Sep, 2005 TR CS-05-09

Search on the Semantic Web

Li Ding, Tim Finin, Anupam Joshi, Yun Peng,
Rong Pan, Pavan Reddivari

Department of Computer Science and Electrical Engineering
University of Maryland, Baltimore County
Baltimore MD 21250

Search on the Semantic Web

Li Ding, Tim Finin, Anupam Joshi, Yun Peng,
Rong Pan, Pavan Reddivari

University of Maryland, Baltimore County
Baltimore MD 21250 USA

Abstract

The Semantic Web provides a way to encode information and knowledge on web pages in a form that is easier for computers to understand and process. This article discusses the issues underlying the discovery, indexing and search over web documents that contain semantic web markup. Unlike conventional Web search engines, which use information retrieval techniques designed for documents of unstructured text, Semantic Web search engines must handle documents comprised of semi-structured data. Moreover, the meaning of data is defined by associated ontologies that are also encoded as semantic web documents whose processing may require significant amount of reasoning. We describe Swoogle, an implemented semantic web search engine that discovers, analyzes, and indexes knowledge encoded in semantic web documents throughout the Web, and illustrate its use to help human users and software agents find relevant knowledge.

1 Introduction

As the scale and the impact of the World Wide Web has grown, search engines have assumed a central role in the Web's infrastructure. In the earliest days of the Web, people found pages of interest by navigating (quickly dubbed *surfing*) from pages whose locations they remembered or bookmarked. Rapid growth in the number of pages gave rise to web directories like Yahoo that manually organized web pages into a taxonomy of topics. As growth continued, these were augmented by search engines such as Lycos, HotBot and AltaVista, which automatically discovered new and modified web pages, added them to databases and indexed them by their keywords and features. Today, the search engines such as Google and Yahoo dominate the Web's infrastructure and largely define our Web experience.

Most knowledge on the Web is presented as natural language text with occasional pictures and graphics. This is convenient for human users to read and view but difficult for computers to understand. This limits the indexing capabilities of state-of-the-art search engines, since they can't infer meaning

– that a page is referring to a bird called Raven or the sports team with the same name is not evident to them. Thus users share a significant burden in terms of constructing the search query intelligently. Even with increased use of XML-encoded information, computers still need to process the tags and literal symbols using application dependent semantics. The Semantic Web offers an approach in which knowledge can be published by and shared among computers using symbols with a well defined, machine-interpretable semantics [4].

Search on the Semantic Web differs from conventional web search for several reasons. We describe the sources of these differences which will manifest themselves in the design of Swoogle [13], the first search engine for the Semantic Web that we have created.

First, Semantic Web content is intended to be published by machines for machines, e.g., tools, web services, software agents, information systems, etc. Semantic Web annotations and markup may well be used to help people find human-readable documents, but there will likely be a layer of "agents" between human users and Semantic Web search engines.

Second, knowledge encoded in semantic web languages such as RDF differs from both the largely unstructured free text found on most web pages and the highly structured information found in databases. Such semi-structured information requires a combination of techniques for effective indexing and retrieval. RDF and OWL introduce aspects beyond those for ordinary XML, allowing one to define terms (i.e., classes and properties), express relationships among them, and assert constraints and axioms that hold for well-formed data.

Third, Semantic Web documents can be a mixture of concrete facts, classes and property definitions, logic constraints and metadata, even within a single document. Fully understanding the document can require substantial reasoning, so search engines will have to face the design issue of how much reasoning to do and when to do it. This reasoning produces additional facts, constraints and metadata which may also need to be indexed, potentially along with the supporting justifications. Conventional search engines do not try to understand document content because the task is just too difficult and requires more research on text understanding.

Finally, the graph structure formed by a collection of Semantic Web documents differs in significant ways from the structure that emerges from the a collection of HTML documents. This will influence effective strategies to automatically discover Semantic Web documents as well as appropriate metrics for ranking their importance.

The remainder of this article discusses how search engines can be adapted to the Semantic Web and describes Swoogle, an implemented metadata and search engine for online Semantic Web documents. Swoogle analyzes these documents and their constituent parts (e.g., terms and triples) and records meaningful metadata about them. Swoogle provides web-scale semantic web data access service, which helps human users and software systems to find relevant documents, terms and sub-graphs, via its search and navigation services.

2 Background: The Semantic Web

The Semantic Web is a framework that allows data and knowledge to be published, shared and reused on the Web and across application, enterprise, and community boundaries. It is a collaborative effort led by World Wide Web Consortium based on a layered set of standards, as shown in Figure 1. The bottom two layers provide a foundation, using XML for syntax and URIs for naming. The middle three layers provide a representation for concepts, properties and individuals based on the Resource Description Framework (RDF) [23], RDF Schema (RDFS) [5] and the Web Ontology Language (OWL) [12]. The top most layers, still under development, extend the semantics to represent inference rules [21], proofs [10] and trust.

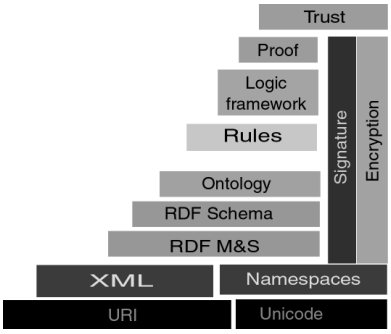

Figure 1: Tim Berners-Lee's layer cake of enabling Semantic Web standards and technologies. (adapted from http://www.w3.org/2002/Talks/04-sweb/slide12-0.html)

The Semantic Web is materialized by Semantic Web Documents (SWDs) typically published as Web pages encoded in XML or one of several other encodings. Figure 2 shows a very simple SWD encoded using the **RDF/XML** syntax [2]. Line 1 declares the document to be an XML document. Lines 2-4 further defines the content to be an RDF document and provide abbreviations for three common "namespaces" for RDF, OWL and FOAF[1].

The SWDs vocabulary consists of **literals** ('Li Ding' at Line 6), **URI-based resources** (`mailto:dingli1@umbc.edu` at Line 7), and **anonymous resources** defined by Lines 5-9. Users assert statements using **RDF triples** such as the triple at Line 5 which has an anonymous resource as the subject, rdf:type as the predicate and foaf:Person as the object. A higher level of granularity is **class-instance**, which is offered by RDFS's object oriented ontology constructs. Lines 5-9 assert that "there is an instance of a foaf:Person having foaf:name 'Li Ding', foaf:mbox `mailto:dingli1@umbc.edu`, and this instance is owl:sameAs another instance identified by `http://www.csee.umbc.edu/~dingli1/foaf.rdf#dingli` ". A visualization of the **RDF graph** is shown in Figure 3.

[1]FOAF, standing for "Friend of a Friend", defines classes and properties for describing people, their common attributes, and relations among them.

```
 1: <?xml version="1.0" encoding="utf-8"?>
 2: <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 3:          xmlns:owl="http://www.w3.org/2002/07/owl#"
 4:          xmlns:foaf="http://xmlns.com/foaf/0.1/" >
 5:   <foaf:Person>
 6:      <foaf:name>Li Ding</foaf:name>
 7:      <foaf:mbox rdf:resource="mailto:dingli1@umbc.edu"/>
 8:      <owl:sameAs rdf:resource="http://www.csee.umbc.edu/~dingli1/foaf.rdf#dingli"/>
 9:    </foaf:Person>
10: </rdf:RDF>
```

Figure 2: An example Semantic Web document written in RDF/XML. The SWD is available at http://ebiquity.umbc.edu/get/a/resource/134.rdf.

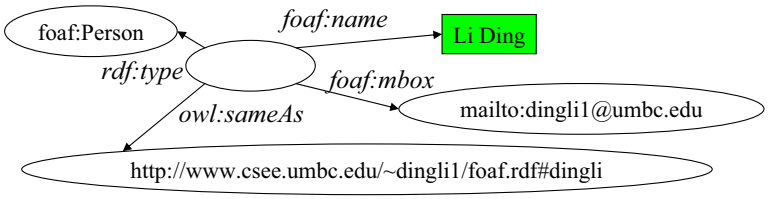

Figure 3: The RDF Graph of the instance of *foaf:Person* from Figure 2.

The Semantic Web can be thought of as a collection of loosely federated databases on the Web. It offers physical independence by separating physical Web storage (enforced by online SWDs) from the logical representation (enforced by the RDF graph model). In this view, the Semantic Web represents a large, universal RDF graph whose parts are physically serialized by SWDs distributed across the Web. However, the formal semantics associated with Semantic Web languages support generating new facts from existing one, while conventional databases only enumerate all facts.

The RDFS and OWL layers support viewing the Semantic Web support as a large knowledge base distributed across the Web. The model theoretic formal semantics [20, 27] for RDFS and OWL are less expressive than many commonly used knowledge representation formalisms. Current research on RuleML and SWRL are attempts to support the top four layers of Figure 1. This will provide a natural mechanism for representing, for example, policy rules governing security and privacy constraints [22].

Today's Semantic Web is firmly grounded in standards and supports a number of well articulated use cases. These standards require RDF content to exist as separate documents (SWDs) that refer to other web resources (e.g., HTML documents, images, web services) using URIs to make assertions about them. Given it's aspiration to be the ultimate data and knowledge sharing framework, the Semantic Web is expected to evolve and change, and Semantic Web search engines will have to change accordingly.

One expected set of extensions is standards for embedding RDF content in various data formats. The W3C is currently working on a new standard to allow

RDF content to be embedding in XHTML documents so that text and semantic markup can co-exist. Adobe has adopted RDF as an extensible way to embed metadata in images, PDF documents and other data formats.

Another direction for growth is the use of RDF outside of documents on the Web. RDF has been wide used to encode metadata, such as Digital Library (Dublin Core) and P2P system (Edutella [25]). Many are using web services to publish SWDs dynamically generated from underlining knowledge bases or databases [18, 6]. RDF is also being used to carry content in agent communication [32] and in pervasive computing [9].

3 Searching the Semantic Web

Search engines for both the conventional Web and the Semantic Web involve the same set of high level tasks: discovering or revisiting online documents, processing users' queries, and ordering search results. In subsequent three sections we will describe how our system, Swoogle, has addressed the tasks in particular. When considering these tasks, two facts should be kept in mind. First, we are processing Semantic Web documents which are distinct from regular HTML documents and there are far fewer of them. Second, on the Semantic Web, search clients are more likely to be software agents than people.

3.1 Discovering and revisiting documents

Conventional search engines scan all possible IP addresses and/or employ crawlers to discover new web documents. A typical crawler starts from a set of seed URLs, visits documents, and traverses the Web by following the hyperlinks found in visited documents. The fact that the Web forms a well connected graph and the ability for people to manually submit new URLs make this an effective process.

A Semantic Web crawler must deal with several problems. SWDs are needles in the haystack of the Web, so an exhaustive crawl of the Web is not an efficient approach. Moreover, the graph of SWDs is not (yet) as dense and well-connected as the graph formed by conventional web pages. Finally, many of the URLs found in a SWD point to documents which are not SWDs. Following these can be computationally expensive, so heuristics to limit and prune candidate links are beneficial.

A Semantic Web crawler can also use conventional search engines to discover initial seed SWDs from which to crawl. Swoogle, for example, uses Google to find likely initial candidate documents based on their file names, e.g., searching for documents whose file names end in *.rdf*, or *.owl*. It also actively prunes links that are unlikely to contain semantic markups.

For the most part, issues of how often to revisit documents to monitor for changes is the same for both the conventional Web and the Semantic Web. However, modifying a SWD can have far reaching and non-local effects if any class or property definitions used by other documents are changed. Depending on

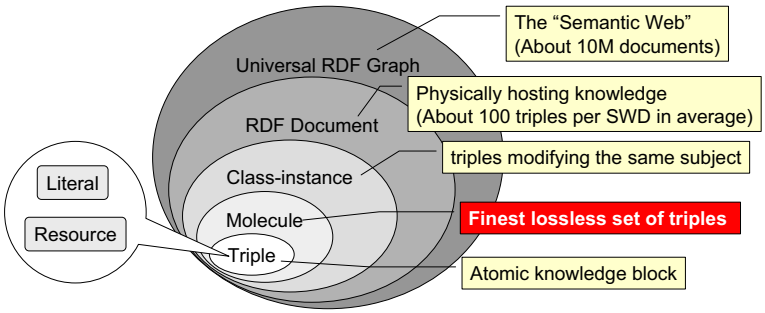

Figure 4: The Semantic Web can be viewed at different levels of granularity, from the universal graph comprising all RDF data on the Web to individuals triples and their constituent resources and literals.

the nature and amount of reasoning that is done when documents are analyzed and indexed, updating a SWD can trigger significant work for a Semantic Web search engine.

3.2 Query Processing

The core task of a search engine is processing queries against the data it has indexed. This can be broken down into three issues: what should be returned as query results, over what data should the queries be run, and what constraints can be used in a query.

As shown in Figure 4, Semantic Web data can be aggregated at several levels of granularity, ranging from the universal graph of all RDF data on the Web to a single RDF triple and the term URIs it comprises. Since search engines usually return the references (or location) of search results, our work has identified three types of output at different levels.

- **The term URI.** At the lowest level is a URI representing a single RDF term – a class, property or instance. For Semantic Web content, these terms are analogous to words in natural language. Knowing the appropriate terms used to describe a domain is an essential requirement for constructing Semantic Web queries.

- **An RDF Graph.** In order to access knowledge in the Semantic Web, users need to fetch an arbitrary sub-graph from a target RDF graph. The sub-graph might correspond to a named graph [8], a collection of triples with a common subject, or an RDF molecule [14].

- **The URL of a Semantic Web Document.** This corresponds to the result returned by a conventional Web search engine – a reference to the physical document that serializes an RDF graph. This level of granularity

helps improve efficiency in filtering out huge amount of irrelevant knowledge. Some documents, such as those representing consensus ontologies, are indented to be shared and reused. Discovering them is essential to the workings of the Semantic Web community.

In order to search the RDF graph, all triples need to be stored. This is essentially the basic features of an RDF database (i.e., triple store), and is beyond the scope of this paper. The first and third output requirements are similar to dictionary lookup and web search respectively; and the prohibitively space cost for storing all triples can be avoid by utilizing compact metadata model.

As for a term, the following metadata needs to be considered: the namespace and local-name extracted from the term's URI, the literal description of the term, the type of a term, in/out degree of the corresponding RDF node, and the binary relation among terms, namespace and SWDs.

For a semantic web document, metadata about itself (such as document URL and last-modified time) and its content (such as terms being defined or populated, and ontology documents being imported) should be considered. One interesting case is searching the provenance of RDF graph that searches for SWDs that imply the given RDF graph in whole or in part. It stores every triples in all indexed SWDs and has the same scalability issue as RDF database.

The structured metadata provides greater freedom in defining matching constraints: users can specify 2D constraints in (property, value) format such as (hasLocalName, 'Person'). Note that the possible values of 'property' are predetermined by the schema of metadata, but the possible values of 'value' are undetermined since metadata is accumulated continuously.

3.3 Ranking

Google was the first search engine to order its search results based in part on the "popularity" of a web page as computed from the Web's graph structure. This idea has turned out to be enormously useful in practice and is equally applicable to Semantic Web search engines. However, Google's PageRank [26] cannot be directly used in the Semantic Web for several reasons – some links connect a document to the ontologies to be imported to interpret it, some reference terms defined in yet other ontologies, some reference semantic web instances and other links point to normal web resources. An appropriate ranking algorithm for the Semantic Web should treat each of these links in a different manner.

4 Swoogle Semantic Web Discovery

4.1 Discovery mechanisms

Rather than using one uniform crawling technique to discover Semantic Web Documents, Swoogle employs a four-fold strategy: (i) running meta-searches on conventional web search engines, such as Google, to find candidates; (ii) using

a focused web crawler to traverse directories in which SWDs have been found; (iii) invoking a custom Semantic Web crawler on discovered SWDs; and (iv) collecting URLs of SWDs and directories containing SWDs submitted by users.

4.1.1 Searching Google for SWDs

Modern comprehensive search engines have done a thorough job of discovering and indexing documents on the web, including a large number Semantic Web documents. Swoogle currently uses Google to find initial "seed" documents that are likely to be SWDs, although other comprehensive search engines can be used as well. In addition to having a large index, Google exposes an API and allows one to constrain a search to documents from a given domain (e.g., *umbc.edu* and of a particular file type (e.g., those ending in *.rdf*. We will discuss how Swoogle uses each of these query constraints to find good candidate SWDs.

filetype query. Since some special file extensions such as '.rdf' are widely used by many SWDs, Google's *filetype* search can be used. Swoogle dynamically selects candidates from popular SWD extensions to run such query. In Swoogle, an extension is called a 'candidate' if it has been used by more than ten SWDs and has at least 50% accuracy in classifying SWDs. Table 1 lists all candidate SWD extensions and a potential one ('xml') that is not yet a candidate due to its low precision.

This data was derived from a dataset *DS-JULY* that was collected by Swoogle as of July 2005. DS-JULY has about 500,000 labeled web pages, 79% of which are SWDs and 64% of which have file extensions. *Recall* is the percentage of dataset SWDs that use the given extension and *Precision* is the percentage of dataset SWDs that use the extension. Most candidate SWD extensions have high precision, but only 'rdf', 'owl' and 'rss' have significant recall. Around 30% SWDs do not use any of these extensions.

extension	# SWDs	Precision	Recall
rdf	207385	93.39%	50.82%
owl	58862	85.41%	14.42%
rss	16328	85.09%	4.00%
n3	2022	55.49%	0.50%
daml	1305	91.51%	0.32%
foaf	915	98.60%	0.22%
nt	826	83.10%	0.20%
xrdf	549	98.92%	0.13%
rdfs	355	89.42%	0.09%
out	125	69.44%	0.03%
owl~	25	100.00%	0.01%
xml	3542	37.25%	0.87%

Table 1: Swoogle uses these eleven file types to query Google for documents that are 'candidate' SWDs.

site query. Using these file types to find candidate SWDs works well, but

Google returns at most 1000 results for any query. In order to get more than 1000 documents with a given filetype (e.g., *.owl*) we take advantage of Google's ability to restrict a search to results from a specified domain or site. After filtering out the non SWDs from the results, we extract a list of the sites from which they have come. For each new site we encounter, we query again but restrict the search to that site.

Site queries work because of the locality hypothesis – a website with one SWD is more likely to have more. An example query string is 'site:ws.audioscrobbler.com RDF or FOAF' (the keywords 'RDF or FOAF' are used to exclude irrelevant web pages in meta search). An important part of Swoogle's database is the list of sites where we've found at least one SWD and the number of SWDs discovered on that site to date. The website to be explored is dynamically determined based on the number of SWDs discovered from it.

In practice, both types of Google queries contributed similar amount of URLs. In addition, since Google also updates its index, running the same query weekly can result in different sets of URLs. Table 2 shows the top five Google queries and the number of URLs they yield from the DS-JULY dataset. The number of contributed SWDs is relatively low because many URLs found by Google have been already found by other discovery mechanisms. For instance, there are 58862 SWDs with the extension '.owl' but only 1777 of them were discovered through Google.

	Google Query	# of SWDs contributed	Google Estimated
1	rdf OR foaf site:ws.audioscrobbler.com	4,245	11,700
2	rdf OR foaf site:blog.drecom.jp	2,800	61,500
3	rdf OR foaf site:yaplog.jp	2,737	85,600
4	rdf OR foaf site:bitzi.com	2,654	17,100
5	rdf+filetype:rdf +xml -version -jp +tw	2,532	1,420
6	rdf+filetype:rdf +xml -version +jp	2,103	43,700
7	rdf OR foaf site:bulkfeeds.net	2,051	674
8	rdf+filetype:rdf +xml iso-8859	1,931	186
9	rdf+filetype:owl	1,777	1,460
10	rdf OR foaf site:blogs.dion.ne.jp	1,703	6,890

Table 2: Swoogle uses Google's site search to find additional SWDs. These are the ten most productive queries for the DS-JULY dataset.

4.1.2 Web Directory Crawler

Once an SWD has been discovered, it's likely that there are more to be found in the same directory. Swoogle uses a simple focused crawler to explore the web environment around discovered SWDs and find more.

A web page P is under a **web directory** W if W's URL is the prefix of P's URL. A web directory crawler is a bounded web crawler; it traverses all

web pages under a given web directory or directly linked by such web pages by following hyperlinks. By exhaustively scanning a web directory, such as inference web proof space[2] Swoogle can often find more SWDs than Google's estimate. By visiting the directly linked web pages, it can handle *hubs*, such as DAML Ontology Library[3] which links to SWDs stored in different websites. Swoogle's Web directory crawler complements Google site query since even the best web search engine index a fraction of the pages on the Web. Swoogle also accepts Web directories suggested by users and regularly visits the directories of some well known SWD repositories.

4.1.3 Semantic Web Crawler

Swooglebot is one of many Semantic Web crawlers (also known as scutters[4]. Unlike the web directory crawlers which process all web pages at HTML level, semantic web crawlers only process SWDs. Scutters follows links selectively using some popular heuristics: (i) namespace of a URIref that links from a resource reference to a resource definition, (ii) URLs of the instances of *owl:Ontology*, and (iii) URL of resource in special triples, such as triples using *rdfs:seeAlso* as widely used in surfing FOAF personal profiles.

In practice, three issues must be considered. First, the URL of a namespace can be redirected, e.g. the namespace URL of the Dublin Core Element ontology, `http://purl.org/dc/elements/1.1/`, redirects to its real physical URL, `http://dublincore.org/2003/03/24/dces#`. A Semantic Web crawler must capture redirection and use it as an addition heuristic to discover URLs of SWDs. Second, a Semantic Web crawler should process RDF content that is embedded or linked in a document of another type, such as an HTML or XHTML document. Usually, the first encountered block of RDF graph encoded by RDF/XML is processed[5]. Third, all URIrefs in SWDs could potentially link to other sources; hence Swoogle use extensions to filter URIrefs after applying the content analysis heuristics.

4.1.4 Users' Submissions and RDF Sitemap

Users' submission complements automated methods for discovering URLs of SWDs. Swoogle provides a web based form to collect these submissions. So far Swoogle has collected a few hundred manual submissions and over 12,000 automatic submissions[6]. These submitted URLs are good starting points for running web directory crawling.

Since the Web serves mainly human users and most content in a website are not SWDs, traversing a website only for SWDs may be unwise. In order to avoid the traffic introduced by exhaustive crawling, a website can publish a RDF Sitemap "sitemap.rdf" which enumerates the URLs of all SWDs within

[2] `http://iw.standford.edu/proofs`
[3] `http://ontologies.daml.org/`
[4] A specification of a scutter is available at `http://rdfweb.org/topic/ScutterSpec`
[5] Standards for embedding RDF are being developed by the W3C at the time of this writing.
[6] Some of which are spam!

the website. Such a RDF Sitemap format can also be used to publish Swoogle's site search result (see `http://swoogle.umbc.edu/site.php`).

4.2 Discovery Results

4.2.1 Performance of Discovery Mechanisms

Figure 5 compares the discovery performance of the above methods. The 'Semantic Web crawler' entry refers to URLs obtained by the Swooglebot crawler. The large number of SWDs and to-crawl URLs result from websites hosting vast numbers of interlinked FOAF documents. The 'google query' entry refers to URLs obtained by sending at most 1000 queries to Google through its web API. The 50% accuracy demonstrates the effectiveness of automatically generated Google queries. The 'WDC+user submit' is the result of crawling user submitted URLs using web directory crawler. Its extremely high accuracy is the result of only adding URLs of validated SWDs in web directory crawling to Swoogle's URL list. Note that all these URLs are different from the 350,000 URLs inherited from prior version of Swoogle.

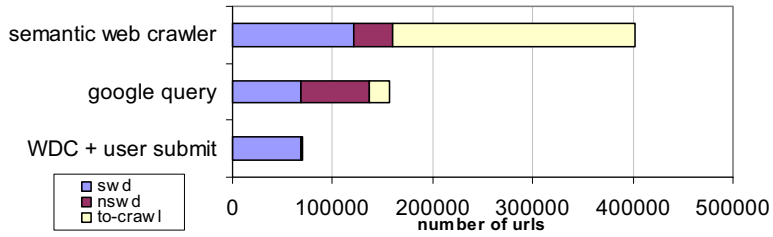

Figure 5: A comparison of the discovery approaches as measured by the number of collected SWDs, non-SWDs and URLs yet to be crawled.

4.2.2 The Size and Growth of the Semantic Web

Semantic Web content exists in many forms – public RDF web files, embedded in PDF documents, JPG images and spreadsheets, as strings databases fields, in messages passed among software agents, and as broadcast data in pervasive computing environments. Our focus is studying the use of Semantic Web data in its "traditional" form, as public web pages encoded in XML/RDF or one of its common variants, so we will use this to comment on the status of Semantic Web.

In 2002 Eberhard [15] reported 1,479 SWDs with about 255,000 triples out of nearly $3x10^6$ web pages. As of July 2005, Swoogle has found over $5x10^5$ SWDs with more than $7x10^7$ triples. Although this number is far less than Google's eight billion web pages, it represents a non-trivial collection of Semantic Web data [19].

Figure 6 plots a Power Law distribution of last modified time of SWDs ('swd' curve) which demonstrates that the Semantic Web is experiencing a

rapid growth rate or at the very least being actively maintained[7]. The apparent growth in the number of ontology documents ('onto' curve) is somewhat biased by the SWDs using Inference Web namespace. These are intended to be instance documents but also included many un-necessary class/property definitions. After removing PML documents, the 'onto*' curve looks very like the 'onto' curve but ends with a much flatter tail. Thus we can see a trend going from massive ontology development to populating and reuse ontologies.

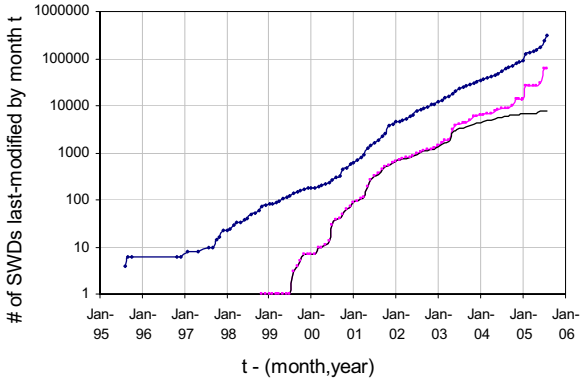

Figure 6: Number of SWDs and ontologies last modified by month t.

4.2.3 Semantic Websites

Swoogle's data shows that the cumulative distribution of number of websites[8] hosting more than m SWDs follows a Power Law distribution, as shown in Figure 7. There are a few websites (we call them semantic websites) hosting tens of thousands of SWDs, and there are also tens of thousands of websites hosting no more than ten. The former group mainly publishes automatically generated SWDs such as personal profiles (i.e., FOAF documents) and personal blog RSS feeds (see Table 3). The latter group is usually driven by virtual host technology; some blog hosting services assign unique virtual host name to each of their users. This Power Law distribution also benefits web directory crawling. Since only 1000 websites have more than 10 SWDs, it is worthwhile crawling them for SWDs.

5 Swoogle Semantic Web Search

This section presents the two primary search services provided by Swoogle: searching for SWDs and searching for Semantic Web terms (i.e., the URIs of classes and properties). Other specialized services have been developed, such as

[7]Its difficult to be definitive since Swoogle has been under active development over this time

[8]An website is uniquely identified by its URL (host name) but not its IP.

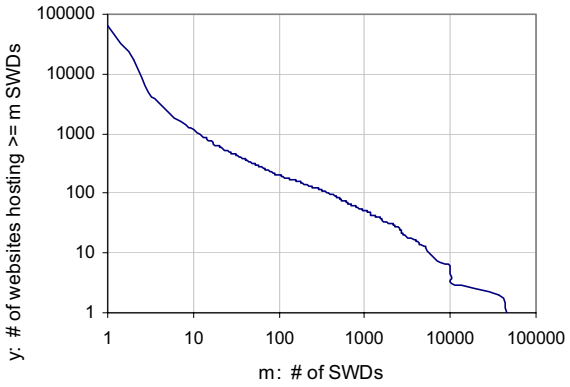

Figure 7: The number of websites hosting more than m SWDs follows a power law distribution ($y \propto x^{-0.7}$). The sharp drop starting at m=10000 is caused by Swoogle's sampling strategy - postpone indexing websites with large amount (i.e., more than 10K) of SWDs for fair sampling.

Table 3: Top 10 Semantic Websites

website URL	**# of SWDs**	**category**
http://onto.stanford.edu:8080/	45278	Inference Web
http://www.livejournal.com/	36141	FOAF
http://ch.kitaguni.tv/	10365	RSS
http://www.tribe.net/	10221	FOAF
http://blog.livedoor.jp/	10111	FOAF
http://www.greatestjournal.com/	10060	FOAF
http://www.wasab.dk/	7746	FOAF
http://yaplog.jp/	6780	FOAF
http://blogs.dion.ne.jp/	6181	FOAF
http://testers.cpan.org/	5684	RSS

searching for online SWDs supporting an hypothetical RDF graph [14], but are not described in this article.

5.1 Search Ontologies and Documents

SWDs are widely used physical containers of RDF graph; hence searching for them, especially those containing domain ontologies, is a common task. In this kind of search, the desired results are the URLs of SWDs and the search criteria include constraints on the document metadata, content metadata and relation metadata.

Document metadata

Document metadata captures the features annotating an SWD as a web page. Table 4 lists some of the document metadata Swoogle collects and maintains [13]. Not all of the properties are useful in queries and some, such as a document's MD5 hash value are primarily used internally (e.g., to easily recognize that two SWDs have identical content).

Table 4: Swoogle maintains a number of metadata for each Semantic Web document.

	Property	**Meaning**
1	url	the URL of the SWDs
2	extension	the detected file extension
3	last-modified	the last-modified field in http header
4	date-discover	the date when this SWD has first discovered by Swoogle
5	date-ping	the date when this SWD has been pinged by Swoogle
6	md5hash	the MD5 hash for this SWD for detect content changes
7	state-ping	whether this SWD has been changed or offline in last ping
8	document size	the size of document in Bytes
9	cache url	the URL of latest cached version of this SWD
10	state-parse	Is this SWD pure or embedded? Does it have parse error?
11	rdf-syntax	Is this SWD written in RDF/XML, NTriples or N3?

Content Metadata

Content Metadata describes the RDF graph encoded in a SWD with a focus on class-instance level features, i.e., individual objects as opposed to classes or properties. Swoogle analyzes the RDF triples in an SWD to recognize which participate in the definition of new term and which make assertions about individuals (e.g., John's age is 26)[9] Based on this analysis, Swoogle computes a measure of an SWD's *ontology ratio* and indexes a document's content using individual level features.

A document's *ontology ratio* is a heuristic measure of the degree to which it can be considered as an ontology. The defining characteristic of an ontology is that it defines or adds to the definition of terms to be used by other documents. Swoogle's ontology metric is the fraction of individuals being recognized as classes and properties. For example, given an SWD defining a class "Color" and populating the class with three class-instances 'blue', 'green' and 'red', its ontology ratio is 25% since only one out of the four is defined as class. A document with a high ontology ratio indicates a preference for adding term definition rather than populating existing terms. According to Swoogle, an SWD is an

[9]In an SWD D, an individual (i.e., a class-instance) X is introduced by a triple (X, *rdf:type*, Y). Here, X could be defined as a class (when Y is the sub-class of *rdfs:Class*), a property (when Y is the sub-class of *rdf:Property*) or an individual object (otherwise).

ontology document if it has defined at least one term, and it is 'strictly' an ontology if its ontology ratio exceeds 0.8. The former one enables us to compute a lower bound of the amount of ontologies, and the latter one intends to meet the common understanding of the term 'ontology'.

In an SWD, triples making assertions about the same class-instance are grouped and analyzed by Swoogle. For each class-instance, both the lexemes extracted from its URI and the literal descriptions in associated triples are good keywords. For example in FOAF ontology, the triple

foaf:Person rdfs:comments "A human being"

contributes a literal description to the defined class *foaf:Person*; moreover, *rdfs:comments* has nothing to do with the content of either the class or the ontology. Hence, Swoogle maintains full-text indexing on only the URI and literal description of the defined resource (instead of the entire SWD) to index the content of SWDs with better precision.

Relation Metadata

Relation Metadata characterizes various binary relations between (i) SWDs and XML Namespaces, (ii) SWDs and RDF resources, and (iii) SWDs and other SWDs, as shown in Figure 8. We briefly describe each kind of relation in turn.

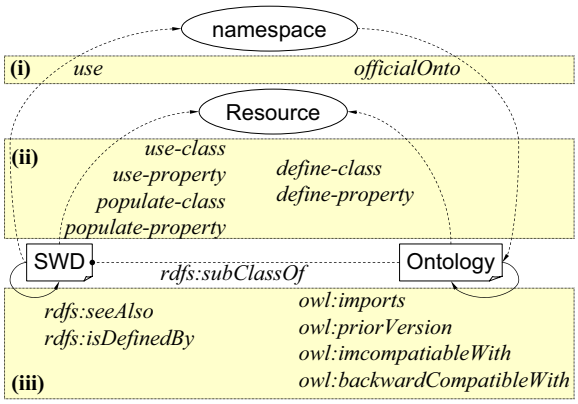

Figure 8: Swoogle discovers and indexes different types of binary relations that can hold among RDF Resources, Semantic Web documents, and namespaces.

SWD-namespace relation. One simple, but effective and efficient, way to find SWDs that use a particular resource or ontology is to search for documents using a given namespace. For example, in order to find all SWDs related to Inference Web, we can search for SWDs using the Inference Web namespace[10].

Maintaining data about the links between SWDs and their namespaces is also very useful in determining a namespace's **official ontology**, which is an

[10]The current Inference Web namespace is `http://inferenceweb.stanford.edu/2004/07/`

SWD that defines all of the classes and properties in that namespace. Web conventions suggest that the namespace part of an URI determines the location (as a URL) of the corresponding ontology document, but it need not be the case and exceptions are common.

Swoogle attempts to find the location of *official ontology* using one of the following: (i) the namespace of RDF resource; (ii) the redirected URL of the namespace (e.g. `http://purl.org/dc/elements/1.1/` is redirected to `http://dublincore.org/2003/03/24/dces`); or (iii) the only URL which has a namespace in its absolute path (e.g. the SWD `http://xmlns.com/foaf/0.1/index.rdf` is the official ontology of `http://xmlns.com/foaf/0.1/`; however, the SWD `http://xmlns.com/wordnet/1.6/Person` is not the official ontology of `http://xmlns.com/wordnet/1.6/` since there are many other candidate SWDs such as `http://xmlns.com/wordnet/1.6/Source`.). Table 5 shows the results of Swoogle's approach. Although the second and third heuristics do not improve the overall performance, they are needed to find the very popular Dublin Core and FOAF ontologies.

Table 5: Swoogle's heuristics for identifying the official ontology for a namespace's official work for slightly over 60% of our 4508 test cases.

Type	**number of ns/percent**
1. namespace correct	2661(59%)
2. redirected	18(0.4%)
3. single-RDF	150(3.4%)
4. confused	1679(37.2%)

SWD-resource relation. Swoogle recognizes six kinds of usage of a named RDF resource T[11] in an SWD D as shown in Table 6. A document can *define*, *use* or *populate* a class or property. For example, if these triples are in a SWD D

univ:Student rdfs:subclass foaf:Person.
univ:john rdf:type univ:Student.

we would record D as defining the class univ:Student, using the class foaf:Person, populating the class univ:Student, and populating the property rdfs:subClass.

SWD-SWD relation. The standard Semantic Web ontologies define properties to link RDF documents directly in order to facilitate finding term definitions and resolving the external term references. RDFS allows documents to be linked with the *rdfs:seeAlso* and *rdfs:isDefinedBy* properties. Both the domain and range of these properties is the generic rdfs:Resource, which means that these links might point to other web pages than SWDs; hence RDF validation is needed to insure the relationship is useful. OWL allows ontology documents to be associated by sub-properties of *owl:OntologyProperty* including *owl:imports*, *owl:priorVersion*, *owl:backwardCompatibleWith*, and *owl:incompatibleWith*.

[11] An RDF resource can be either named by a URI or anonymous according to RDF [23].

Table 6: We identify six different binary relations of interest that can hold between a Semantic Web Document and an RDF resource.

resource usage	**condition**
define-class	D has a triple (T, *rdf:type*, *META*) such that *META* is a sub-class of *rdfs:Class*.
define-property	When D has a triple (T, *rdf:type*, *META*) such that *META* is a sub-class of *rdf:Property*.
use-class	When D has a triple (_, P, T) such that the range of P is a sub-class of either *rdfs:Class*, or D has a triple (T, P, _) such that the domain of P is a sub-class of *rdfs:Class*,
use-property	When D has a triple (_, P, T) such that the range of P is a sub-class of either *rdf:Property*, or D has a triple (T, P, _) such that the domain of P is a sub-class of *rdf:Property*,
populate-class	When D has a triple (_, rdf:type, T).
populate-property	When D has a triple (_, T, _).

5.2 Searching for Semantic Web Vocabulary

Swoogle provides a *Term Search* capability to search for RDF vocabulary – URI references for terms (i.e., classes and properties), and for merging and reporting on information relevant to RDF terms. We will first describe how URI references are processed to yield indexible keywords and then describe the definitional information collected for terms.

5.2.1 Deconstructing URIs

As described in [3], a URI consists of two parts: a namespace, which helps making the URI unique, and a *local -name*, which conveys the meaning. For example, the URI `http://xmlns.com/foaf/0.1/#mbox` has the namespace `http://xmlns.com/foaf/0.1/` and the local name *mbox*.

Since not all URIs use '#', such as `http://xmlns.com/foaf/0.1/Person`, special operations must be taken to correctly extract local-name from such URIs. In order to avoid errors due to simply break an URI at the last slash character, e.g., the URI `http://foo.com/ex.owl` is not splitable, Swoogle uses the namespace declaration obtained during syntactic parsing to fulfill this task. It is common practice for ontology developers to give terms long and descriptive local names such as *number_of_wheels* or *GraduateResearchAssistant*. To provide flexibility, Swoolge provides several mechanisms for retrieving RDF terms based on portions of their local names. First, Swoogle uses a simple grammar derived from a variety of heuristics to parse a local name into a sequence of *lexemes*. The local name *FoodWeb*, for example, is indexed by itself as well as

the lexemes *food* and *web*. Swoogle also provides a more general, and more expensive, ability to retrieve RDF terms whose names include a given sub-string. A user can lookup terms whose local name is either exactly "person", or has a substring "person", or whose full URI has the substring "foo.com".

5.2.2 Resource Description

Figure 9 shows three kinds of information that together describe what RDF terms mean and how they are used. The *Class-property (C-P) bonds* in this figure show the properties that can be used to modify the instances of a given class, i.e., the *rdfs:domain* relation. Given a named class X, its *Term definition* consists of a set of triples in form of (X, $?Y$, $?Z$); its *ontological C-P bond* can be captured by *rdfs:domain* in RDFS, and by *owl:onProperty* in OWL; and its *empirical C-P bond* is learned from triples grouped by X's instances.

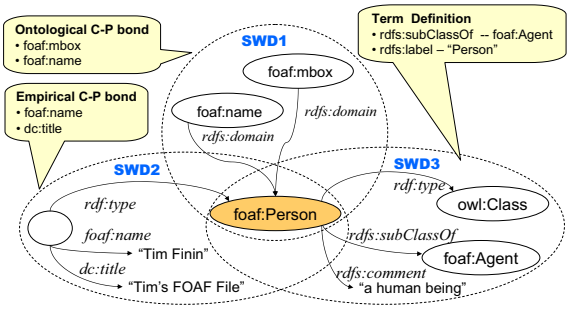

Figure 9: Class definition and class-property bond of *foaf:Person*

A Swoogle query can constrain results in a Term Search using any of the following modifiers:

- *Resource family filter.* Swoogle categorizes terms as belonging to a representational family which is the namespace of meta class to which the term belongs. For example, when the RDF class foaf:Person is defined by asserting that foaf:Person's rdf:type is owl:Class, we say that foaf:Person's family is OWL. Current Swoogle uses the four well-known families: "RDFS", "RDF", "DAML" and "OWL".

- *Resource type filter.* Terms can also being categorized by the type they have been defined, i.e., a *class*, a *property* or both. For example, the statement that foaf:Person's rdf:type is owl:Class indicates that the type of foaf:Person is class. Note that the type of a term can be defined as both class and property. While this is logically inconsistent, it does arise in practice, due in part to the distributed nature of the Semantic Web.

- *Literal definition filter.* Swoogle builds a full-text index on the literal definition of the given resource. For example, the statement that foaf:Person's rdf:comment is "a human being' helps generate keywords for foaf:Person.

Swoogle supports relation queries over both types of C-P bonds. Using Swoogle's Ontology Dictionary[12] to explore the *foaf:Person* term, one finds that it has 156 ontological properties (properties formally defined to hold with foaf:Person according to 17 ontology documents indexed by Swoogle) and over 500 empirical properties (properties asserted by foaf:Person instances from over 70K SWDs).

That people and agents use properties in ways that do not adhere to their formal specification (or to fail specify them completely) is an interesting phenomenon. It suggests a process by which ontologies might emerge from use. Agents use properties in an undisciplined manner to make assertions about the world. Ontology engineers can use systems like Swoogle to recognize such unexpected and unplanned usage and consider formally extending their ontologies to sanction it.

We are planning to extend this approach to model the Property-Class bond. The ontological version of this occurs when the *rdfs:range* property is used to connect a property and the class of objects that to which its values must belong. The empirical version is detected whenever a property is used to make an assertion.

6 Swoogle Semantic Web Ranking

Google's success with its PageRank algorithm has demonstrated the importance of using a good technique to order the results returned by a query. Swoogle uses two custom ranking algorithms, OntoRank and TermRank, to order a collection of SWDs or RDF terms, respectively. These algorithms are based on an abstract "surfing" model that captures how an agent might access Semantic Web information published on the Web. Navigational paths on the Semantic Web are defined by RDF triples as well as by the resource-SWD and SWD-SWD relations. However, a centralized analysis is required to reveal most of these connections.

6.1 Ranking SWDs using OntoRank

Since a web document is the primary unit of data access on the Web, Swoogle aggregates navigational paths to the SWD level [13] and recognizes three generalized inter-document links.

- An **extension (EX)** relation holds between two SWDs when one defines a term using terms defined in another. EX subsumes the define-class and define-property *resource-SWD* relations, the sub-class and sub-property *resource-resource* relations, and the officialOnto *namespace-SWD* relation. For example, an SWD $d1$ *EX* another SWD $d2$ when both conditions are met: (i) $d1$ defines a class $t1$, $t1$ is subclass of a class$t2$, and $t2$'s official ontology is $d2$; and (ii) $d1$ and $d2$ are different SWDs.

[12] `http://swoogle.umbc.edu/modules.php?name=Ontology_Dictionary&option=0`

- An **use-term (TM)** relation holds between two SWDs when one uses a term defined by another. TM subsumes the use-class, use-property, populate-class and populate-property *resource-SWD* relations, and the officialOnto *namespace-SWD* relation. For example, an SWD $d1$ *TM* another SWD $d2$ when both conditions are met: (i) $d1$ uses a resource t as class, t's official ontology is $d2$; and (ii) $d1$ and $d2$ are different SWDs.

- An **import (IM)** relation holds when one SWD imports, directly or transitively, another SWD and corresponds to the imports *resource-resource* relation.

Google's simple *random surfer* model is not appropriate for these paths. For example, an agent reasoning over the content found in an SWD should access and process all of the ontologies it imports. Swoogle's *OntoRank* is based on the *rational surfer model*, which emulates an agent's navigation behavior at document level. Like the random surfer model, an agent either follows a link from one SWD to another with a constant probability d or jumps to a new random SWD. It is 'rational' in that it jumps non-uniformly with the consideration of link semantics and models the need for agents to access the ontologies referenced in SWDs. When encountering an SWD α, the *rational surfer* will (transitively) import the "official" ontologies that define the classes and properties used by α.

Let $link(\alpha, l, \beta)$ be the semantic link from an SWD α to another SWD β with tag l; $linkto(\alpha)$ be a set of SWDs directly link to an SWD α; $weight(l)$ be a user specified navigation preference on semantic links with tag l; $OTC(\alpha)$ be a set of SWDs that (transitively) import α as ontology; $f(x, y)$ and $wPR(x)$ be two intermediate functions.

OntoRank is computed in two steps: (i) iteratively compute the rank, $wPR(\alpha)$, of each SWD α until it converges using equations 1 and 2; and (ii) transitively pass an SWD's rank to all ontologies it imported using equation 3.

$$wPR(\alpha) = (1 - d) + d \sum_{x \in linkto(\alpha)} \frac{wPR(x) \times f(x, \alpha)}{\sum\limits_{link(x,\_,y)} f(x, y)} \tag{1}$$

$$f(x, \alpha) = \sum_{link(x,l,\alpha)} weight(l) \tag{2}$$

$$OntoRank(\alpha) = wPR(\alpha) + \sum_{x \in OTC(\alpha)} wPR(x) \tag{3}$$

We evaluated *OntoRank* using a Swoogle-collected dataset *DS-JAN* consisting of 330,000 SWDs. Of these, about 1.5% were ontologies, 24% were FOAF documents and 60% were RSS documents. The documents included about 1.7M document level relations. Table 7 compares the performance between PageRank and OntoRank in boosting ontologies, i.e., rating ontology document higher than normal SWDs. Ten popular local-names in Swoogle vocabulary were selected as Swoogle document search terms. For each query, the same search

result is ordered by both PageRank and OntoRank respectively. We compared the number of *strict* ontology documents (SWDs with at least 0.8 OntoRatio) in the first 20 results in either order. The difference reflects an average 40% improvement of OntoRank over PageRank.

query	C1:# of SWOs by OntoRank	C2:# of SWOs by PageRank	Difference (C1-C2)/C2
name	9	6	50.00%
person	10	7	42.86%
title	13	12	8.33%
location	12	6	100.00%
description	11	10	10.00%
date	14	10	40.00%
type	13	11	18.18%
country	9	4	125.00%
address	11	8	37.50%
organization	9	5	80.00%
Average	11.1	7.9	40.51%

Table 7: OntoRank vs. PageRank: OntoRank helps Swoogle Search find more ontologies in top 20 results

6.2 Ranking Terms

Swoogle uses the TermRank algorithm to order the RDF terms returned by a term search query. This ranks terms based on how often they are used, estimated by the cardinality of the *swoogle:uses* relation for each term, i.e., the number of SWDs that use the term. However, this does not take into account OntoRank's estimate of the likelihood that an SWD will be accessed. Equations 4 and 5 are used to compute the TermRank of a Semantic Web term. Intuitively, we split the rank of SWDs to the terms populated by them. Given a term t and an SWD α, $TWeight(\alpha, t)$ is computed from $cnt\_uses(\alpha, t)$, which shows how many times α uses t, and $|\{\alpha|uses(\alpha, t)\}|$, which shows how many SWDs in the entire SWD collection have used t.

$$TermRank(t) = \sum_{uses(\alpha,t)} \frac{OntoRank(\alpha) \times TWeight(\alpha,t)}{\sum_{uses(\alpha,x)} TWeight(\alpha,x)} \quad (4)$$

$$TWeight(\alpha, t) = cnt\_uses(\alpha, t) \times |\{\alpha|uses(\alpha, t)\}| \quad (5)$$

Table 8 lists the ten highest ranked classes in *DS-JAN* having 'person' as local name as ordered by TermRank. For each class, *pop(swd)* refers to the number of SWDs that populate (create instances of) the class; *pop(i)* refers to the number of its instances; and *def(swd)* refers to the number of SWDs that contribute to its definition. Not surprisingly, the *foaf:Person* class is number

one. Note that the sixth term is a common mis-typing for the first – the correct local name is capitalized. The tenth term has apparently made the list by virtue of the high OntoRank score of the SWD that defines it.

TR	Resource URI	pop(swd)	pop(i)	def(swd)
1	http://xmlns.com/foaf/0.1/Person	74589	1260759	17
2	http://xmlns.com/wordnet/1.6/Person	2658	785133	80
3	http://www.aktors.org/ontology/portal#Person	267	3517	6
4	ns1:Person [1]	257	935	1
5	ns2:Person [2]	277	398	1
6	http://xmlns.com/foaf/0.1/person	217	5607	0
7	http://www.amico.org/vocab#Person	90	90	1
8	http://www.ontoweb.org/ontology/1#Person	32	522	2
9	ns3:Person [3]	0	0	1
10	http://description.org/schema/Person	10	10	0

Table 8: Swoogle's TermRank algorithm returns these terms as the top ten results when searching for classes with 'person' in their local name.

[1] ns1 - http://www.w3.org/2000/10/swap/pim/contact#
[2] ns2 - http://www.iwi-iuk.org/material/RDF/1.1/Schema/Class/mn#
[3] ns3 - http://ebiquity.umbc.edu/v2.1/ontology/person.owl#

7 Other Approaches

There are several possible models for what a Semantic Web search engine should be and the paradigm is not yet fixed. We will very briefly mention approaches that others are pursuing as well as some of our own alternative ideas.

One model of a Semantic Web search engine is based on a database, either a centralized one such as Intellidimention's RDFGateway[13], Sesame [6], and TAP [18], or a distributed peer-to-peer federation of agents such as Edutella [25], RDFPeers [7], and SERSE [30]. Another model is typified by the Schemaweb[14] and DAML ontology library, which is supported by manual submissions. There are also "niche" search engines that focus on a particular kind of Semantic Web information, such as several that collect FOAF information.

We have explored the problems and corresponding solutions through several earlier prototype systems. While these systems do not exhaust the space of possibilities, they have challenged us to refine the techniques and provided valuable experience.

The first prototype, OWLIR [28], is an example of a system that takes ordinary text documents as input, annotates them with semantic web markup, swangles the results and indexes them in a custom information retrieval system. OWLIR can then be queried via a custom query interface that accepts free text as well as structured attributes.

While we used OWLIR to explore the general issues of hybrid information retrieval, the implemented system was built to solve a particular task – filtering UMBC student event announcements. Students received weekly email message

[13] `http://www.intellidimension.com/`
[14] http://www.schemaweb.info/

listing over 50 events – public lectures, club meetings, sporting matches, movie screenings, outing, etc. Our goal was to process these messages, produce sets of event descriptions containing both text and markup, enrich the descriptions using local knowledge and reasoning and indexing the result with a custom information retrieval system. A simple form-based query system allows a student to enter a query that includes both structured information (e.g., event dates, types, etc.) and free text. The form generates a query document in the form of text annotated with DAML+OIL markup. Queries and event descriptions were processed by reducing the markup to triples, enriching the structured knowledge using a local knowledge base and inference, and swangling the triples to produce acceptable indexing terms. The result was a text-like query that can be used to retrieve a ranked list of events that match the query.

Our second prototype, Swangler [24, 16], is a system that annotates RDF documents encoded in XML with additional RDF statements attaching *swangle terms* that are indexible by Google and other standard Internet search engines. We call this process swangling, for 'Semantic Web mangling.' These documents, when available on the Web, are discovered and indexed by conventional search engines like Google and can be retrieved using queries containing text, bits of XML and swangle terms.

8 Applications

We have used Swoogle to support several applications and use cases, including helping Semantic Web researchers find ontologies and data, semantic search over documents representing proofs, and finding and evaluating semantic associations in large graph databases. These have helped us to explore what services a Semantic Web search engine can provide.

In the NSF supported SPIRE project a group of biologists and ecologists is exploring how the Semantic Web can be used to publish, discover and reuse models, data and services [17]. This leads to a requirement to help researchers find appropriate ontologies and terms to annotate their data and services and also for services to discover data and services published by others. Swoogle's Ontology Search interface allow a user to search for existing ontology documents which define terms having user-supplied keywords as the substring of their local-name. For example, to find an ontology that can be used to describe temporal relations one might search for ontologies with the keywords *before*, *after* and *interval*. Swoogle's Ontology Dictionary can be used to find the definitions of properties or classes with a given set of keywords. It can assemble and merge definitions from multiple sources, lists terms sharing the same namespace or sharing the same local-name, and list associations between classes and properties. Those associations can either be "ontological" (e.g., the foaf:knows property is defined to exist between instances of foaf:person) or "empirical" (e.g., the dc:creator property has been applied to an instance of foaf:Person). Judging the ranking or popularity of terms and ontologies is also of relevance here. Consensus models of the community as reflected in the ontologies would

tend to be ranked highly, and thus used more often by those searching.

Swoogle is also being used in conjunction with Inference Web (IW) [11], which explicitly represents proofs using an OWL ontology, Proof Markup Language (PML) [10]. One IW component, IWSearch [15], uses Swoogle to discover newly published/updated PML documents in the Web, and itself is powered by a specialized instance of Swoogle to index and search instances found in a corpus of over 50,000 PML documents. By indexing the conclusion part of a proof NodeSet instance, one may discover additional NodeSets sharing the same conclusion as the one from the given justification tree, and thus expand the justification tree with additional proofs.

Swoogle is also being used by SEMDIS, an NSF project jointly conducted with colleagues at U. Georgia that automates the discovery, merging and evaluation of semantic associations in data drawn from variety of information sources. SEMDIS augments information collected from the Semantic Web with additional data extracted from text documents and Databases [1]. The result, encoded as a large RDF graph along with provenance assertions and trust information is processed to discover and evaluate "interesting" semantic associations [29]. Two kinds of Semantic Web searches are done: (i) searching for a semantic association (connected sub-graph) in the large scale RDF graph and (ii) searching SWDs that (partially) support a given semantic association. The first reduces to the problem of finding paths between two nodes in a graph, and is common issue in RDF databases. The second is a type of provenance search, i.e., finding a set of SWDs that (partially) imply a hypothesized semantic association; and it has been prototyped by a RDF molecule based approach [14].

9 Conclusions and Future Work

Search engines became a critical component of the Web's infrastructure as the Web's size grew. As the Semantic Web grows, we will need search engines that can efficiently handle Semantic Web content. While we can't be sure what form this content will take in the future, the current standard is based on Semantic Web documents. We have discussed the general differences encountered in building a search engine for Semantic Web documents rather than HTML documents. We've also described in some detail the design and implementation of Swoogle, the first search engine designed for the Semantic Web in the context of the Web.

We are continuing to use Swoogle to study the growth and characteristics of the Semantic Web and the current practices in using RDF and OWL. We are also developing new features and capabilities and exploring how it can be used in novel applications. Many open issues remain.

One set of open problems involves scale. Techniques which work today with 5×10^6 documents may fail when the Semantic Web has 5×10^8 documents. Extending Swoogle to index and effectively query over large amounts of instance data is one challenge. We estimate that the SWDs currently on the Web contain

[15] http://iw4.stanford.edu/iwsearch/IWSearch

over 5×10^8 triples, a number that neither current relational database nor custom triple stores can handle efficiently. Some of these problems could potentially be solved by moving away from COTS open source software we are using to creating custom designed index stores and distributed systems – analogous to what Google has done for conventional Web searches. It remains to be seen however if that alone would suffice. We are also interested in developing a query system that can be used to find *RDF molecules* [14] in a reasonably efficient manner.

We also need to experiment with how much and where a Semantic Web search engine should reason over the contents of documents and queries. In OWLIR we experimented with expanding documents using reasoning prior to indexing. A complementary approach is to use a kind of query expansion [31] on queries containing RDF terms. Partly this is related to the problem of scale – the larger the collection becomes, the less one can afford to reason over it. Other issues involve trust and the use of local knowledge not part of the Semantic Web.

Information encoded in RDF is beginning to show up embedded in other documents, such as PDF and XHTML documents, JPG images and EXCEL spread sheets. When techniques for such embedding become standard, we expect the growth of Semantic Web content on the Web to accelerate dramatically. This will add a new requirement for hybrid information retrieval systems [16] that can index documents based on words as well as RDF content.

Finally, we are continuing to experiment with how Swoogle can be used to support applications, including Spire, Semdis and Inference Web, Each of which exercises different aspects of a Semantic Web search engine.

References

[1] Boanerges Aleman-Meza, Chris Halaschek, I. Budak Arpinar, and Amit Sheth. Context-aware semantic association ranking. In *Proceedings of 1st International Workshop on Semantic Web and Databases*, 2003.

[2] Dave Beckett. RDF/XML syntax specification (revised). `http://www.w3.org/TR/2004/REC-rdf-syntax-grammar-20040210/`, February 2004.

[3] T. Berners-Lee, R. Fielding, and L. Masinter. RFC 2396 - uniform resource identifiers (URI): Generic syntax. `http://www.faqs.org/rfcs/rfc2396.html`, 1998.

[4] Tim Berners-Lee, Jim Hendler, and Ora Lassila. The semantic web. *Scientific American*, 284(5):35–43, 2001.

[5] Dan Brickley and Ramanathan V. Guha. RDF vocabulary description language 1.0: RDF schema. `http://www.w3.org/TR/2004/REC-rdf-schema-20040210/`, February 2004.

[6] Jeen Broekstra, Arjohn Kampman, and Frank van Harmelen. Sesame: A generic architecture for storing and querying RDF and RDF Schema. In

Proceedings of the 1st International Semantic Web Conference, pages 54–68, 2002.

[7] Min Cai and Martin Frank. RDFPeers: a scalable distributed RDF repository based on a structured peer-to-peer network. In *Proceedings of the 13th international conference on World Wide Web*, pages 650–657, 2004.

[8] Jeremy J. Carroll, Christian Bizer, Patrick Hayes, and Patrick Stickler. Named graphs, provenance and trust. Technical Report HPL-2004-57, HP Lab, May 2004.

[9] Harry Chen, Tim Finin, and Anupam Joshi. Semantic web in in the context broker architecture. In *Proceedings of the 2nd IEEE International Conference on Pervasive Computer and Communications*, 2004.

[10] Paulo Pinheiro da Silva, Deborah L. McGuinness, and Richard Fikes. A proof markup language for semantic web services. Technical Report KSL-04-01, Stanford, 2004.

[11] Paulo Pinheiro da Silva, Deborah L. McGuinness, and Rob McCool. Knowledge provenance infrastructure. *Data Engineering Bulletin*, 26(4):26–32, 2003.

[12] Mike Dean and Guus Schreiber. OWL web ontology language reference. `http://www.w3.org/TR/2004/REC-owl-ref-20040210/`, February 2004.

[13] Li Ding, Tim Finin, Anupam Joshi, Rong Pan, R. Scott Cost, Yun Peng, Pavan Reddivari, Vishal C Doshi, and Joel Sachs. Swoogle: A search and metadata engine for the semantic web. In *Proceedings of the 13th ACM Conference on Information and Knowledge Management*, 2004.

[14] Li Ding, Tim Finin, Yun Peng, Paulo Pinheiro da Silva, and Deborah L. McGuinness. Tracking rdf graph provenance using rdf molecules. Technical Report TR-CS-05-06, UMBC, April 2005.

[15] Andreas Eberhart. Survey of rdf data on the web. Technical report, International University in Germany, 2002.

[16] Tim Finin, James Mayfield, Clay Fink, Anupam Joshi, and R. Scott Cost. Information Retrieval and the Semantic Web. In *Proceedings of the 38th International Conference on System Sciences*, January 2005.

[17] Tim Finin and Joel Sachs. Will the Semantic Web Change Science? *Science Next Wave*, September 2004. `http://nextwave.sciencemag.org/`.

[18] Ramanathan V. Guha and Rob McCool. TAP: A semantic web test-bed. *Journal of Web Semantics*, 1(1):81–87, 2003.

[19] Yuanbo Guo, Zhengxiang Pan, and Jeff Heflin. An evaluation of knowledge base systems for large OWL datasets. In *International Semantic Web Conference*, pages 274–288, 2004.

[20] Patrick Hayes. RDF semantics. http://www.w3.org/TR/2004/REC-rdf-mt-20040210/, February 2004.

[21] Ian Horrocks, Peter F. Patel-Schneider, Harold Boley, Said Tabet, Benjamin Grosof, and Mike Dean. SWRL: A semantic web rule language combining owl and RuleML. http://www.w3.org/Submission/2004/SUBM-SWRL-20040521/, May 2004.

[22] Lalana Kagal. Rei: A policy language for the me-centric project. Technical Report HPL-2002-270, HP Labs, 2002.

[23] Graham Klyne and Jeremy J. Carroll. Resource description framework (RDF): Concepts and abstract syntax. http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/, February 2004.

[24] James Mayfield and Tim Finin. Information retrieval on the Semantic Web: Integrating inference and retrieval. In *Proceedings of the SIGIR Workshop on the Semantic Web*, August 2003.

[25] Wolfgang Nejdl, Boris Wolf, Steffen Staab, and Julien Tane. Edutella: Searching and annotating resources within an rdf-based p2p network, 2001.

[26] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

[27] Peter F. Patel-Schneider, Patrick Hayes, and Ian Horrocks. OWL web ontology language semantics and abstract syntax. http://www.w3.org/TR/2004/REC-owl-semantics-20040210/, February 2004.

[28] Urvi Shah, Tim Finin, Anupam Joshi, James Mayfield, and R. Scott Cost. Information retrieval on the semantic web. In *Proceeding of the 11th ACM Conference on Information and Knowledge Management*, 2002.

[29] Amit Sheth, Boanerges Aleman-Meza, I. Budak Arpinar, Chris Halaschek, Cartic Ramakrishnan, Clemens Bertram, Yashodhan Warke, David Avant, F. Sena Arpinar, Kemafor Anyanwu, and Krys Kochut. Semantic association identification and knowledge discovery for national security applications. *Special Issue of Journal of Database Management on Database Technology for Enhancing National Security*, 16(1), 2005.

[30] Valentina A. M. Tamma, Ian Blacoe, Ben Lithgow Smith, and Michael Wooldridge. Serse: Searching for semantic web content. In Ramon López de Mántaras and Lorenza Saitta, editors, *ECAI*, pages 63–67, 2004.

[31] Ellan M. Voorhees. Query expansion using lexical-semantic relations. In *17th International Conference on Research and Development in Information Retrieval (SIGIR '94)*, 1994.

[32] Youyong Zou, Tim Finin, Li Ding, Harry Chen, and Rong Pan. Using semantic web technology in multi-agent systems: a case study in the TAGA trading agent environment. In *Proceeding of the 5th International Conference on Electronic Commerce*, 2003.