Swoogle: A Search and Metadata Engine for the Semantic Web [*]

Li Ding Tim Finin Anupam Joshi Rong Pan R. Scott Cost Yun Peng
Pavan Reddivari Vishal Doshi Joel Sachs

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

ABSTRACT

Swoogle is a crawler-based indexing and retrieval system for the Semantic Web. It extracts metadata for each discovered document, and computes relations between documents. Discovered documents are also indexed by an information retrieval system which can use either character N-Gram or URIrefs as keywords to find relevant documents and to compute the similarity among a set of documents. One of the interesting properties we compute is *ontology rank*, a measure of the importance of a Semantic Web document.

Categories and Subject Descriptors:
H.3.3 [Information storage and Retrieval]: Information Search and Retrieval - *Retrieval models; Search process.*

General Terms:
Algorithms, Design.

Keywords:
Semantic Web, Search, Metadata, Rank, Crawler.

1. INTRODUCTION

Currently, the Semantic Web, (i.e. online documents written in RDF or OWL), is essentially a web universe parallel to the web of HTML documents. Semantic Web documents (SWDs) are characterized by semantic annotation and meaningful references to other SWDs. Since conventional search engines do not take advantage of these features, a search engine customized for SWDs, especially for ontologies, is needed by human users as well as by software agents and services. At this stage, human users are expected to be semantic web researchers and developers who are interested in accessing, exploring and querying RDF and OWL documents found on the web.

We introduce a prototype Semantic Web search engine called *Swoogle* to facilitate the development of the Semantic Web, especially the following three activities:

Finding appropriate ontologies. Failure to easily find an appropriate ontology for a particular markup task typically leads to the creation of a new ontology (or to the abandonment of the markup effort). Swoogle allows users to query for ontologies that contain specified terms anywhere in the document (including comments); for ontologies that contain specified terms as Classes or Properties; or for ontologies that are about a specified term (as determined by our IR engine). The ontologies returned are ranked according to our Ontology Rank algorithm, which seeks to capture the extent to which ontologies are used by the community. We believe that this use of Swoogle will both ease the burden of marking up data, and contribute to the emergence of canonical ontologies.

Finding instance data. In order to help users to integrate Semantic Web data distributed on the Web, Swoogle enables querying SWDs with constraints on what classes and properties being used/defined by them.

Characterizing the Semantic Web. By collecting metadata – especially inter-document relations – about the Semantic Web, Swoogle reveals interesting structural properties, allowing us to answer questions such as "how is the Semantic Web connected?", "how are ontologies referenced?", and "how are ontologies modified externally?".

Our development of Swoogle has highlighted a number of interesting research issues such as "what is the best way to index, digest and cache SWDs?", and "is it possible to create a meaningful *rank* measure that uses link semantics?".

Swoogle is designed as a system that automatically discovers SWDs, indexes their metadata and answers queries about it. This distinguishes it from other semantic web repositories and query systems in the literature. *Ontology based annotation systems*, such as SHOE [15], Ontobroker [9], WebKB [16], QuizRDF [8] and CREAM [11], focus on annotating online documents. However, their document indexes are based on the annotations rather than on the entire document, and they use their own ontologies which may not be suited for Semantic Web documents. It is notable that CREAM [11] had indexed 'proper reference' and 'relational

[*] Partial research support provided by DARPA contract F30602-00-0591 and NSF-ITR-IIS-0326460 and NSF-ITR-IDM-0219649.

CIKM'04, November 8–13, 2004, Washington, DC, USA.
Copyright 2004 ACM 1-58113-874-1/04/0011 ...$5.00.

metadata'. *Ontology repositories*, such as the DAML Ontology Library [1], SemWebCentral [4] and Schema Web [2], do not automatically discover semantic web documents but rather require people to submit URLs. They only collect ontologies which constitute a small portion of the Semantic Web. In addition, they simply store the entire RDF documents. Recently, some *Semantic Web browsers* have been introduced. Ontaria [5] is a searchable and browsable directory of RDF documents under development by the W3C; it stores the full RDF graphs of harvested SWDs, rather than focusing on metadata. Semantic Web Search [3] indexes individuals of well-known classes (e.g. foaf:Person, rss:Item). While the advantages of Swoogle's crawler-based discovery system are obvious, the decision to only store and reason over metadata is less obviously a good one. We made this choice since our key goal in building Swoogle is to design a system that will scale up to handle millions and even tens of millions of documents. Moreover, Swoogle enables rich query constraints on semantic relations.

The Swoogle architecture consists of web crawlers that discover SWDs; a metadata generator; a database that stores metadata about the discovered SWDs; a semantic relationships extractor; an N-Gram based indexing and retrieval engine; a simple user interface for querying the system; and agent/web service APIs to provide useful services.

We describe an algorithm, *Ontology Rank*, inspired by the Page Rank algorithm [14, 18, 19] which ranks online documents based on hyperlinks. Our algorithm takes advantage of the fact that the graph formed by SWDs has a richer set of relations than the graph of the World Wide Web. In other words, the edges in this graph have explicit semantics. Some are defined or derivable from the RDF and OWL languages (e.g., imports, usesTerm, version, extends, etc.) and others by common ontologies (e.g., FOAF's knows [1]).

Below, we describe the Swoogle architecture and the various metadata that we compute, and present some preliminary results summarizing the characteristics of the portion of the semantic web that our system has crawled and analyzed.

2. SEMANTIC WEB DOCUMENTS

Semantic web languages based on RDF (e.g., RDFS [2], DAML+OIL [3], and OWL [4]) allow one to make statements that define general terms (classes and properties), extend the definition of terms, create individuals, and to make assertions about terms and individuals already defined or created.

We define a *Semantic Web Document* (SWD) to be a document in a semantic web language that is online and accessible to web users and software agents. Similar to a document in IR, a SWD is an atomic information exchange object in the Semantic Web.

Current practice favors the use of two kinds of documents which we will refer to as semantic web ontologies (SWOs) and semantic web databases (SWDBs). These correspond to what are called T-Boxes and A-Boxes in the description logic literature [6, 13]. Since a document may consist of both T-Boxes and A-Boxes, we adopt a threshold based measure. We consider a document to be a SWO when a significant proportion of the statements it makes define new terms (e.g., new classes and properties) or extend the definitions of terms defined in other SWDs by adding new properties or constraints. A document is considered as a SWDB when it does not define or extend a significant number of terms. A SWDB can introduce individuals and make assertions about them or make assertions about individuals defined in other SWDs.

For example, SWD http://xmlns.com/foaf/0.1/index.rdf is considered a SWO since its 466 statements (i.e. triples) define 12 classes and 51 properties but introduce no individuals. The SWD http://umbc.edu/~finin/foaf.rdf is considered to be a SWDB since it defines or extends no terms but defines three individuals and makes statements about them.

Between these two extremes, some SWDs are intended to be both an ontology that defines a set of terms to be used by others, as well as a useful database of information about a set of individuals. Even a document that is intended as an ontology might define individuals as part of the ontology. Similarly, a document that is intended as defining a set of individuals might introduce some new terms in order to make it easier to describe the individuals, [5] (e.g. http://www.daml.ri.cmu.edu/ont/USCity.daml, http://reliant.teknowledge.com/DAML/Government.owl).

3. SWOOGLE ARCHITECTURE

As shown in Figure 1, Swoogle's architecture can be broken into four major components: SWD discovery, metadata creation, data analysis, and interface. This architecture is data centric and extensible; components work independently and interact with one another through a database.

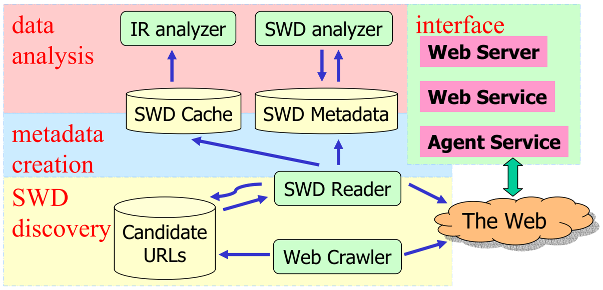


Figure 1: The architecture of Swoogle

The *SWD discovery* component discovers potential SWDs throughout the Web and keeps up-to-date information about SWDs.

The *metadata creation* component caches a snapshot of a SWD and generates objective metadata about SWDs at both the syntax level and the semantic level.

The *data analysis* component uses the cached SWDs and the created metadata to derive analytical reports, such as classification of SWOs and SWDBs, rank of SWDs, and the IR index of SWDs.

The *interface* component focuses on providing data services to the Semantic Web community. We have implemented a Web interface at http://www.swoogle.org, and we

[1] http://xmlns.com/foaf/0.1/
[2] http://www.w3.org/TR/rdf-schema/
[3] http://www.w3.org/TR/daml+oil-reference
[4] http://www.w3.org/TR/owl-semantics/
[5] Programming languages introduce the notion of a modules, importing and exporting to make these intentions explicit

are working on making Swoogle a Web Service for software agents.

We elaborate on each component in the following sections.

4. FINDING SWDS

Finding URLs of SWDs is by itself an interesting engineering challenge. A straightforward approach is to search through a conventional search engine. By May 25, 2004, Google had indexed 4,285,199,774 web documents. It is not possible for Swoogle to parse all documents on the web to see if thery are SWDs. (Even if it were computationally feasible, most search engines, including Google, return at most 1,000 results per query). We developed a set of crawlers employing a number of heuristics for finding SWDs.

First, we developed a *Google crawler* to search URLs using the Google Web Service. We start with type extensions, such as ".rdf", ".owl", ".daml", and ".n3". Although they are not perfect SWD indicators, Table 1 shows that they have fair precision.

extension	# discovered(100%)	# SWD
rdf	184,992	111,350(60%)
rss	8,359	7,712(92%)
owl	4,669	3,138(67%)
n3	4,326	1,523(35%)
daml	3,699	2,256(61%)
no extension	154,591	7,258(5%)

Table 1: Extensions of SWD (Aug 30, 2004)

To overcome Google's limit of returning only the first 1000 results for any query, we append some constraints (keywords) to construct more specific queries, and then combine their results (see Table 2). Such query expansion techniques have enabled us to collect as many as 20K candidate URLs of SWDs at a time. Since Google changes its PageRanks daily, we also expect to discover new SWDs by running the same query weekly (in fact, Swoogle already has 200K urls discovered by the Google Crawler).

query string	number of pages
rdf	5,230,000
filetype:rdf rdf	246,000
filetype:rss rdf	13,800
filetype:daml rdf	4,360
filetype:n3 rdf	2,630
filetype:owl rdf	1,310
filetype:rdfs rdf	304

Table 2: Google search results (May 25,2004)

We have also developed the *Focused Crawler*, which crawls documents within a given website. In order to reduce search complexity and improve precision, simple heuristics, like extension constraint (e.g. documents with ".jpg" or ".html" extensions are seldom SWDs) and focus constraint (e.g. only crawl URLs relative to the given base URL), are used to filter out those documents likely to be irrelevant. Swoogle provides a web interface where registered users can submit a URL of either a SWD or a web directory under which many SWDs may be present, e.g. http://daml.umbc.edu/ontologies/. We have initiated focused crawls from many Semantic Web URLs know to us, and actively invite the SW community to submit further URLs for focused crawling.

Since SWDs can be discovered by semantic links while parsing SWDs, we developed the JENA2 [6] based *Swoogle Crawler*. It both analyzes the content of a SWD and discovers new SWDs. First, it verifies if a document is a SWD or not, and it also revisits discovered URLs to check updates. Secondly, several heuristics are used to discover new SWDs through semantic relations: (1) a URIref is highly likely to be the URL of an SWD; (2) owl:imports links to an external ontology, which is a SWD; (3) the semantics of the FOAF ontology show that the rdfs:seeAlso property of an instance of foaf:Person often links to another FOAF document, which often is a SWD.

5. SWD METADATA

SWD metadata is collected to make SWD search more efficient and effective. It is derived from the content of SWDs as well as the relations among SWDs. Swoogle identifies three categories of metadata: (i) basic metadata, which considers the syntactic and semantic features of a SWD, (ii) relations, which consider the explicit semantics between individual SWDs, and (iii) analytical results such as SWO/SWDB classification, and SWD ranking. The first two categories are objective information about SWDs, and we will discuss them in the rest of this section. The third category is subjective and will be discussed in section 6.

In order to simplify notations, we use qualified names (QNames [7]) in the following context. E.g. "rdf:" stands for the RDF namespace, "daml:" stands for the DAML namespace, etc.

5.1 Basic metadata

The basic metadata about a SWD falls into three categories: language feature, RDF statistics and ontology annotation.

Language feature refers to the properties describing the syntactic or semantic features of a SWD. Swoogle captures the following features:

1. *Encoding* shows the syntactic encoding of a SWD. There are three existing encodings, namely "RDF/XML", "N-TRIPLE" and "N3".

2. *Language* shows the Semantic Web language used by a SWD. Swoogle considers four meta level languages, namely "OWL", "DAML", "RDFS", and "RDF".

3. *OWL Species* shows the language species of a SWD written in OWL. There are three possible species, namely "OWL-LITE", "OWL-DL", and "OWL-FULL".

RDF statistics refers to the properties summarizing node distribution of the RDF graph of a SWD. We focus on how SWDs define new classes, properties and individuals. In an RDF graph, a node is recognized as a *class* iff it is not an anonymous node and it is an instance of rdfs:Class; similarly, a node is a *property* iff it is not an anonymous node and it is an instance of rdf:Property; an *individual* is a node which is an instance of any user defined class.

[6] http://jena.sourceforge.net/

[7] http://www.w3.org/TR/1999/REC-xml-names-19990114/#NT-QName

Let foo be a SWD. By parsing foo into an RDF graph, we may get RDF statistics about foo. Let $C(foo), P(foo), I(foo)$ be the set of classes, properties and individuals defined in the SWD foo respectively. The *ontology-ratio* $R(foo)$ is calculated by Equation (1). The value of ontology-ratio ranges from 0 to 1, where "0" implies that foo is a pure SWDB and "1" implies that foo is a pure SWO.

$$R(foo) = \frac{|C(foo)| + |P(foo)|}{|C(foo)| + |P(foo)| + |I(foo)|} \tag{1}$$

Ontology annotation refers to the properties that describe a SWD as an ontology. In practice, when a SWD has an instance of OWL:Ontology, Swoogle records its properties as the following:

1. *label.* i.e. rdfs:label
2. *comment.* i.e. rdfs:comment
3. *versionInfo.* i.e. owl:versionInfo and daml:versionInfo

5.2 Relations among SWDs

Looking at the entire semantic web, it is hard to capture and analyze relations at the RDF node level. Therefore, Swoogle focuses on SWD level relations which generalize RDF node level relations. Swoogle captures the following SWD level relations:

TM/IN captures *term reference* relations between two SWDs, i.e. a SWD is using terms defined by some other SWDs. By retrieving and processing the reference SWD, the type of term (class, property or individual) can be determined. The referenced SWDs are collected by recording the namespaces of all valid URIrefs in the given SWD.

IM shows that an ontology *imports* another ontology. The URLs of referenced ontolgoies are collected by recording the objects in triples whose predicate is owl:imports or daml:imports.

EX shows that an ontology *extends* another. Such a relation may be produced by many properties as shown in Table 3. For example, if ontology A defines class AC which has the "rdfs:subClassOf" relation with class BC defined in ontology B, Swoogle will record the EX relation from A to B.

PV shows that an ontology *is a prior version of* another.

CPV shows that an ontology *is a prior version of and is compatible with* another.

IPV shows that an ontology *is a prior version of but is incompatible with* another.

The last five relations are types of *inter-ontology relation.* They are extracted from each SWD by analyzing triples containing "indicators" listed in Table 3.

6. RANKING SWDS

PageRank, introduced by Google [18, 12], evaluates the relative importance of web documents. Given a document A, A's PageRank is computed by Equation 2:

$$\begin{aligned} &PR(A) = PR_{direct}(A) + PR_{link}(A) \\ &PR_{direct}(A) = (1-d) \\ &PR_{link}(A) = \mathrm{d}\left(\frac{\mathrm{PR(T_1)}}{\mathrm{C(T_1)}} + ... + \frac{\mathrm{PR(T_n)}}{\mathrm{C(T_n)}}\right) \end{aligned} \tag{2}$$

where $T_1, \ldots, T_n$ are web documents that link to A; $C(T_i)$ is the total outlinks of T_i; and d is a damping factor, which is typically set to 0.85. The intuition of PageRank is to measure the probability that a random surfer will visit a page. Equation 2 captures the probability that a user will arrive at a given page either by directly addressing it (via $PR_{direct}(A)$), or by following one of the links pointing to it (via $PR_{link}(A)$).

Unfortunately, this random surfing model is not appropriate for the Semantic Web. The semantics of links lead to a non-uniform probability of following a particular outgoing link. Therefore, Swoogle uses a *rational random surfing model* which accounts for the various types of links that can exist between SWDs.

Given SWDs A and B, Swoogle classifies inter-SWD links into four categories: (i) imports(A,B), A imports all content of B; (ii) uses-term(A,B), A uses some of terms defined by B without importing B; (iii) extends(A,B), A extends the definitions of terms defined by B; and (iv) asserts(A,B), A makes assertions about the individuals defined by B.

These relations should be treated differently. For instance, when a surfer observes imports(A,B) while visiting A, it is natural for it to follow this link because B is semantically part of A. Similarly, the surfer may follow the extends(A,B) relation because it can understand the defined term completely only when it browses both A and B. Therefore, we assign different weights to the four categories of inter-SWD relations.

Since we generalized RDF node level relations to SWD level relations, we also count the number of references. The more terms in B referenced by A, the more likely a surfer will follow the link from A to B.

Based on the above considerations, given SWD a, Swoogle computes its raw rank using Equation 3.

$$\begin{aligned} &rawPR(a) = (1-d) + d \sum_{x \in L(a)} rawPR(x) \tfrac{f(x,a)}{f(x)} \\ &f(x,a) = \sum_{l \in links(x,a)} weight(l) \\ &f(x) = \sum_{a \in T(x)} f(x,a) \end{aligned} \tag{3}$$

where $L(a)$ is the set of SWDs that link to a, and $T(x)$ is the set of SWDs that x links to.

Type	**Classes and Properties**
IM	owl:imports, daml:imports
EX	rdfs:subClassOf, rdfs:subPropertyOf, owl:disjointWith, owl:equivalentClass, owl:equivalentProperty, owl:complementOf, owl:inverseOf, owl:intersectionOf, owl:unionOf daml:sameClassAs, daml:samePropertyAs, daml:inverseOf, daml:disjoinWith daml:complementOf, daml:unionOf daml:disjointUnionOf, daml:intersectionOf
PV	owl:priorVersion
CPV	owl:DeprecatedProperty, owl:DeprecatedClass, owl:backwardCompatibleWith
IPV	owl:incompatibleWith

Table 3: Indicators of inter-ontology relation

Then Swoogle computes the rank for SWDBs and SWOs using Equations 4 and 5 respectively.

$$PR_{SWDB}(a) = rawPR(a) \tag{4}$$

$$PR_{SWO}(a) = \sum_{x \in TC(a)} rawPR(x) \tag{5}$$

where TC(a) is the transitive closure of SWOs imported by a.

Our hypothetical *Rational Random Surfer*(RRS) retains PageRank's direct visit component; the rational surfer can jump to SWDs directly with a certain probability d. However, in the link-following component, the link is chosen with unequal probability – $\frac{f(x,a)}{f(x)}$, where x is the current SWDB, a is the SWD that x links to, $f(x, a)$ is the sum of all link weights from x to a, and $f(x)$ is the sum of the weights of all outlinks from x. The control flow of such a surfer is is shown in Figure 2.

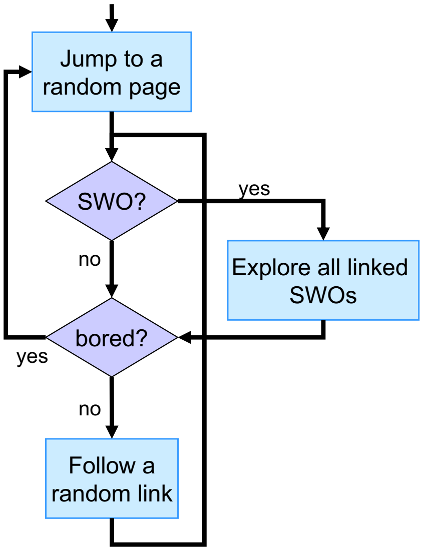


Figure 2: Rational Random Surfer

Figure 3 illustrate how the rank of a SWO is computed. Let A, B, C, D, E and F be SWOs, and assume that the probability for a RRS to visit any of these SWOs from a SWDB is 0.0001. The probability that she visits B is $PR_{SWO}(B) + (PR_{SWO}(E) + PR_{SWO}(D)) = 0.0003$. The probability that she visits F is $PR_{SWO}(F) + PR_{SWO}(C) = 0.0002$.The probability that she visits A is 0.0006 since A will be visited when any of {B, C, D, E, F} is visited.

Table 4 shows the top ranked SWDs discovered by Swoogle.

7. INDEXING AND RETRIEVAL OF SWDS

Central to a Semantic Web search engine is the problem of indexing and searching SWDs. While there is significant semantic information encoded in marked-up documents, reasoning over large collections of documents can be expensive. Traditional information retrieval techniques have the advantage of being faster, while taking a somewhat more coarse view of the text. They can thus quickly retrieve a set of SWDs that deal with a topic based on similarity of the source text alone.

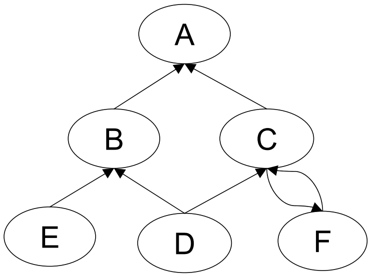


Figure 3: Combine ranks of SWO

In addition to efficiency, there are a number of reasons why one would want to apply IR techniques to this problem. For one thing, documents are not entirely markup. We would like to be able to apply search to both the structured and unstructured components of a document. Related to this point, it is conceivable that there will be some text documents that contain embedded markup. In addition, we may want to make our documents available to commonly used search engines, such as Google. This implies that the documents must be transformed into a form that a standard information retrieval engine can understand and manipulate. Information retrieval techniques also have some value characteristics, including well researched methods for ranking matches, computing similarity between documents, and employing relevance feedback. These compliment and extend the retrieval functions inherent in Swoogle.

There has been work [20, 17, 10] demonstrating that such techniques can be made to work with both RDF dcuments as well as text documents with embedded RDF markup, and that they can be made to leverage some of the semantic information encoded.

Traditional IR techniques look at a document as either a collection of tokens, typically words or N-Grams. An N-Gram is an n-character segment of the text which spans inter-word boundaries. The N-Gram approach is typically employed by sliding a window of n-characters along the text, and taking a sample at each one character step. The use of N-Grams can result in a larger vocabulary, as single words can contain multiple N-Grams. One advantage to this approach is that inter-word relationships are preserved, where they are typically not in word based approaches. N-Grams are also known to be somewhat resistant to certain kinds of errors, such as mis-spellings.

The use of N-Gram is particularly important to this approach because of the treatment of URIrefs as terms. Given a set of keywords defining a search, we may want to match documents that have URIrefs containing those keywords. For example, consider a search for ontologies for "time". The search keywords might be *time temporal interval point before after during day month year eventually calendar clock durations end begin zone*. Candidate matches might include documents containing URIrefs such as:

```
http://foo.com/timeont.owl#timeInterval
http://foo.com/timeont.owl#CalendarClockInterval
http://purl.org/upper/temporal/t13.owl#timeThing
```

Clearly, exact matching based on words only would miss these documents (based on the URIrefs given). However, N-Grams would find a number of matches.

Rank	URL	Value
1	http://www.w3.org/1999/02/22-rdf-syntax-ns	2845.97
2	http://www.w3.org/2000/01/rdf-schema	2814.21
3	http://www.daml.org/2001/03/daml+oil	311.65
4	http://www.w3.org/2002/07/owl	192.18
5	http://www.w3.org/2000/10/rdf-tests/rdfcore/testSchema	59.82
6	http://www.w3.org/2002/03owlt/testOntology	22.50
7	http://ontology.ihmc.us/Entity.owl	21.28
8	http://www.w3.org/2001/XMLSchema	17.45
9	http://www.daml.org/2000/12/daml+oil	10.44
10	http://www.daml.org/2000/10/daml-ont	8.88
11	http://ontology.ihmc.us/Group.owl	7.67
12	http://ontology.ihmc.us/Actor.owl	6.83

Table 4: Top 12 ranked SWDs (May 25,2004)

It is also possible, however, to use a word based approach. Bear in mind that ontologies define vocabularies. In OWL, URIrefs of classes, properties and individuals play the role of words in a natural language. We can take an SWD, reduce it to triples, extract the URIrefs (with duplicates), discard URIrefs for blank nodes, hash each URIref to a token, and index the document. Whereas a conventional information retrieval system treats a text document as a *bag of words*, we could treat a SWD as a *bag of URIrefs*. This would support retrieval using queries which are also sets of URIrefs as well as other functions, such as document similarity metrics.

We are currently adapting the Sire, a custom indexing and retrieval engine we built for the Carrot2 distributed IR system [7], to augment Swoogle. Sire can be made to use either n-grams or words, and employs a TF/IDF model with a standard cosine similarity metric. Sire is being enhansed to process RDF docouments using either character level n-grams computed over the RDF source or to process the URIrefs in the document as indexible tokens.

8. CURRENT STATUS

Swoogle is an ongoing project undergoing constant development. This paper primarily describes the features of version 1. Figure 4 shows the Swoogle start page.

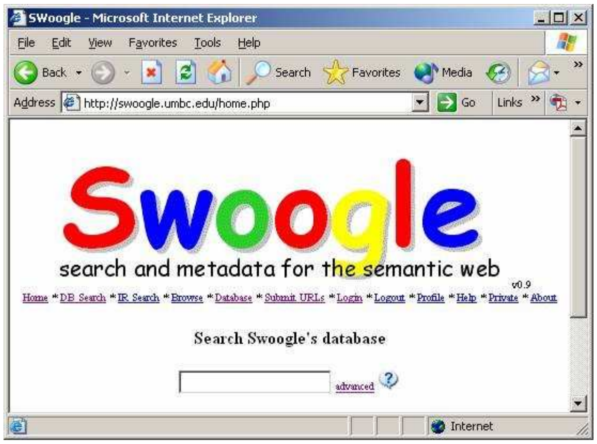

SWoogle - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://swoogle.umbc.edu/home.php Go Links
Swoogle
search and metadata for the semantic web
v0.9
Home * DB Search * IR Search * Browse * Database * Submit URLs * Login * Logout * Profile * Help * Private * About
Search Swoogle's database
advanced
Internet

Figure 4: Swoogle interface

A general user can query with keywords, and the SWDs matching those keywords will be returned in ranked order.

As shown in Table 4, Swoogle ranks SWOs higher than SWDBs; thus, SWOs will be returned as query results before SWDBs. The highest ranked SWDs typically are the base ontologies that define the Semantic Web languages, which are used by almost all SWDs and are always imported by SWOs.

For advanced users, an "advanced" search interface is provided (Figure 5), which essentially allows them to fill in the constraints to a general SQL query on the underlying database.

The user can query using keywords, content based constraints (e.g. type of SWD, OWL syntax, number of defined classes/properties/individuals), language and encoding based constraints (N3 vs XML), and/or the Rank of the document. Sample results are shown in Figure 6.

At present, the metadata are stored in a mySQL database, and we have indexed about 135, 000 SWDs. We anticipate that in future versions, we will need to migrate to a commercial database in order to scale to indexing millions of SWDs.

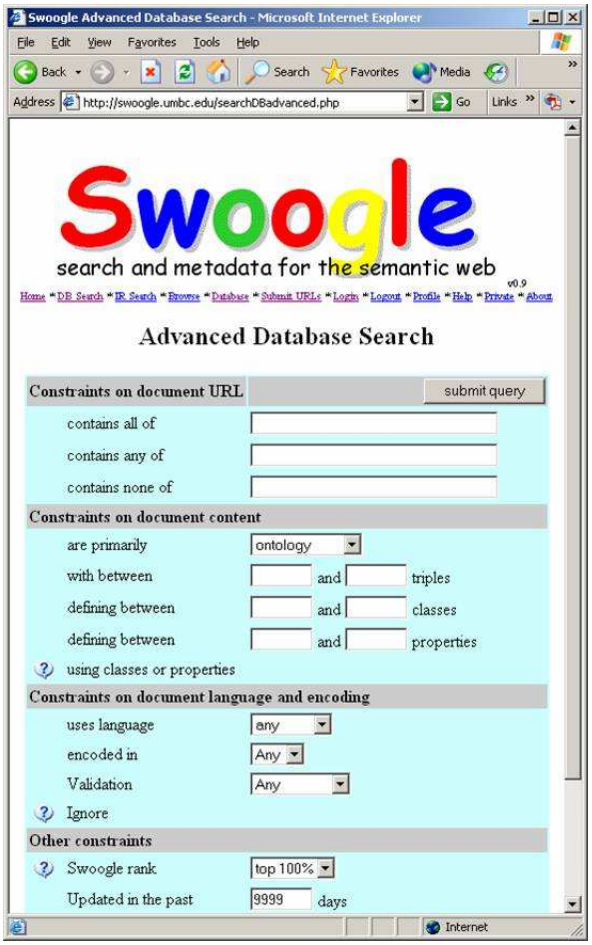
In our database, 13.29% SWDs are classified as SWOs, and others as SWDBs (with 0.8 as the ontology ratio threshold). We find that about half of all SWDs have rank of 0.15, which means they are not referred to by any other SWDs. Also, the mean of ranks is 0.8376, which implies that the SWDs we have crawled are poorly connected.

We are completing are transition to Swoogle2, which featured three components: Swoogle Search, Ontology Dictionary and statistical measures of the collection of SWDs. We invite readers to visit Swoogle at http://swoogle.umbc.edu.

9. CONCLUSIONS AND FUTURE WORK

Powerful search and indexing systems are needed by Semantic Web developers and researchers to help them find and analyze SWDs. Such systems can also be used to support tools (e.g. markup editors), as well as software agents whose knowledge comes from the Semantic Web.

Current web search engines such as Google and AlltheWeb do not work well with SWDs, as they are designed to work with natural languages and expect documents to contain unstructured text composed of words. Failing to understand the structure and semantics of SWDs, they do not take advantage of them.


Swoogle Advanced Database Search - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://swoogle.umbc.edu/searchDBadvanced.php
Go Links
Swoogle
search and metadata for the semantic web
v0.9
Home * DB Search * IR Search * Browse * Database * Submit URLs * Login * Logout * Profile * Help * Private * About
Internet

Advanced Database Search

Constraints on document URL	submit query
contains all of	
contains any of	
contains none of	
Constraints on document content	
are primarily	ontology
with between	and triples
defining between	and classes
defining between	and properties
using classes or properties	
Constraints on document language and encoding	
uses language	any
encoded in	Any
Validation	Any
Ignore	
Other constraints	
Swoogle rank	top 100%
Updated in the past	9999 days

Figure 5: Swoogle advanced query

We have described a prototype crawler-based indexing and retrieval system for Semantic Web documents. It runs multiple crawlers to discover SWDs through meta-search and link-following; analyzes SWDs and produces metadata about SWDs and the relations among SWDs; computes ranks of SWDs using a "rational random surfing model"; and indexs SWDs using an information retrieval system by treating URIrefs as terms. One of the interesting properties computed for each semantic web document is its ontology rank – a measure of the document's importance on the Semantic Web.

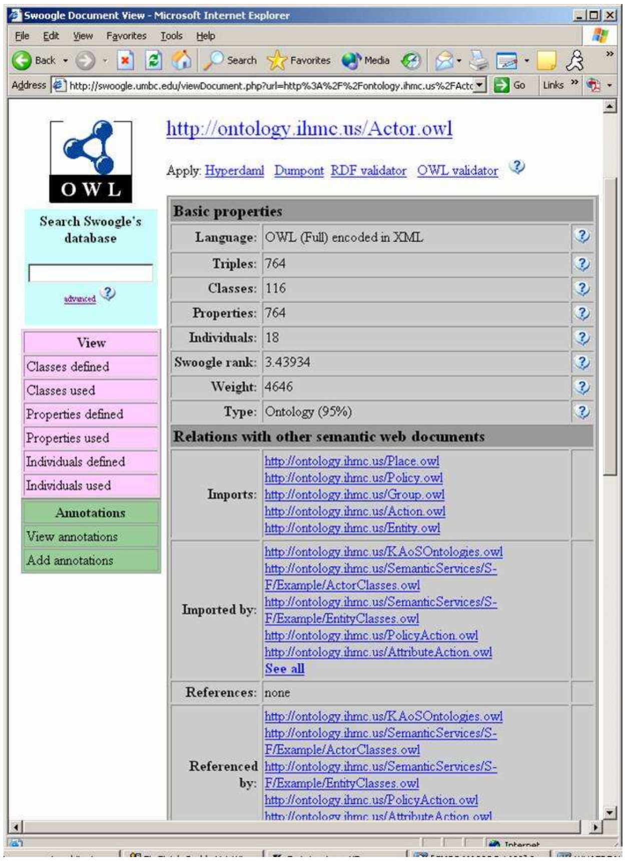
The current version of our system (Swoogle2) has discovered and analyzed over 137, 000 semantic web documents with 3,900,000 triples. It has been designed and partially implemented to capture more metadata on classes and properties and to support millions of documents. We have also built an ontology dictionary based on the ontologies discovered by Swoogle, which we continue to refine.

10. REFERENCES

[1] http://www.daml.org/ontologies/, daml ontology library, by daml.

[2] http://www.schemaweb.info/, schema web.

[3] http://www.semanticwebsearch.com/, semantic web search, by intellidimension.


Swoogle Document View - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://swoogle.umbc.edu/viewDocument.php?url=http%3A%2F%2Fontology.ihmc.us%2FActo
Go Links
OWL
Internet

http://ontology.ihmc.us/Actor.owl

Apply: Hyperdaml Dumpont RDF validator OWL validator

Search Swoogle's database

advanced

View
Classes defined
Classes used
Properties defined
Properties used
Individuals defined
Individuals used

Annotations
View annotations
Add annotations

Basic properties	
Language:	OWL (Full) encoded in XML
Triples:	764
Classes:	116
Properties:	764
Individuals:	18
Swoogle rank:	3.43934
Weight:	4646
Type:	Ontology (95%)
Relations with other semantic web documents	
Imports:	http://ontology.ihmc.us/Place.owl http://ontology.ihmc.us/Policy.owl http://ontology.ihmc.us/Group.owl http://ontology.ihmc.us/Action.owl http://ontology.ihmc.us/Entity.owl
Imported by:	http://ontology.ihmc.us/KAoSOntologies.owl http://ontology.ihmc.us/SemanticServices/S-F/Example/ActorClasses.owl http://ontology.ihmc.us/SemanticServices/S-F/Example/EntityClasses.owl http://ontology.ihmc.us/PolicyAction.owl http://ontology.ihmc.us/AttributeAction.owl See all
References:	none
Referenced by:	http://ontology.ihmc.us/KAoSOntologies.owl http://ontology.ihmc.us/SemanticServices/S-F/Example/ActorClasses.owl http://ontology.ihmc.us/SemanticServices/S-F/Example/EntityClasses.owl http://ontology.ihmc.us/PolicyAction.owl http://ontology.ihmc.us/AttributeAction.owl

Figure 6: Swoogle query result

[4] http://www.semwebcentral.org/, semwebcentral, by infoether and bbn.

[5] http://www.w3.org/2004/ontaria/, ontaria, by w3c.

[6] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.

[7] R. S. Cost, S. Kallurkar, H. Majithia, C. Nicholas, and Y. Shi. Integrating distributed information sources with carrot ii. In *Proceedings of the 6th International Workshop on Cooperative Information Agents VI*, pages 194–201. Springer-Verlag, 2002.

[8] J. Davies, R. Weeks, and U. Krohn. Quizrdf: search technology for thesemantic web. In *WWW2002 workshop on RDF and Semantic Web Applications, 11th International WWW Conference (WWW11)*, 2002.

[9] S. Decker, M. Erdmann, D. Fensel, and R. Studer. Ontobroker: Ontology based access to distributed and semi-structured information. In *DS-8*, pages 351–369, 1999.

[10] T. Finin, J. Mayfield, C. Fink, A. Joshi, and R. S. Cost. Information retrieval and the semantic web, January 2004.

[11] S. Handschuh and S. Staab. Cream: Creating metadata for the semantic web. *Comput. Networks*, 42(5):579–598, 2003.

[12] T. Haveliwala. Efficient computation of pageRank. Technical Report 1999-31, 1999.

[13] I. Horrocks. DAML+OIL: A description logic for the semantic web. *IEEE Data Engineering Bulletin*, 25(1):4–9, 2002.

[14] M. Lifantsev. Rank computation methods for Web documents. Technical Report TR-76, ECSL, Department of Computer Science, SUNY at Stony Brook, Stony Brook, NY, November 1999.

[15] S. Luke, L. Spector, D. Rager, and J. Hendler. Ontology-based web agents. In *Proceedings of the First International Conference on Autonomous Agents (Agents97)*, pages 59–66, 1997.

[16] P. Martin and P. Eklund. Embedding knowledge in web documents. In *Proceedings of the 8th International World Wide Web Conference (WWW8)*, pages 324–341, 1999.

[17] J. Mayfield and T. Finin. Information retrieval on the semantic web: Integrating inference and retrieval. In *Proceedings of the SIGIR 2003 Semantic Web Workshop*, 2003.

[18] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

[19] I. Rogers. The google pagerank algorithm and how it works. http://www.iprcom.com/papers/pagerank/, May 2002.

[20] U. Shah, T. Finin, and A. Joshi. Information retrieval on the semantic web. In *Proceedings of the 11th international conference on Information and knowledge management*, pages 461–468, 2002.