Efficient Proposed Framework for Semantic Search Engine using New Semantic Ranking Algorithm

M.M.El-gayar[1]
P.H.D Student,
Information Technology Department,
Faculty of Computer and Information Science,
Mansoura, Egypt

N.Mekky[2]
Assistant Professor,
Information Technology Department,
Faculty of Computer and Information Science,
Mansoura, Egypt

A. Atwan[3]
Associate Professor,
Information Technology Department,
Faculty of Computer and Information Science,
Mansoura, Egypt

***Abstract*—The amount of information raises billions of databases every year and there is an urgent need to search for that information by a specialize tool called search engine. There are many of search engines available today, but the main challenge in these search engines is that most of them cannot retrieve meaningful information intelligently. The semantic web technology is a solution that keeps data in a readable format that helps machines to match smartly this data with related information based on meanings. In this paper, we will introduce a proposed semantic framework that includes four phases crawling, indexing, ranking and retrieval phase. This semantic framework operates over a sorting RDF by using efficient proposed ranking algorithm and enhanced crawling algorithm. The enhanced crawling algorithm crawls relevant forum content from the web with minimal overhead. The proposed ranking algorithm is produced to order and evaluate similar meaningful data in order to make the retrieval process becomes faster, easier and more accurate. We applied our work on a standard database and achieved 99 percent effectiveness on semantic performance in minimum time and less than 1 percent error rate compared with the other semantic systems.**

Keywords—Semantic Search Engine; Ontology; Semantic Ranker; Crawler; RDF;SPARQL

I. Introduction

There is a huge amount of data stored on the Internet that is only useful and helpful if accessed as information, not as pure data. To access information from Internet, we need a ‘smart or intelligent’ search facility. Search engines are the tools to help users to find data from the huge warehouse of web pages. To extract data, most of the search engines use syntax-based search or full-text search methods. Full-text searching is a technique whereby a computer program matches terms in a search query with terms embedded within individual documents in a database [1]. An important issue in full-text searching technique is that “Because full-text searching relies on linguistic matching—matching a word or phrase in a search query with the same word or phrase in a document in the database being searched—it is subject to failure when a variant term exists and is not matched” [2].

Recent syntax based search engines use various techniques to solve the limitations of a syntax-based search such as page ranking and content score [3]. To get web pages ranked by the search engines, website developers use a method called Search Engine Optimization (SEO). Keywords and meta-tags are the main tools used for SEO. These methods enhance the factor of user friendliness and increase the chances of more accurate results, but these are not the ultimate solution. Data searched by a syntax-based search engine has some limitations, including high recall with low precision (e.g. thousands of results in response to one or few keywords).

A semantic web is optimized solution to these challenges. Semantic Web can be defined as some documents linked in such a way so that the data becomes readable and understandable in a meaningful way [4]. One way of viewing this semantic web is that it is a concept of utilizing the Internet in such a way that searching the World Wide Web returns results relevant to the meaning of the search query. On the Semantic information is illustrated via a new W3C model called the Resource Description Framework (RDF). Semantic Search system is a search system for the Semantic Web. Existing Web sites can be utilized by both individuals and computers to trace exactly and gather information available on the Semantic Web. Ontology is the most significant conception used in the semantic web infrastructure, and RDF(S) (Resource Description Framework/Schema) and Web Ontology Languages (OWL) that used to represent ontologies.

In recent years, the Resource Description Framework (RDF) has become a popular protocol for storing web-based data with well-defined meanings,that usedto link data to improve semantic meaning has widened the scope of this protocol. While RDF data is routinely used by many organizations (e.g. gov.uk and bbc.co.uk) its potential to improve semantic searches is now of interest to the database and Internet research communities.[5]

The Semantic Web will maintain more professional discovery, computerization and reuse of data and offer some support for combinational problem that cannot be solved with existing web techniques. At this time of research on semantic web search systems are not in the beginning stage, but unlike the traditional search systems such as Google, Yahoo, and Bing (MSN) and so forth still lead the present markets of search systems.

In this manuscript, we suggest a new semantic ranking algorithm based on HTML parsing and crawling algorithm to handle the search engine challenges discusses in Section-3. Experimental outcomes display that the recommended technique is more efficient to retrieve and sort a large amount of data from huge datasets or data warehouses. The manuscript is categorized as follows Section two (2) discuses related work of semantic systems, and Section three (3)discuses the global challenges that face search engines. Section four (4) described the several component of the recommended framework and recommended ranking algorithm. In Sect. 5, experiments results and analysis will be presented and in the last section conclusions and references will be given.

II. Related Works

Information recovery and retrieval by searching on the web is not a fresh idea but has different problems when it is evaluated to general information retrieval. Dissimilar search systems return different search results due to the differences in indexing and searching process. Google, Yahoo, and Bing have been out there which holds the queries after developing the keywords. They only search information given on the web page, recently, some research group's start distributing results from their semantics-based search engines. Many novel search engines have been developed for the data Web. Most of these systems are focused on RDF document search like (d'Aquin, Baldassarre, Gridinoc, Sabou, Angeletou, & Motta, 2007; Oren, Delbru, Catasta, Cyganiak, Stenzhorn, &Tummarello, 2008) or ontology search like (Ding, Pan, Finin, Joshi, Peng, &Kolari, 2005). Recall that an RDF document serializes an RDF graph; an ontology, as a schema on the data Web, defines classes and properties for describing objects. Although both RDF document search and ontology search are essential for application developers, they can hardly serve ordinary Web users directly. Instead, object-level search is in demand and dominates all other Web queries (Pound, Mika, & Zaragoza, 2010).

Swooglesystem that described in W3Cand Duckduckgohavesome limitations especially regarding user experience, time of query response and storage capacity. As shown in figure 1, Swoogle's architecture can be broken into four major components: SWD discovery, metadata creation, data analysis, and interface.

Swoogle architecture is data centric and extensible: different components work on different tasks independently. Swoogle offers the some services such as search SW terms and documents, i.e. URIs that have been defined as classes and properties and Provide metadata of SW documents and support browsing the Semantic Web. But, Swoogle has some limitations such as poor indexing of documents and long response time of query.[6]

Another example of semantic search engine is Semantic Web Search Engine (SWSE). Following traditional search engine structural design, SWSE contains of crawling, improving data, indexing process and an interface for search to retrieve an information; unlike traditional search engines, SWSE works over RDF Web data tightly also known as Linked Data which implies unique challenges for the system design, architecture, algorithms, implementation and user interface. [7]

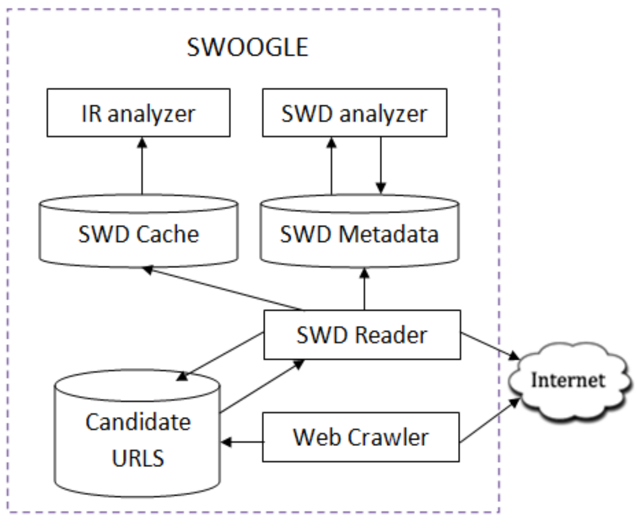


Fig. 1. Swoogle architecture

The sophisticated system design of SWSE loosely follows that of conventional HTML search engines. Figure 2 details the pre-runtime architecture of SWSE system, viewing the mechanism involved in realizing a local index of RDF Web data agreeable for search. Like usual search systems, SWSE includes modules for crawling, ranking and indexing data; on the other hand, there are also factorsspecially designed for treatment RDF data, namely the consolidation module and the reasoning module. The high-level index building process is as follows:

- The crawler recognizes a set of seed URIs. Results analysis for keyword query Bill Clinton Fig. 2. Focus analysis for entity Bill Clinton recover a large set of RDF data from the Web;
- The consolidation module tries to and identical (i.e., equivalent) identifiers in the data
- The ranking module achieves links-based analysis over the crawled data and gains scores indicating the significance of individual factors in the data (the ranking module also considers URI redirections encountered by the crawler when performing the links-based analysis);
- The reasoning modulematerializes new data which is implied by the natural semantics of the input data (the reasoning module also requires URI redirection information to assess the trustworthiness of sources of data);
- The indexing module organizes an index which supports the information retrieval tasks required by the user interface.

But, SWSE has some limitations such as poor ranking of documents because the ranking stage is coming before the indexing stage. Ranking technique is coming independently with data indexed in dataset.

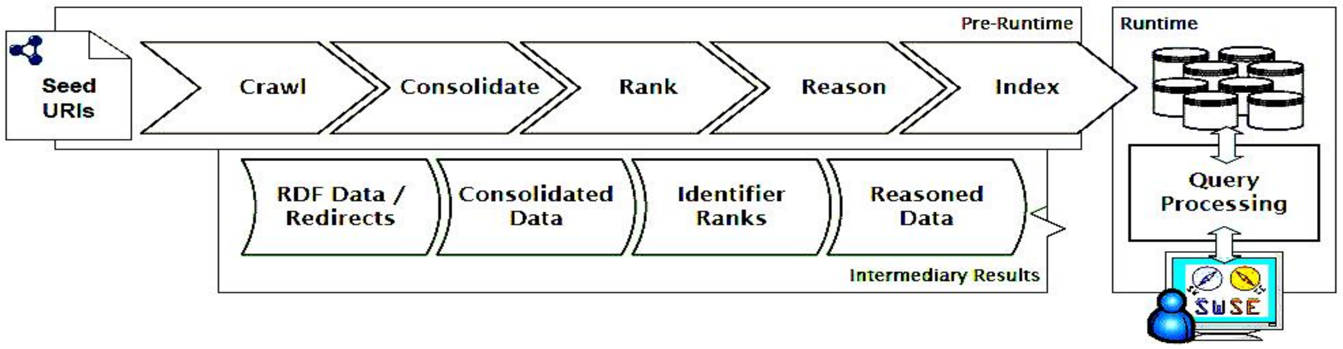


Fig. 2. SWSE Architecture

Another solution model of semantic search engine called Falcons Object Search [8] which firstly is a keyword-based object search engine. For each discovered object, the system constructs an extensive virtual document consisting of textual descriptions extracted from its concise RDF description. Then an inverted index is built from terms in virtual documents to objects for supporting basic keyword-based search. That is, when a keyword query arrives, based on the inverted index, the system matches the terms in the query with the virtual documents of objects to generate a result set. Unfortunately this model is not interested to rank these objects according to query.

This paper investigates some concepts on how the semantic web might be queried in the context of semantic search engines and proposes a framework that facilitates an effective search over the semantic web. Firstly the various factors that influence the search experience over the Internet will be reviewed. Secondly the semantic core technologies necessary to perform a basic search over the Internet will be described - that is RDF and RDF Query Language (SPARQL). Thirdly the academic and social impact of this work is clarified. Finally a proposed framework for a complete search experience for a semantic search engine is presented.

III. CHALLENGES FOR SEMANTIC WEB SEARCH ENGINE

A semantic web search engine should be able to search data over the Internet with maximum precision and accuracy and should be able to link related data. Semantic search engine should consider the following criteria: user experience, efficiency (performance and associated time) , ranking process, scalability, and cost effectiveness.

A. *User Experience*

A friendly user interface is the mainly significant feature that will increase the user experience. Search engines such as Yahoo, Bing and especially google have all been through a number of enhancements in order to give end-users with the best potential user experience. Even if the results are incomplete or sometimes not accurate due to syntax-only based search algorithms, end-users still remain to these search engines with good user experience. Enhancements to the end-user interface of a semantic query search engine needs important development so that poor input representation of a query will automatically suggest corrections for spelling mistakes and poor grammar, and of course find the best matched results with a high accuracy.

B. *Efficiency*

An efficient semantic search engine’s performance depends upon the size of data to be matched, the request time to server or database and the associated response time. For a semantic web query the finishing time also depends on factors such as delays caused by looking up URL (Uniform Resource Locators) [9], indexing large-scale of data [5], and dealing with query termination and broken links problems[10]. Some smart semantic search systems cannot illustrate their important performance in developing precision and lowering recall. In Ding’s semantic flash system, the source of the search system is based on the top-50 returned results from Google that is not a semantic search engine, which could be low precision and high recall [11].

C. *Ranking Process*

The main idea of the Semantic web search engine is to retrieve the most relevant (most precision) and accurate results in response to a query. Ranking process such as page rank algorithm is a method of rating web pages so that the web pages with the highest ranking are presented at the top of a list of search results [12]. This is a challenging task given that there are “more than 12.3 billion web pages in the World Wide Web” at the time of writing [13], and a single user query on a search engine may return millions of results. It is consequently critical that the search engine can sort and rank the retrieved documents effectively in order of either relevancy or authenticity. There are a number of techniques used by search engines to rank page results. Page ranking techniques can organize results in order of relevance, significance and content score [13].

D. *Scalability*

Scalability within the perspective of a search engine is the ability of a system to handle a hurriedly growing amount of data. Relational database management systems have frequently shown that they are very efficient with the structure of relational data but not scale well [1]. However scalability for data in a semantic web presents additional challenges because of the open source of the RDF protocol.

E. *Cost Effectiveness*

A high-quality search system must give a solution that is cost effective. Due to the open source of RDF, the queries can be quite expensive while dealing large data sets. For efficient data retrieval, search engines use indexing techniques at the cost of additional storage. As a semantic search engine processes an open structure like RDF, complex queries can be very expensive to process. Only a few solutions have been proposed to solve this issue (e.g. caching process) or use other technique in indexing process. Some efforts have been made to introduce cost effective search algorithms such as the SPARQL as search technique[5]. In section 4 which discuss the proposed framework that handle and overcome these challenges with suitable user interface like Google search engine that overcome user experience. In addition to standard crawler model, indexing algorithm to overcome scalability and cost effectiveness. Finally we introduce the proposed ranking algorithm that considered as main contribution of this paper to overcome ranking and efficiency problems.

IV. Proposed Framework

Proposed Framework is designed in a modular fashion and logically composed of two separate phases, the online phase (retrieving phase - which user deal directly with the server) and offline phase (Crawling, indexing and ranking phase - which server deal not directly with any users). This section describe in details the two phases that described in figure 3 and figure 4.

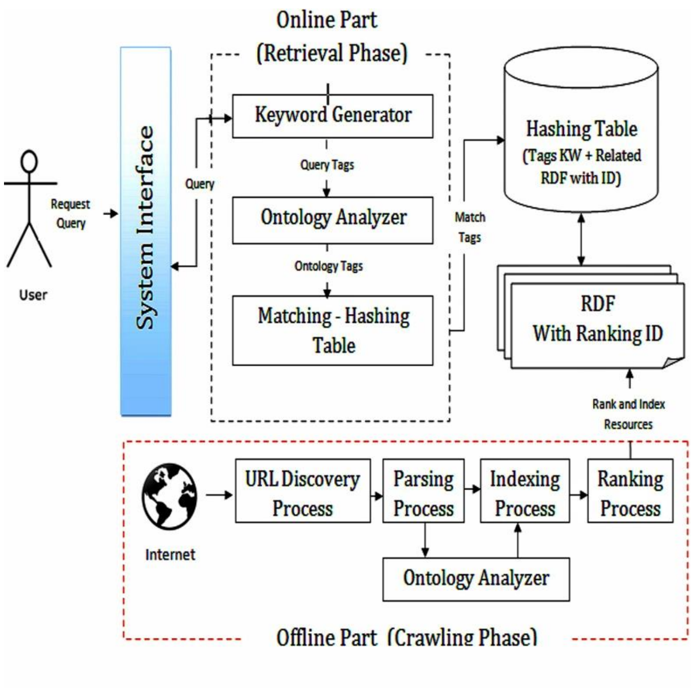


Fig. 3. Logical Architecture of Proposed Framework

A. *Offline Phase (Crawling Phase)*

In this module, three steps are used. The First Step is crawling process that contains two sub-steps, the first sub-step is URL Discovery Process (pre-crawling process) and the second sub-step is officially crawling process. The second step is indexing process that contains also two sub steps, the first sub-step is parsing HTML document and extract useful information, however the second sub-step is officially indexing process. The third step is semantic ranking process using proposed semantic ranking algorithm that discussed in algorithm #2.

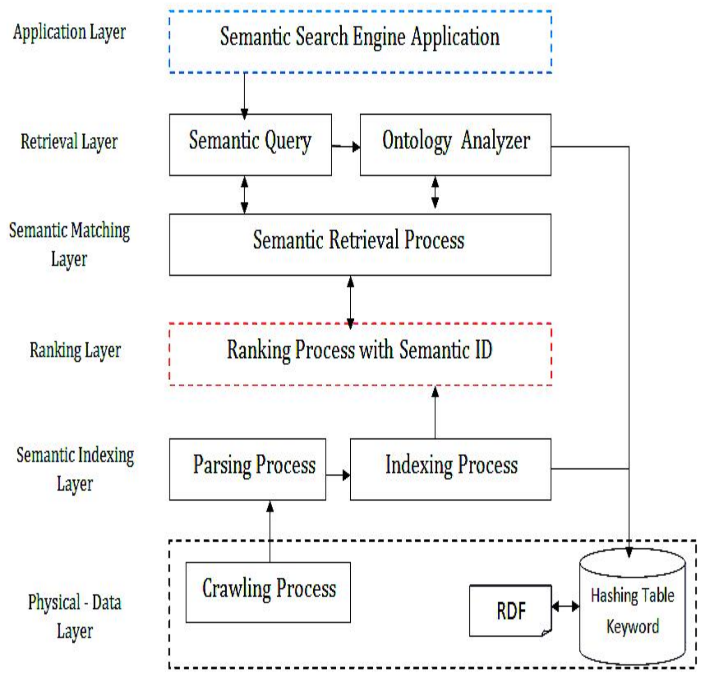


Fig. 4. A Layer Diagram of Proposed Framework

1) *Crawling Process*

The First Step, discover URLs from World Wide Web by using URL Discovery Process and crawl these URLs to find related keywords and meta data about queries with most frequently used in pages with heights page rank algorithm. In crawling module, crawling algorithm is used that discussed in algorithm #1.

```
ALGORITHM#1: Page_Crawling
INPUT : URL
OUTPUT : arrayOfKeywords, Position, Title, URLs
1: Procedure Page_Crawling
2: Begin
3: B_url = getBaseURL (URL)
4: P = Download(URL)
5: Urls = ExtractOutgoingURL in p with B_url
6: ForeachUrls as Url do
7:        Position = ParsePosition(Url)
8:        IF pageTitle is not Null
9:            Title = ExtractPageTitle(Url)
10:       IF pageMetaTags is not null
11:           arrayOfKeywords[Url][Title][Position] =
ExtractPageMetaTags(url)
12:       Else
13:           Continue
14:  End Foreach
15: IF  arrayOfKeywords =  null
16:   Page_Crawling(Urls)
17: End IF
18: End Procedure
```

Algorithm.1. Crawling Algorithm

Since our architecture is currently implemented to index RDF/XML, we would feasibly like to maximizethe ratio of HTTP lookups which result in RDF/XML content; i.e., given the total HTTP lookups as L, and the total number of downloaded RDF/XML pages as R. In order to reduce the amount of HTTP lookupswasted on non-HTML/ RDF/XML content, we implementthe following heuristics:

a) firstly, we blacklist non-http protocol URIs;

b) secondly, we queue URIs with common extensions that are highly unlikely to return RDF/XML/HTML/PDF and we blacklist the extensions of images or videos like (jpg, gif, AVI, MKV, , etc.)

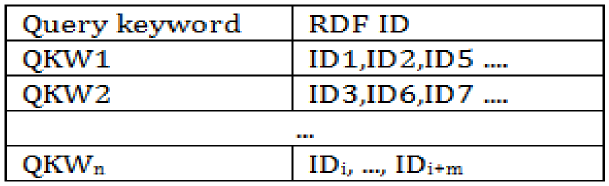
c) thirdly, we check the returned HTTP header and only retrieve the content of URIs reporting Content-type: HTML or application/rdf+xml

2) Indexing Process

The first sub-step in this process is parsing HTML document to extract useful information such as meta data, title, time stamp, author, keywords and related URLs to be crawled again. The second sub-step is indexing results from the crawling process in the first step into database. But with huge data cannot be indexed into relational database because of scalability, storage, sorting, semantic and retrieval issues. Due to semantic, we use RDF instead of relational database, but we face the unstructured problem. So, we can use hybrid dataset from relational and RDF (based on XML). Relational dataset used for storing hash tables (such as table shown in figure 5.a and figure 5.b) that contains keywords and related RDF ID. RDF used for storing all data that extracted from parsing process.

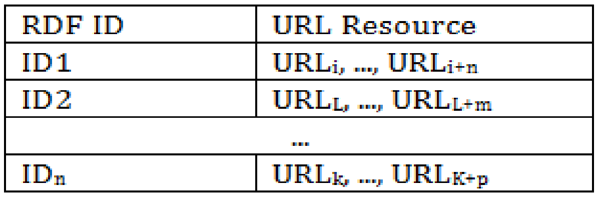
TABLE I. DESCRIPTION OF 8-BITS CHILD DATA OF SEMANTIC ID

Priority	**Title**	**Meta Tag**	**Duplication in text**	**Body URL**	**Header URL**	**Left or Right URL**	**Footer URL**
Priority Of text > 10	If Title exits	If Meta tag exits	If Duplication in text	If URL in bodyof the page	If URL in headerof the page	If URL in left or rightof the page	If URL in footer of the page



Query keyword	RDF ID
QKW1	ID1,ID2,ID5 ....
QKW2	ID3,ID6,ID7 ....
...	
QKW_n	ID_i, ..., ID_{i+m}

Fig. 5. a. Hash table of Query Keyword



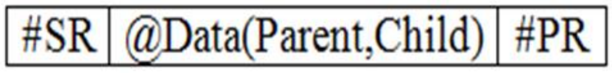
RDF ID	URL Resource
ID1	URL_i, ..., URL_{i+n}
ID2	URL_L, ..., URL_{L+m}
...	
ID_n	URL_k, ..., URL_{K+p}

Fig. 5. b - Hash table of URL Resources

3) Semantic Ranking Process

The idea of developing ontology-based annotations for information is not a fresh idea; semantic search system would consider keyword impression and would return a page only if keywords (or synonyms, homonyms, etc.) are founded within the page and linked to associate concept. The success is measured by the “predictability” that the user would have guessed such anrelationship exists.

The ranking strategy assumes that given a query “Q”, and a page “p”, it is possible to build a query sub-graph $G_{Q,p}$exploiting the information available in page annotation according to ranking ID from the data stored in RDF. Ranking algorithm uses global ID that consist of Semantic ID, Child Data, Parent Data and page rank as shown in figure 6.



#SR	@Data(Parent,Child)	#PR

Fig. 6. 32 bits Semantic ID

Ranking algorithm can rank data according to parse and index step that be implemented in the first phase as the following:

- Child partthat called (URL Resource) consist of(8 bits) 2 digits hexadecimal - with in right 4 bits for position and left 4bits for data as discussed in algorithm#3 (Child_Data_Part_in_Semantic_Rank). This ID expresses the child RDF data from another parent RDF Data.

- Parent Partconsidered as (Semantic Id of parent URL)Semantic ID for parent URL - (16 bits) 4 digits hexadecimal. This ID expresses the parent RDF data of the child RDF data in the above.

- SR (Semantic Rank) in figure 5 used as a header (4 bits) 1 digit hexadecimal as discussed in algorithm#2 (Create_Semantic_ID_of_Child_URL). This ID is global semantic ID that express of all these IDs and can be retrieved faster than others.

- PR (Google Page Rank) that famous ranking algorithm that used in Google according to page that crawled in the first phase. This ID uses values from 0 to 10 - (4 bits) 1 digit hexadecimal

B. *Online Phase (Retrieving Phase)*

In this module as shown in figure 7, three steps are used. The First Step is keyword generator process that used to split the query request into distinct words. The second step is ontology analyzerprocess that help system to recommend tags to query keywords where tags will have associated ontologies. The retrieving of ontologies from an online store or library in order to tag a word is a lengthy process that will have a cost in terms of efficiency. The third step is matching process that used to match the query tags against the keywords stored in hashing table. After matching process, system can detect the related RDF with related URL resources.

TABLE II. DESCRIPTION OF 4 BITS SEMANTIC RANK (SR)

(PR Child + SR Parent) > 15	Relevance feedback > 10	Description >= 4	Position >= 4

ALGORITHM#2: Create_Semantic_ID_of_Child_URL

INPUT : 8bits Child_Data, Child_URL

OUTPUT : Semantic ID

```
1:  Procedure Create_Semantic_ID_of_Child_URL
2:  Begin
3:  Create 4bits Semantic Rank of Child_URL (SR) from step 4 to step 15
4:  Get 4bits of Child_Data from algorithm #3 Child_Data_Part_in_Semantic_Rank
5:  If Right 4bits of Child_Data hexadecimal number greater or equal than 4
6:      Set least significant bit of 4bits Semantic Rank (SR)  to 1
7:  End If
8:  If Left 4bits of Child_Data hexadecimal number greater or equal than 4
9:      Set second bit from least significant bit of 4bits Semantic Rank (SR)  to 1
10:  End If
11: If Relevance feedback of users greater than 10
12:     Set third bit from least significant bit of 4bits Semantic Rank (SR)  to 1
13: End IF
14: If (Semantic Rank of Parent URL + Page Rank of Child URL) greater than 15
15:     Set most  significant bit of 4bits Semantic Rank (SR)  to 1
16: End If
17: Create Semantic ID from step 17 to step
18: most significant 4bits is Semantic Rank
19: Left-Middle 8bits is child_data
20: Right-Middle 16bits is Parent_Semantic_ID
21: Least significant 4bits is page rank number in binary of child URL
22: Return Semantic ID of child URL
23: End Procedure
```

Algorithm.2. Creation of Semantic ID from Child URL

ALGORITHM#3: Child_Data_Part_in_Semantic_Rank

INPUT : Child_URL, arrayOfKeywords, child_Url_position, Page_Title, Ontology_text

```
1:  Procedure Child_Url_Semantic_Rank
2:  Begin
3:  Create Right_four_ bits of child data from step 4 to step 12
4:  If child_Url_position in  Footer_position
5:      Set Right_four_ bits of child data to 0001 binary = 1 in hexadecimal
6:  Else If child_Url_position in  Left_position or Right_position
7:      Set Right_four_ bits of child data to 0010 binary = 2 in hexadecimal
8:  Else If child_Url_position in  Header_position
9:      Set Right_four_ bits of child data to 0100 binary = 4 in hexadecimal
10:  Else If child_Url_position in  Body_position
11:     Set Right_four_ bits of child data to 1000 binary = 8 in hexadecimal
12: End IF
13: Create Left_four_ bits of child data from step 14 to step 25
14: If duplication of keyword in text exists
15:     Set least significant bit in Left_four_ bits of child data to 1
16: End If
17: If arrayOfKeywords not equal null
18:     Set the second bit of least significant bit in Left_four_ bits of child data to 1
19: End If
20: If Page_Title not equal null
21:     Set the third bit of least significant bit in Left_four_   bits of child data to 1
22: End If
23: If arrayOfKeywords exits in Ontology_text
24:     Set most significant bit in Left_four_ bits of child data to 1
25: End If
26: Return 8bits of Child Data part in Semantic ID
27: End Procedure
```

Algorithm.3. Child Data Part in Semantic Rank

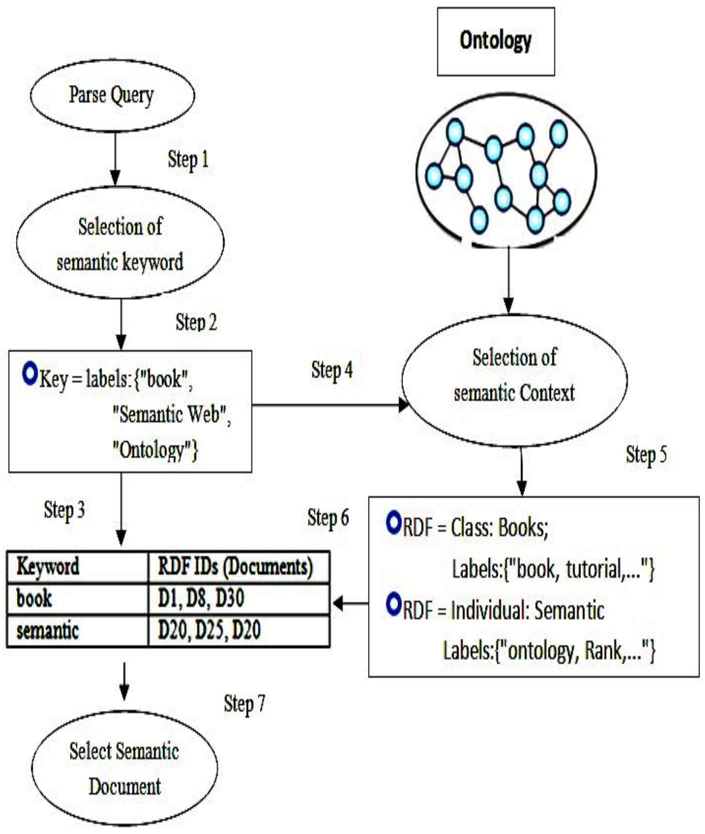

Ontology
Parse Query
Step 1
Selection of semantic keyword
Step 2
Key = labels:{"book", "Semantic Web", "Ontology"}
Step 4
Selection of semantic Context
Step 5
Step 3
Step 6
RDF = Class: Books; Labels:{"book, tutorial,..."}
RDF = Individual: Semantic Labels:{"ontology, Rank,..."}
Keyword | RDF IDs (Documents)
book | D1, D8, D30
semantic | D20, D25, D20
Step 7
Select Semantic Document

Fig. 7. Flow Diagram of Online Retrieval Phase

V. DATA SET AND EXPERIMENT RESULTS

A. *Machine Specifications Used in Testing*

The machine specifications are Corei7 CPU, 2GB RAM, 500GB Hard Disk and Windows 7. The Software specifications are Apache Server (localhost) with PHP version 5.3 and MYSQL Database version 5.5.

B. Data Collection

A standard assessment data gathering should be not influenced towards any exacting system or towards aexact domain, as our objective is to assess general idea entity search over RDF data. Therefore, we needed a collection of documents that would be a realistically large estimation to the amount of RDF data accessible 'live' on the Web and that contained related information for the queries, while concurrently of a size that could be convenient by the resources of a research groups. We chose the 'Billion Triples Challenge' (BTC) 2011 data set, a data-set created for the Semantic Web Challenge in 2011 as displayed in table 3.

TABLE III. BILLION TRIPLE CHALLENGE 2011 DATASET

Description	Billion Triples Challenge
Author	Andreas Harth
Size	20GB gzipped
Download	http://km.aifb.kit.edu/projects/btc-2011/

C. Query sets

The Semantic Search Challenge comprised two tracks. The Entity Search track is identical in nature to the 2010 challenge. However, we created a new set of queries for the entity search task based on the Yahoo! Search Query Tiny Sample v1.0 dataset. We selected 10 queries which name an entity explicitly and may also provide some additional context about it.

D. Proposed System Evaluation

Table 4 is datasheet that describe the result of retrieval process after crawling process of 10 samplesfrom proposed system. Datasheet in table 4 shows summation of total results for each query included relevant result (performance) and irrelevant (error) result. Figure 8 shows the performance and error chart of the retrieval process.

Table 5 is datasheet that describe total time of retrieval process in seconds of 10 samples from proposed system. Figure 9 shows the time chart of the retrieval process.

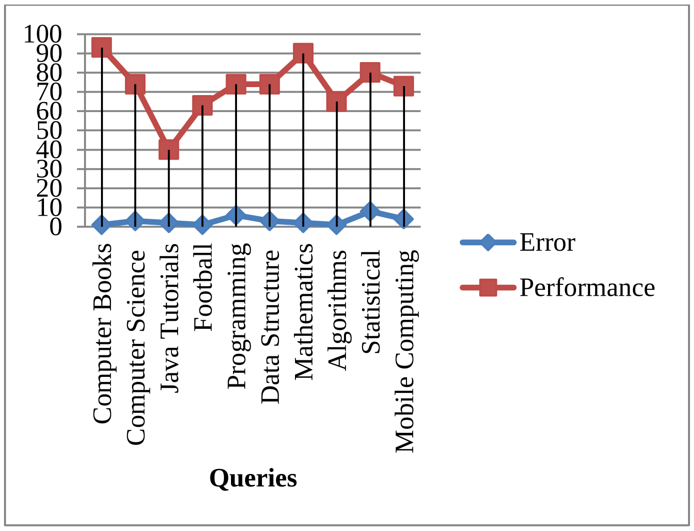


Fig. 8. Performance and Error Chart for Retrieval Process

TABLE IV. RELEVANT AND IRRELEVANT RESULTS OF RETRIEVAL PROCESS AFTER CRAWLING PROCESS ON 10 QUERIES AS SAMPL

10 QueriesSamples	**Crawler Result After ontology Analyzer**	**Retrieval Error (Irrelevant Result)**	**Retrieval Performance (Relevant Result)**
Computer Books	94	1	93
Computer Science	77	3	74
Java Tutorials	42	2	40
Football	64	1	63
Programming	80	6	74
Data Structure	77	3	74
Mathematics	92	2	90
Algorithms	66	1	65
Statistical	88	8	80
Mobile Computing	77	4	73

TABLE V. TIME FOR EACH QUERY OF RETRIEVAL PROCESS ON 10 QUERIES AS SAMPLE

10 Queries Samples	**Total Time of Crawling and Filtering in ms**
Computer Books	30
Computer Science	50
Java Tutorials	20
Football	40
Programming	60
Data Structure	40
Mathematics	40
Algorithms	40
Statistical	60
Mobile Computing	50

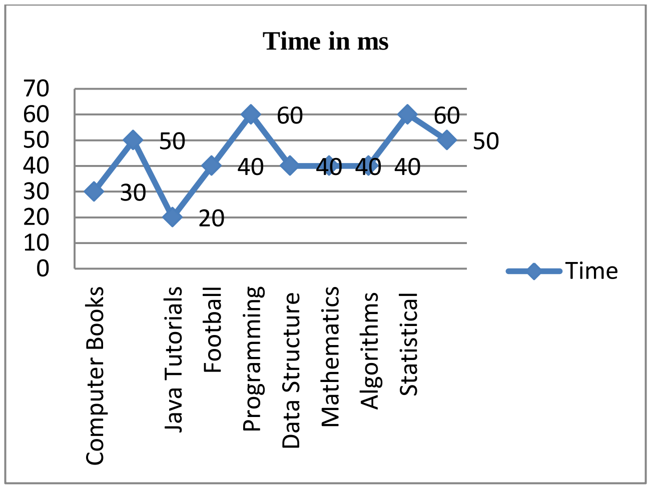


Fig. 9. Time Chart for Retrieval Process

VI. Conclusion

The topic of the semantic search engine has attracted large interests both from industry and research with resulting variety solutions in different tasks. There is no standardized framework that helps to monitor and stimulate the progress in this field. In this paper, Four standard tasks of semantic search engine are discussed including crawling, indexing, ranking and finally retrieving task.

We focus on ranking phase that considered as the main contribution of this paper. New ranking algorithm is produced to rank similar meaningful data after indexing phase. In addition to, data retrieval process become faster, easier and more accurate. The performance achieved with 99 percent relevant results in maximum time 60 ms and 1 percent only for irrelevant results. The proposed framework and ranking algorithm can be further developed for future use in detecting more accurate semantic information from social networks in a short time.

References

[1] J. Beal, “Weaknesses of Full text search”, The Journal of Academic Librarianship, vol. 34, Number 5, 2008, pp. 438-444.

[2] J. Beal, “Geographical research and the problem of variant place names in digitized books and other full-text resources” Library Collections, Acquisitions, and Technical Services, vol. 34, Issues 2–3, 2010, pp. 74-82.

[3] M. P. Selvan, C. A. Sekar and P. A. Dharshini, “Survey on Web Page Ranking Algorithms”, International Journal of Computer Applications, vol. 41, no.19, Published by Foundation of Computer Science, March 2012.

[4] T. Berners-Lee, “Weaving the Web”, pp. 2-5. The Original Design and Ultimate Destiny of the World Wide Web by its Inventor. Harper: San Francisco, 1999.

[5] L. Chang, W. Haofen, Y. Yong and X. Linhao, “Towards Efficient SPARQL Query Processing on RDF Data”, Tsinghua Science & Technology, vol. 15, issue 6, Dec. 2010, pp. 613-622.

[6] Tim Finin, Yun Peng, R. Scott, Cost Joel , " Swoogle: A Search and Metadata Engine for the Semantic Web " University of Maryland Baltimore County, Baltimore MD 21250, USA, 2011.

[7] Aidan Hogan and Andreas Harth and Jürgen Umrich and Sheila Kinsella and Axel Polleres and Stefan Decker: "Searching and Browsing Linked Data with SWSE: the Semantic Web Search Engine." In JWS 9(4), 2012.

[8] Gong Cheng and Yuzhong Qu: "Searching Linked Objects with Falcons: Approach, Implementation and Evaluation." In IJSWIS 5(3), 2009.

[9] O. Harting and F. Huber, F., “A main memory index structure to query linked data”. Proc. 4th Linked Data On the Web (LDOW11), March 2011.

[10] O. Harting and J. Freytag, J “Foundations of traversal based query execution over linked data”, Proc. 23rd ACM conference on Hypertext and social media (HT '12). ACM, New York, NY, USA, 2012, pp. 43-52.

[11] D. Ding, J. Yang, Q. Li, L. Wang, and W. Liu, "Towards a flash search engine based on expressive semantics," in Proceedings of WWW Alt.'04 New York, 2004, pp. 472-473.

[12] G. P. Schneider and J. Evans, “New perspectives on the Internet”. Ohio : South-Western, 2012.

[13] M. P. Selvan, C. A. Sekar and P. A. Dharshini, “Survey on Web Page Ranking Algorithms”, International Journal of Computer Applications, vol. 41, no.19, Published by Foundation of Computer Science, March 2012.