projects using swrc ontology as well.

The table below summarizes the results of the queries. It shows that by adding the mapping ontologies and using them as query perspectives, we retrieve more results with a slight increase in query time. Also, most of the queries were under 1 second and the longest query was in 5 seconds in our experiment. Considering the size and reasoning involved in the system, we believe our approach is fairly scalable given the number of maps used in the experiment.

Query formula	Perspective	# of Results	time(ms)
1	O_s	7	3964
1	map of O_f, O_h, O_s, O_a	365	5132
2	O_d	0	5
2	map on O_d	7	22
3	O_a	196	10
3	map of O_a and O_s	371	31

Related work

We claim that this is the first attempt to reason with the Semantic Web as a whole. Here we use reasoning strictly as a knowledge representation term. We should however note that there are several projects that process the Semantic Web in various other ways. For example, Swoogle is the largest index of Semantic Web documents. However, Swoogle's query and retrieval mechanism is basically an IR system. The mechanism relies on string match and relevancy computation. This does not exploit the reasoning that can be

done over Semantic Web data.

Flink (Mika 2005) is a presentation of the scientific work and social connectivity of Semantic Web researchers. This interesting application uses FOAF to develop social maps.

The Carnot (Huhns *et al.* 1993) system is one the earliest work that uses articulation (i.e. mapping) axioms to address heterogeneity in enterprise integration. Carnot depends on commonsense knowledge and is more of a centralized approach which is not a suitable model for the Web.

There are some on going efforts to bootstrap the Semantic Web by providing ontologies and reusable knowledge bases. TAP (Guha 2002) is an example of one such effort. They provide a fairly comprehensive source of basic information about popular entities, like music, authors, autos etc. They do not replace the upper ontologies but complement them. Their work can be seen as "grounding" of some fragment of those upper ontologies with real world data.

The C-OWL work (Bouquet *et al.* 2003) proposed that ontologies could be contextualized to represent local models from a view of a local domain. They suggested that each ontology is an independent local model. If some other ontologies' vocabularies need to be shared, some bridge rules should be appended to the ontology which extends those vocabularies. Compared to C-OWL, our perspective approach also provides multiple models from different views without modifying the current Semantic Web languages.

In the past few years there has been a growing interest in the development of systems that will store and process large amount of Semantic Web data. The general design approach of these systems is similar to ours, in the sense that they all use some database systems to gain scalability while supporting as much inference as possible by processing and storing entailments. However, most of these systems emphasize RDF and RDF(S) data at the expense of OWL reasoning. The system that most resembles the capabilities of DLDB is KAON2 (Hustadt, Motik, & Sattler 2004), which uses a novel algorithm to reduce OWL DL into disjunctive datalog programs.

Conclusion and future work

We present an integration framework for dealing with heterogeneity in the Semantic Web. The framework does not require any additional language primitives beyond OWL. We present a mechanism to implement this framework. This ontology perspective mechanism gives the user the flexibility to choose the type of integration (s)he desires (or believes in). We have demonstrated that it is feasible to perform essential OWL reasoning with very large (45M triples), authentic Semantic Web data. Our system is optimized for working with OWL instances. We put forward that scalability is more critical in processing the data sources as opposed to ontologies, because data sources will substantially outnumber the ontologies in the Semantic Web.

Although we believe our work is a first step in the right direction, we have discovered many issues that remain unsolved. First, although our system scales well to the current size of the Semantic Web, it is still unknown if such techniques will continue to scale well as the Semantic Web grows. Second, although perspectives resolve some form of inconsistencies; inconsistency between data sources that commit to the same ontology still result in inconsistent perspectives. These are some interesting research questions that we plan to explore and invite others to do the same.

Acknowledgment

This material is based upon work supported by the National Science Foundation (NSF) under Grant No. IIS-0346963. We sincerely thank Tim Finnin of UMBC for providing us access to the Swoogle's index of URLs.

References

Bouquet, P.; Giunchiglia, F.; van Harmelen, F.; Serafini, L.; and Stuckenschmidt, H. 2003. C-OWL: Contextualizing ontologies. In *Proc. of the 2003 Int'l Semantic Web Conf. (ISWC 2003)*, LNCS 2870, 164–179. Springer.

Ding, L.; Finin, T.; Joshi, A.; Peng, Y.; Pan, R.; and Reddivari, P. 2005. Search on the semantic web. *IEEE Computer* 10(38):62–69.

Guha, R. 2002. Tap: Towards the semantic web. Demo on World Wide Web 2002 Conference. At: http://tap.stanford.edu/www2002.ppt.

Heflin, J., and Pan, Z. 2004. A model theoretic semantics for ontology versioning. In *Proc. of the 3rd International Semantic Web Conference*, 62–76.

Heflin, J. 2001. *Towards the Semantic Web: Knowledge Representation in a Dynamic, Distributed Environment*. Ph.D. Dissertation, University of Maryland.

Huhns, M.; Jacobs, N.; Ksiezyk, T.; Shen, W. M.; Singh, M.; and Canata, P. 1993. Enterprise information modeling and model integration in carnot. In *Proc. of the International Conference on Intelligent and Cooperative Information Systems*, 12–14.

Hustadt, U.; Motik, B.; and Sattler, U. 2004. Reducing shiq description logic to disjunctive datalog programs. In *Proc. of the 9th International Conference on Knowledge Representation and Reasoning*, 152–162.

Maedche, A., and Staab, S. 2001. Learning ontologies for the semantic web. *IEEE Intelligent Systems* 16(2):72 – 79.

Mika, P. 2005. Flink: Semantic web technology for the extraction and analysis of social networks. *Journal of Web Semantics* 3(2).

Noy, F., and Musen, A. 2002. Evaluating ontology-mapping tools: Requirements and experience. In *The Proceedings of OntoWeb-SIG3 Workshop at the 13th International Conference on Knowledge Engineering and Knowledge Management*.

Pan, Z., and Heflin, J. 2003. DLDB: Extending relational databases to support semantic web queries. In *Proc. of the Workshop on Practical and Scaleable Semantic Web Systms, ISWC*, 109–113.

Theoharis, Y.; Christophides, V.; and Karvounarakis, G. 2005. Benchmarking database representations of rdf/s stores. In *Proc. of the 4th International Semantic Web Conference*, 685–701.