QUESEM: Towards building a Meta Search Service utilizing Query Semantics

Neelam Duhan[1] and A. K. Sharma[2]

[1] Department of Computer Engineering, YMCA University of Science & Technology Faridabad, Haryana, India

[2] Department of Computer Engineering, YMCA University of Science & Technology Faridabad, Haryana, India

Abstract

Current Web Search Engines are built to serve needs of all users, independent of the special needs of any individual. The documents are returned by matching their queries with available documents, with no emphasis on the semantics of query. As a result, the generated information is often very large and inaccurate that results in increased user perceived latency. In this paper, a Semantic Search Service is being developed to help users gather relevant documents more efficiently unlike traditional Web search engines. The approach relies on the online web resource such as dictionary based sites to retrieve possible semantics of the query keywords, which are stored in a definition repository. The service works as a meta-layer above the keyword-based search engine to generate sub-queries based on different meanings of user query, which in turn are sent to the keyword-based search engine to perform Web search. This approach relieves the user in finding the desired information content and improves the search quality for certain types of complex queries. Experiments depict its efficiency as it results in reduced search space.

Keywords: *World Wide Web, Information Retrieval, Search Engine, Query Processing, Semantic Search Techniques.*

1. Introduction

Search Engines enable users to navigate through the web information content incrementally and interactively. The outcomes of search typically depend on the submitted queries. But, the effectiveness of queries cannot be guaranteed as they vary from user to user, although exposing common information needs. In contrast, the same user query may convey different interpretations or different information needs. So, most of the users get irrelevant results or the effort for reaching needed information becomes very high due to vague or ambiguous formation of their queries. In other words, users often do not know how to combine the right words and in what order. Due to the lack of domain knowledge, users tend to post very short queries, which generally do not express their information needs clearly and thus, the precision and recall of the search results get decreased. Query refinement comes in handy in these situations.

Keyword based indexing in search engines is another factor towards irrelevancy of search results. They are unable to associate the query words with the related fields of our daily world. We have a magnitude of words, out of which many possess more than one meaning. The meanings are sometimes totally unrelated; for instance, how can "*lead*" be a verb meaning *to go first* and also the name of a *heavy metal*? Consider a user, who intends to search any one term out of "mouse", "cloud", "cluster" or "tree" on a search engine. Most commonly, the terms return the results concerning computer field only or the one which is most popular. In case, user did not find the required information, the modified query may be resubmitted and even then, it is not guaranteed that he will find the exact required pages. This process is very time consuming and irritating.

There are following issues, which need to be addressed in modern day search engines:

1. A term can have several synonyms, which are not considered while returning the search results to the user as a lack of their availability.
2. A term may have several different meanings in different contexts. One may be interested in a particular field and other contexts, which are not needed, may increase the volume of search results for nothing good and just become a hurdle in finding the appropriate URLs. Thus, the problem of "Information Overkill" arises.
3. Search engines are unable to provide different descriptions of query terms to users so as to assist them in searching in the right direction.

Most of the information about the synonyms, contexts and descriptions (or definitions) exists in *definition based* or *dictionary based* sites, which can be utilized by the search engines to resolve the above said issues. This paper proposes the concept of *QUESEM*, a Meta search service over the keyword based search, which utilizes the online web resources to provide semantic and context-oriented web search to the users. The rest of the paper has been organized as follows. Section 2 describes the related research done towards the semantic and context based searching. Section 3 explains the proposed approach in detail, while in Section 4, the system architecture, various components and the definition repository have been discussed. Section 5 gives the practical evaluation of the proposed approach and finally, Section 6 concludes the paper with a discussion of the future research.

2. Related Work

The development of semantic web search systems has been an emerging area of research since the last few years and many researchers have shown their interest in this particular field. Query Refinement and expansion has become an essential information retrieval approach that interactively recommends new terms related to a particular query. As keyword based queries are likely to miss important results due to unstructured and semantically heterogeneous nature of the Web, therefore query expansion is considered an effective method to bridge the gap between users' internal information needs [11,15] and external query expressions.

Thesaurus-based query expansion [12,13] generally relies on statistics to find certain correlations between query terms. Cui et al. [14] mine click-through records of search results from query logs to establish mappings from query terms to strongly correlated document terms which are then used for query expansion.

Giorgos Akrivas et al. [9] used the semantic entities, rather than normal terms and for this, they used the knowledge stored in a semantic encyclopedia, specially the ordering relations, in order to perform a semantic expansion of the query. Their work of query expansion also considered the query context, which is defined as a fuzzy set of semantic entities created by an inclusion function. Furthermore, they integrated the approach with the user's profile also.

In another approach [10], sentence context is used to perform web searches rather than keywords. In this method, contexts replace specific words in the search request with other predetermined words. The authors reduced false positives with an intelligent search based on grammar and English sentence structure. In their work, intelligent sentence searching converts each document into a set of simple sentences using only words in the predefined dictionary. These simple sentences capture the essence of the document. The conversion methodology uses synonyms, idiomatic expressions, grammar, patterns of speech and word location to create a searchable index. Because of the limited dictionary and elimination of most ambiguities, they claimed that such searches can be free of false positives.

Yaling Lie et al. [6] proposed a process-based search engine to handle certain type of queries that have an implicit process in it. The authors proposed to extract the process step by step of any given query as they assumed that most of the queries are in the form of a process and it would be better to find sub steps and then find the related URLs. To do this task they have taken help of process handbook to get the possible steps of any given query.

Y. Li et al. [8] proposed a solution to solve Web search queries having transactional intent, called *transactional queries*; for example, *Download software or fill in an online study-plan.* They claimed that many Web searches belong to this category. Existing search engines do not recognize these specific queries. In their work, a hand-crafted rule-based classifier is developed to recognize a collection of transactional Web pages for a set of transactions. General users are not able to contribute to or access the classifier. Therefore, it is difficult to build rules covering most of the transactional needs.

In [4], a novel method, Q-Rank, has been proposed to leverage the implicit feedbacks from the logs about the users' search intents. The authors claimed that to improve the relevance ranking for underspecified queries requires better understanding of users' search goals. By analyzing the semantic query context extracted from the query logs, they invented Q-Rank to effectively improve the ranking of search results for a given query. Experiments showed that Q-Rank outperforms the current ranking system of large-scale commercial Web search engines, improving the relevance for 82% of the queries with an average increase of 8.99% in terms of discounted cumulative gains. Because Q-Rank is independent of the underlying ranking algorithm [17, 18], it can be integrated with existing search engines.

The mechanisms proposed in the literature have their own advantages in particular areas of applications, but a critical look at the available literature indicates that most of existing semantic and context-based search techniques suffer from a couple of the following limitations:

- They are not able to capture the full set of synonyms for each type of query submitted by the users.

- Most of the techniques require complex analysis involving natural language processing and linguistic preprocessing to discover the context and semantics of query terms.
- The techniques utilizing external web resources for query expansion require complex retrieval efforts and the resource integration.
- Many of the techniques are targeted towards relevant page retrieval, but the produced results may not be presented in an easy user navigational manner. Moreover, the techniques do not aim to provide the user with a list of choices related to the query, from which the user can browse according to his interest.

This paper contributes towards developing a search service for semantic and context-based information retrieval, while at the same time, keeping in view the above limitations. The proposed technique has been made to take the advantage of dictionary based information available on the Web to gather possible meanings of a term and generate the query responses accordingly. The results will be represented in the form of clusters according to the newly found sub-terms. The next section explains in detail the proposed approach.

3. Proposed Approach of Semantic Search

In order to address the problems associated with keyword based search engines [16], a system called Query Semantic Search System (abbreviated as QUESEM, pronounced */'Qu-sem/*) to improve searching quality and reduce searching time is proposed. *QUESEM* maintains a database of definitions (referred to as Definition Repository), as the core of the system to accomplish its desired task.

In an abstract form, the approach to be followed by the system can be shown diagrammatically as in Fig. 1. Given a query, *QUESEM* first analyses it, after which, it matches terms in the query with the data (term_title and/or descriptive fields) stored in Definition Repository (explained in subsequent sections) to locate the relevant definitions also called sub-terms related to query. The *definition* here is meant to describe a semantic description of the term in question. For every distinct definition of the query, it uses an existing keyword-based search engine (e.g., Google or Yahoo) to search the Web pages on the WWW. *QUESEM* then displays the results in the form of clusters corresponding to each extracted definition.

A Topical Crawler is developed here to download Web sites, which specialize in definition-related or dictionary-related content. A Definition-Generator/Annotator with machine learning techniques is designed to automatically extract the relevant definitions from the crawled Web pages.

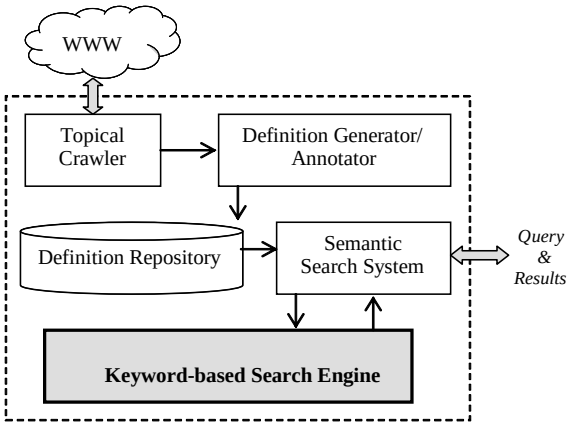

Fig. 1 An Abstract Architecture of QUESEM

An example scenario of a common search is given below to better illustrate the proposed approach:

A User 'X' wants to gain knowledge about "cluster". He is totally new to this term and has no idea about this.

As shown in Fig. 2, instead if displaying the URLs matching the query keywords or their combinations as in traditional search engines, *QUESEM* displays the matched terms having distinct meanings (definitions) related to the term 'cluster'. It may also include the related term descriptions.

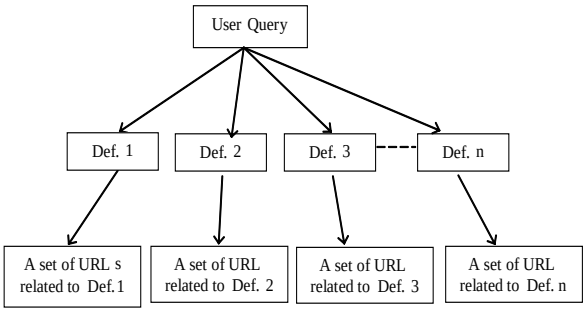

Fig. 2 The Expected output of QUESEM

'X' chooses the matched term which is of his interest, reads the description and then if found interesting, further explores the related URLs.

The aim here is to solve the information overkill problem with the help of dictionary based sites. The terms that are

found closer to the given search query on *yourdictionary.com* or other such sites are extracted by the definition generator module, annotated by annotator and stored in the Definition Repository for further use by the system. The next section describes the detailed system architecture and the functioning of various components involved over there.

4. System Architecture

The detailed system architecture of *QUESEM* is shown in Fig. 3, where the dashed line represents the proposed meta-search service. In order to achieve the required task, architecture is divided into two major sub-systems as given below:

1. Definition Repository Generation
2. Definition based Search

The basic definitions and the working of these two subsystems are explained under.

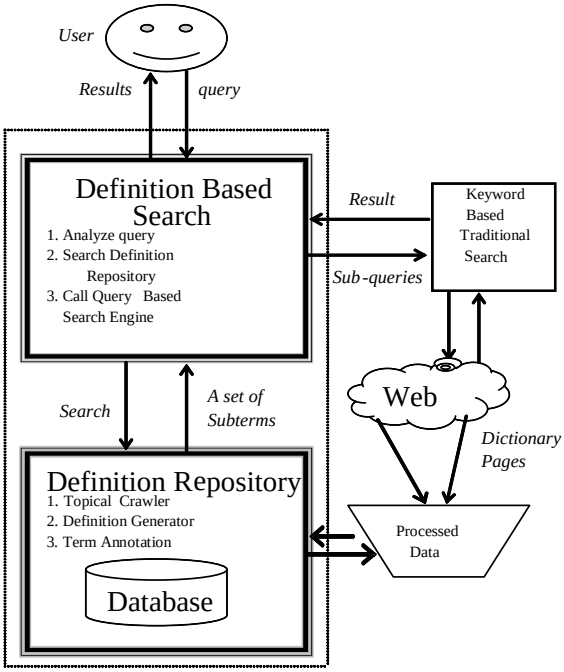

Fig. 3 High-Level System Architecture of QUESEM

4.1 Basic Definitions

Some definitions are formulated here, which are related to the proposed search system.

Definition of *"Definition"*:
A *definition* is a phrase or set of symbols that define the meaning of a term or similar kind of things. A term may have many different senses or meanings in different contexts. For each such specific sense, a definition is a set of words that defines it. Its existence in a particular field defines a terminology of that field.

Definition of *"Definition Repository"*:

A database for storing terms, their related definitions and a group of programs, which provide means to collect and access the data. The schema must contain at least the two relations as shown in Fig. 4, and it may further contain optional fields or relations to facilitate term hierarchy. It collects and manages definitions, which are used by Definition based search sub-system to expand the initial query into multiple sub-queries or definitions.

TERM

Term_ID	*Term_Title*	*Term_Description*

DEFINITION ANNOTATION

Def_ID	*Def_Title*	*Def_Description*	*Term_ID*

Fig. 4 Schema of Definition Repository

The schema is very simple and is made to handle only single level of hierarchy, however whenever there is a need to extend the schema complexity to handle the large size of definition repository. The fields with a solid underline represent the primary key and a dashed underline represents a foreign key. Table 1 gives description of various fields in the Definition Repository.

Table 1. Various Fields in Definition Repository

Field	Description
Term_Id	An attribute that gives an identity number to the term under consideration.
Term_Title	It contains the actual query terms that are entered by the user.
Term_Description	It contains a limited length snippet to have a small description of the term.
Def_Id	It represents the Identity numbers of the semantic Definitions (subqueries) extracted for each query term.
Def_Title	This column specifies the definitions for query terms that are found relevant in the dictionary-based sites after parsing.
Def_Description	Small description of each definition.

Fig. 5 gives the state of the repository for the term "Cluster" and its various definitions viz. "Cluster Headache", "Cluster Bomb" etc.

TERM

1	Cluster	A group of similar objects…

DEFINITION ANNOTATION

1.1	Cluster headache	It is less common than migraine headache…..	1
1.2	Cluster bomb	Cool crust band…..	1
1.3	Cluster analysis	Cluster analysis classifies a set of observations……..	1
1.4	Cluster bean	Drought tolerant herb……	1
1.5	Cluster (computer)	It's a technique to categorize ...	1

Fig. 5 Example Illustration of Definition Repository

Definition of *"Definition based Search"*:
It is an extension to the traditional Web search, which discovers the implied definitions *d1, d2,… dn,* of the initial query *q* (if they exist) using the definition repository. It performs traditional Web search on each definition d_i, *i= 1, 2… n*. Let r_i denote the obtained URL list by searching on d_i, then *<r1, r2… rn>* represents the result of the Query Semantic search on the initial query *q*.

An example of this type of search was briefly described (illustrated in Fig. 2) earlier. The current assumption for this search is that the titles of definition in the *definition annotation* relation represent potential definitions for the initial user query term. For instance, for a given query "cluster", definitions extracted are 'cluster headache', 'cluster beans' etc. and the results for the query "cluster" now contain the result URLs of "cluster headache', 'cluster beans' etc.

4.2 Definition Repository Generation

Definition Repository is maintained by a series of steps as illustrated in Fig. 6. Various inputs, outputs and the components involved in the process with their mode of working are explained below in detail.

4.2.1 Information Resource: Dictionary Based Sites

For automatically populating the repository with high-quality definitions, the input resource taken is the dictionary based sites, which are a rich store of semantic definitions of terms explicitly published on the Web. In general, the web pages, which possibly contain this type of information, are very sparsely distributed on the Web. In order to efficiently extract more sub terms related to the query terms, while downloading relatively fewer pages, a topical crawler is needed to crawl only those Web sites, which are specialized in dictionary-related content.

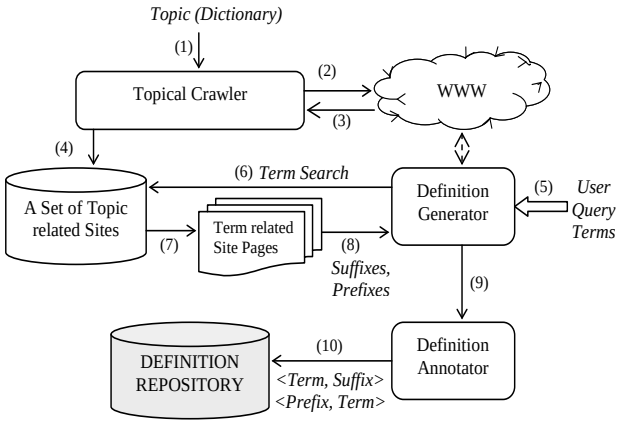

Fig. 6 Definition Repository Generation Process

4.2.2 Topical Crawler

A typical crawler for a generic web search engine recursively traverses through hyperlinks to explore the undiscovered portion of the Web [1]. The basic idea behind topical crawling is to estimate the relevance of undiscovered documents by the relevance of fetched documents, which directly or indirectly link to the undiscovered ones [2, 3]. To start the crawl process, some topic of interest such as "dictionary" or related content is needed. Topical crawlers maintain priority queues, where most possibly related documents have the highest priority to be downloaded under the constraints of limited computing resources. Thus, topical crawlers traverse the topic-related portion of the Web.

A publicly available Web search service (Google SOAP API) can be used to filter dictionary-related Web sites by searching dictionary-related keywords (e.g. *dictionary, thesaurus, synonyms, answers, definition etc.*) in the title of the home page of a Web site. Here, we have taken only one site *yourdictionary*.com, but a number of sites can be used statically to do this task.

The topical crawler will output a set ***S*** of sites related to dictionary based content, which is stored in a local repository to be referred further by *definition generator.*

4.2.3 Local-Site Search by Definition Generator

When the user first time submits a query to *QUESEM*, its keywords would be passed to *definition generator* module. This component consults the set *S* of dictionary-based sites and passes these query terms over their interfaces to perform a local-site search. After that, the resultant set of pages related to query terms are retrieved by this component and placed in a set *D*. The resultant pages are

further parsed to extract the linked pages related to query terms. The extracted pages are also kept in *D*.

An approach [5], which uses local site-searches to estimate the relevance of the documents before fetching them, is used here to help getting the definitions. The algorithm for local-site searching is shown in Fig. 7.

Algorithm: Local_Site_Searching (Initial query, S)

I/P= Initial query *q* and Set *S* of Sites stored in a repository
O/P= Unstructured document set *D* containing documents that are response pages against the initial query *q*.
// Start of Algorithm
Begin
For (every site $si \in S$) *// perform the local-site search*
Begin
Step1: fetch the homepage of *si*
Step2: find the local-site search form
Step3: Submit the query terms (*q*)
Step4: Fetch the response pages in local Repository D
Step5: For (every response Page *//Find linked pages*
Begin
Parse the Page;
Fetch all result links;
If synonyms link exist
Begin
Fetch synonyms page and place in D
End
End
End
Return the fetched page set D
End

Fig. 7 Algorithm for Local Site Search for Query Terms

The set *D* of documents d_i for $i= 1. .. n$ is the returned set of pages fetched from dictionary-based sites, which contain the relevant information regarding the semantics or context of query terms. This set *D* is used for *definition generation* and *annotation*.

4.2.4 Definition Generation & Annotation

As the response pages are the result of the dictionary based sites, it is assumed that the pages will contain the direct thesaurus and synonyms of the query terms. The approach to be followed to find semantic definitions is given below:

"Extract the prefix and suffix tokens of query terms from pages d_i, annotate them with query terms and store the result in Definition Repository"

The *Definition Generator* simply extracts the *prefix* and *postfix* present consecutively in combination with the query terms from the pages belonging *D*, while *Definition Annotator* combines them suitably with the query terms to result in proper annotations of definitions. We can expect that the result will be the required modified input for our search goal. The algorithm for definition generation and annotation is shown in Fig. 8.

Algorithm: Definition_Generator_Annotator (*D*)

I/P= Unstructured document set D of pages containing semantics of query terms.
O/P= Term and their associated Annotated Definitions
//Start of Algorithm
Begin
Term_title= *q*
For (every d_i in D)
Begin
Set_before= Consecutive tokens occurring before *q* in *di* ;
Set_after= Consecutive tokens occurring after *q* in *di* ;
End
For (every token t_i in *Set_before*)
t_i _Title ← < t_i + q> *// ti_Title is the title of definition*
For (every *ti* in *Set_ after*)
t_i _Title ← < q+ t_i >
Place term_titles, definition_titles *ti* in the Definition Repository
End

Fig. 8 Algorithm for Definition Generation/Annotation

The prefixes and postfixes of the query terms play an important role in semantic definition generation. It is assumed that most of the definition based sites manage the data in the thesaurus and synonyms form. This assumption may restrict the number of definitions that could be found, but for a precise search, this assumption may be much more relevant as in dictionary based sites, the most relevant sub terms can be found nearby to the basic term.

Therefore, prefix and suffix represent the synonyms/contexts, which have some how a meaning equivalent to query term. To extract query semantics from the pages d_i, which are in the form of prefix and suffix, parsing is required. A parser is used to do tokenization of the response pages and the query terms are kept in track to find their relevant prefixes or suffixes, which are in turn annotated with the query terms by the *Definition Annotator* to generate the definition titles. For example, as was shown in Fig. 5, "bean", "bomb", "headache", "analysis", "computer", "controller" etc. all represent the prefixes or suffixes related to the term "cluster". The figure also shows the definition titles after annotations e.g. "cluster headache" is the annotated definition.

4.2.5 Populating the Definition Repository

Initially definition repository would be empty and it will be maintained as the user queries would be submitted to *QUESEM*. The definitions generated by the *Definition Generator & Annotator* with respect to each new query will be populated into the Definition Repository. It may be

possible that the definition repository may contain some inadequate definitions, but as user explores a cluster of URLs relative to his interest, some abrupt clusters may not affect the search.

4.3 The Definition based Search

When user submits a keyword based query to *QUESEM*, the keywords are passed to "Definition Repository Generation" subsystem to build the definition Repository as well as to the "Definition based Search" subsystem to respond to the user in the form of cluster of result URLs. Contrary to the traditional keyword based search, semantic or definition based search requires slightly complex query processing. Various modules which are used in definition based query processing are given below and are outlined in Fig. 9.

1. Query Analyzer
2. Definition Searcher
3. Query Transformer and Processor

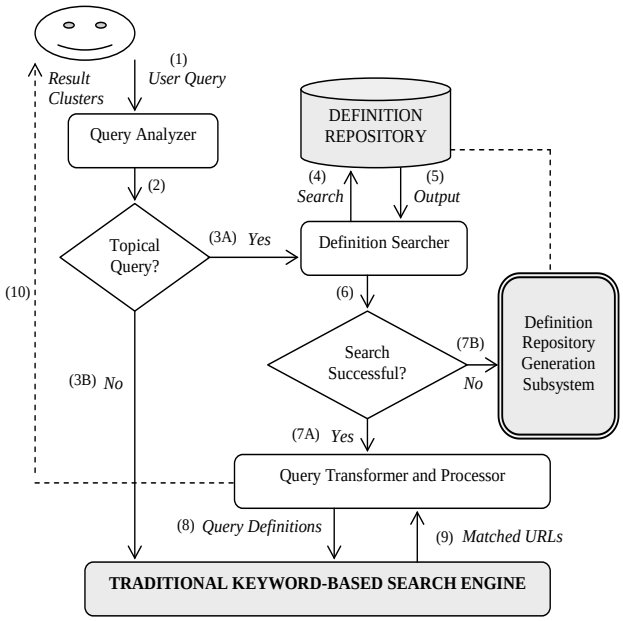

Fig. 9 Definition Based Search Subsystem

Query processing involves analysis of user query to perform extended search for it. Given a query, a *Query Analyzer* first decides whether to perform definition based search or a traditional search for the query? The *QUESEM* works only for topical queries. The topical user queries, which may be solved by this system, can observe two characteristics:

- They can have multiple meanings.
- They can have a number of synonyms.

After the analysis, the next step is to find the related definitions by searching the definition repository. This is performed by the *Definition Searcher* module. Finally, the initial query is transformed into a sequence of sub-queries or definitions by the *Query Transformer & Processor*. Obtained sub-queries are then sent individually to the traditional keyword-based search engine (like Google) to find the matched pages. Now the *Query Processor* represents the results obtained by different sub-queries in the form of clusters to the user.

The functioning of different components is described briefly in following subsections:

4.3.1 Query Analyzer

Query analyzer is responsible to check whether definition based search is applicable to the query or not? Analyzer first examines the query to decide whether a query is topical or not? A topical is the query, which is generally framed by simple keywords. For example "*mouse*", "*waiter*", "*data mining*", "*HCL laptop*" etc. are topical queries, while "*how to drive a car*", "*when monsoon will come*" etc. are not topical queries.

For analyzing the queries, openNLP functions [7] are used by the system. A query which contains a combination of <predicate, object> is considered a goal based query [6, 7], otherwise queries containing either <predicate> or <object> is considered a topical query, for which QUESEM gives better results. It is assumed that topical search query either have a *verb phrase* or a *noun phrase*. The queries, which are not topical, are made to be searched for using the traditional search system.

4.3.2 Definition Searcher

The job of Definition Searcher is to check whether the topical query already exists in the system's Definition Repository. If it is there, that means some user has already queried it and as result of which its definitions are already stored in the database. Now, it is not required again to follow the same procedure, but simply extracting the definitions from the definition repository. If, on the other side, if definition repository doesn't return any set or say return NULL then it is the functional responsibility of the topical crawler of *Definition Generation Subsystem* to become active and perform all the tasks necessary to get the new definitions matching the new query and expand the definition repository.

In searching the definitions, literal term matching is done. It is simple keyword-to-keyword matching which return a set of definitions corresponding to the term. As the user queries tend to contain common words, punctuation marks,

stop words, case sensitivity etc, handling all these aspects comes in preprocessing the query which is sent to the definition searcher. The related definitions are passed to Query Transformer and Processor to process the queries.

4.3.3 Query Transformer & Processor

Query transformer is responsible for making the retrieved definitions as a sequence of well-defined queries so that they could be searched individually as independent queries. The original query and the matched set of definitions will be combined to form a new set of sub-queries. Query Processor searches these sub-queries individually with the help of a traditional search engine and then represents the resultant URLs in the form of a cluster with cluster label being the *definition_Title*.

5. Performance Evaluation

QUESEM was implemented in asp.net 2.0, C# and HTML with MS SQL Server 2005 at the back end to support definition repository. For the experimental purposes, only *yourdictionary.com* is examined for the current scenario. A group of 25 users from different domains were asked to search on *QUESEM* and other keyword based search engines like Google, Yahoo etc. The net performance of QUESEM in terms of quality of search results and reduced navigation time is observed to be higher than normal traditional web search. Fig. 10 shows interface of the search service.

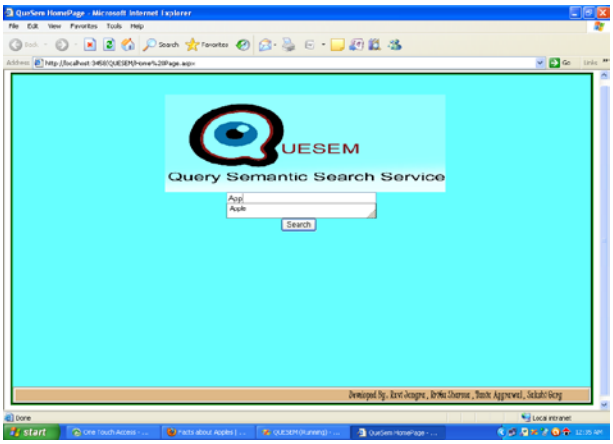

Fig. 10 Interface of QUESEM

Fig. 11 shows the result screen after submitting a query “apple”. It can be observed that QUESEM displays related terminologies in terms of definitions like “apple mobile”, “apple computer”, “apple fruit” etc. Similarly, Fig. 12 displays the terminologies related to query “stand”. When user clicks on a definition, corresponding results are displayed using a traditional keyword based search engine. Here Fig. 13 and 14 show the results of queries “apple” and “history of apple” after redirecting them to Google.

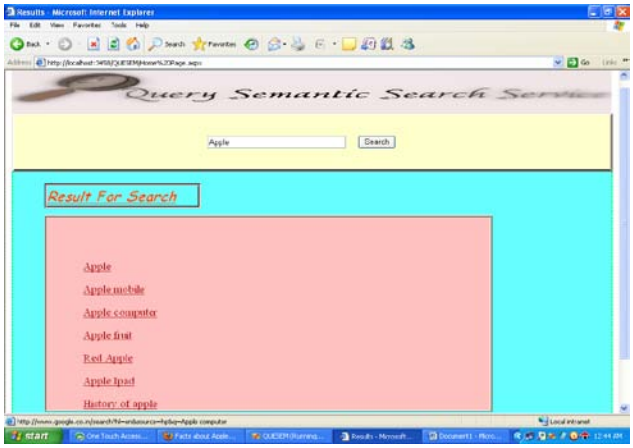

Fig. 11 The Definitions after submitting query “apple”

Fig. 12 The Definitions after submitting query “stand”

Fig. 13 Results for query “apple”

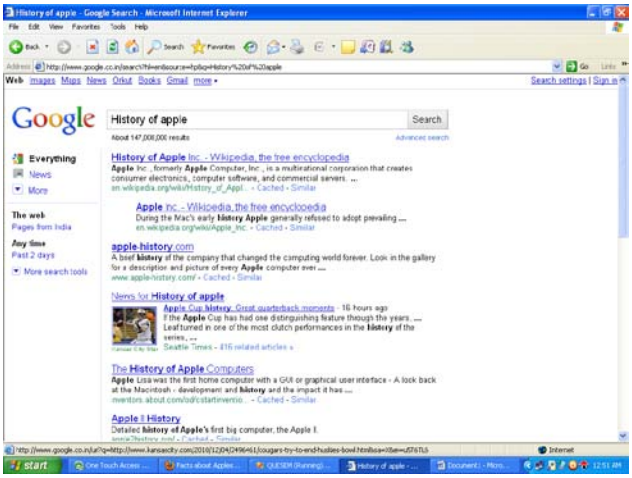

Fig. 14 Results for query "History of Apple"

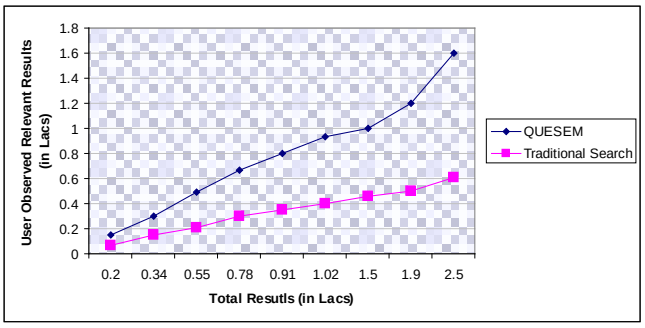

Fig. 15 Performance of QUESEM Vs Normal Search

The performance of QUESEM in terms of quality of search results and reduced navigation time is observed to be higher than the normal traditional web search. It is also assumed to be less complex as compared to other meta-search engines. Fig. 15 shows the performance comparison of QUESEM with the keyword-based search engines. It may be observed from the figure that in a normal web search, as the number of results increases, the user observed relevancy of documents decreases, while it remains approximate constant in semantic web search.

5. Conclusion

In traditional query/keyword based web search, some times, there are situations where a document is very much relevant to the user query but doesn't contain any similar term as described in his query and thus, does not appear in the search results. In this paper, a Query Semantic Search system to address these types of situations and "information overkill problem" has been developed. It is made to utilize the existing Web resources to automatically extract the synonyms (or semantics, thesaurus and contexts) related to user queries and enhance the user search efficiency. The proposed system QUESEM is designed to serve as a Meta layer above the traditional Web search engines. The different components are integrated to form a complete system towards effective search. NLP techniques are utilized for examining the user queries to be solved by the proposed system. Assisted by the information of definitions related to semantics of query terms, QUESEM is able to understand users' queries in a better way to perform more meaningful searches.

The future research includes enhancing the system towards serving different types of complex user queries, which may involve multiple keywords and even the disordered keywords. This will require building efficient query analyzers so as to direct the user search towards the right direction.

References

[1] Arasu, A., Cho, J., Garcia-Molina, H., Paepcke, A., and Raghavan, S., "Searching the Web". ACM Transactions on Internet Technology. Aug. 2001, pp: 2-43.

[2] Aggarwal, C. C., Al-Garawi, F., and Yu, P. S., "On the design of a learning crawler for topical resource discovery". ACM Transactions on Information Systems. Vol. 19, No. 3, Jul. 2001, pp: 286-309.

[3] Chakrabarti, S., Vandenberg, M., and Dom, B. "Focused crawling: a new approach to topic-specific Web resource discovery". In Proceedings of the Eighth International Conference on World Wide Web, Toronto, Canada, 1999. pp: 1623-1640.

[4] Ziming Zhuang, Silviu Cucerzan, "Exploiting Semantic Query Context to Improve Search Ranking". IEEE International Conference on semantic Computing, 2008 DOI: http://doi.ieeecomputersociety.org/10.1109/ICSC.2008.8

[5] Liu, Y. and Agah, A, "Crawling and Extracting Process Data from the Web". In Proceedings of the 5th International Conference on Advanced Data Mining and Applications, Beijing, China, August 17-19, 2009, Springer-Verlag, Berlin, Heidelberg, LNAI 5678, pp: 545-552.

[6] Liu, Y. and Agah, A, "A Prototype Process-Based Search Engine". In Proceedings of the third IEEE International Conference on Semantic Computing, Berkeley, CA, September 14-16, 2009.

[7] OpenNLP, http://opennlp.sourceforge.net/

[8] Y. Li, R. Krishnamurthy, S. Vaithyanathan, and H. V. Jagadish, "Getting work done on the web: supporting transactional queries," In Proceedings of the 29th ACM SIGIR, Seattle, Washington, 2006, pp. 557- 564.

[9] Giorgos Akrivas, Manolis Wallace, Giorgos Andreou, Giorgos Stamou and Stefanos Kollias, "Context Sensitive Semantic Query Expansion". Proceedings of the 2002 IEEE International Conference on Artificial Intelligence Systems (ICAIS'02).

[10] Alan Chickinsky, Chief Scientist, "Intelligent Searching using Sentence Context". IEEE International Conference on

Technologies for Homeland Security, 2008, May 2008. Greater Boston.
[11] Broder, A, "A Taxonomy of Web Search". SIGIR Forum, pp. 3-10, 2002.
[12] Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T.K., and Harshman, R, "Indexing by Latent Semantic Analysis". Journal of American Society of Information Sciences, 41 (6), pp. 391-407. 1990.
[13] Qiu, Y. and Frei, H, "Concept-based Query Expansion". In Proceedings of the 16th ACM SIGIR Conference, pp. 160-169. 1993.
[14] Cui, H., Wen, J. Nie, J., and Ma, W, "Probabilistic Query Expansion Using Query Logs". In Proceedings of the 11th WWW Conference, 2002.
[15] Jens Graupmann, Jun Cai and R. Schenkel, "Automatic Query Refinement Using Mined Semantic Relations", International Workshop on Challenges in Web Information Retrieval and Integration, April 2005, pp. 205-213.
[16] A. K. Sharma, Neelam Duhan, Bharti Sharma, "A Semantic Search System using Query Definitions". Proceedings of First ACM International Conference on Intelligent Interactive Technologies and Multimedia, Allahabad, India, Dec. 28-30, 2010, pp: 269-272.
[17] Neelam Duhan, A. K. Sharma, Komal Kumar Bhatia, "Page Ranking Algorithms: A Survey". In proceedings of the IEEE International Advanced Computing Conference (IACC'09), Patiala, India, 6-7 March 2009, pp: 1530-1537.
[18] Neelam Duhan, A. K. Sharma. "A Novel Approach for Organizing Web Search Results using Ranking and Clustering". International Journal of Computer Applications 5(10):1–9, August 2010. Published By Foundation of Computer Science.

Neelam Duhan received her B.Tech. degree in Computer Science & Engineering with Hons. from Kurukhetra University, Kurukshetra in 2002 and M.Tech. degree with Hons. in Computer Engineering from Maharshi Dayanand University, Rohtak in 2005. Presently, she is working as Assistant Professor in Computer Engineering Department in YMCA University of Science & Technology, Faridabad and has a teaching experience of 7 years. She is pursuing Ph.D. in Computer Engineering from Maharshi Dayanand University, Rohtak and her areas of interest are Databases, Data Mining, Search Engines and Web Mining.

Prof. A. K. Sharma received his M.Tech. in Computer Science & Technology with Hons. from University of Roorkee (Presently I.I.T. Roorkee) in 1989 and Ph.D (Fuzzy Expert Systems) from JMI, New Delhi in the year 2000. He obtained his second Ph.D. in IT from IIITM, Gwalior in 2004. His research interests include Fuzzy Systems, Object Oriented Programming, Knowledge representation and Internet Technologies. Presently he is working as the Dean, Faculty of Engineering and Technology & Chairman, Dept of Computer Engineering at YMCA University of Science and Technology, Faridabad. His research interest includes Fuzzy Systems, OOPS, Knowledge Representation and Internet Technologies. He has guided 9 Ph.D. thesis and 8 more are in progress with about 175 research publications in International and National journals and conferences. He is the author of 7 books. Besides being member of many BOS and Academic councils, he has been Visiting Professor at JMI, IIITM, and I.I.T. Roorkee.