Web Query Processing Approaches – A Survey and Comparison

M.Manikantan
Assistant Professor, Department of MCA
Kumaraguru College of Technology Coimbatore, India

S.Duraisamy
Professor and Head, Dept of MCA
Sri Krishna College of Engg. and Technology
Coimbatore, India

ABSTRACT

World Wide Web, in short www or simply web, is interconnection of hypertext documents through internet and accessed with the help of web browser. The web search is enabled by navigating hyperlinks in a webpage or through search engines or by web programming. The search queries are classified mainly in four types as Informational queries, Navigational queries, Transactional queries and Connectivity queries. We can classify the evolutionary development of web query processing from database query processing and SQL optimizations as Learning and Adaptive query processing, Web query through HTML and web search taxonomies, Web search query and search engines, Web query languages and its models, Semantic web and Ontologies, Web query optimizations on distributed web as well as on semantic web and Use of context-based techniques in web query processing. In this survey we are discussing on each of these topics and including how synonyms adding approach and Linguistic based approach are used in web query processing. There are three stages of query processing in general namely, Statistics generation, query optimization and query execution. Further, queries are optimized using performance and correctness measures namely Precision, Recall, Fall-out, F-measure, Average precision, R-Precision, Mean average precision, discounted cumulative gain and some more measures. Some of this surveyed paper discusses these details and others concentrate on their research work in different contexts. Our further work will be on the query using synonym based classifier or statistical classifiers, such as Naive Bayes (NB) and Support Vector Machines (SVMs). Other future work will be how to use unlabeled query logs to help with query classification and also on solution to adapt the changes of the queries and categories. We propose to use web query modeling using soft-computing techniques.

Keywords

Web query- Semantic Web - Ontology - query classification - query optimizations

1. INTRODUCTION

World Wide Web, in short www or simply web, is interconnection of hypertext documents through internet and accessed with the help of web browser. There are varieties of web browsers now available through which one can access web search results that may yield text, image, videos and other multimedia. The web search is enabled by navigating hyperlinks in a webpage or through search engines or by web programming. A search query is distinctive as plain text or hypertext with optional search-directives (such as "and"/"or" with "-" to exclude). The search queries are classified mainly in four types as Informational queries, Navigational queries, Transactional queries and Connectivity queries. The queries that cover broad topics with thousands of results are called Informational queries. The queries to seek a website or a web page are called Navigational queries. Transactional queries are given to complete a specific transaction or task like purchasing a mobile phone through online or doing a banking transaction. The queries which finds what links points to this web page or this URL with the help of indexed web graph. Mostly the web queries are fed into software called web search engine which is designed to get search results from World Wide Web. Search engines maintain real-time information through a software called web crawler. Web crawling looks into the indexing of web pages of websites and then searches for a result. Web crawler is also called spider. The search engines like Google or AltaVista keeps cache which store all or part of the source page as well as information about web pages and even sometimes store every word of every page they find. Also search engines use a principle called principle of least astonishment which returns users expected pages. Proximity search allows users to define the distance between keywords. Sometimes users provide search query through boolean operators AND OR and NOT to enhance the better result. In Concept based search, statistical analysis on pages containing words or phrases that the user searches for. An interesting query pattern is using natural language queries that allow the user to type a question in the same for one would ask to a human. The search results include a word or phrase or pages which may be more relevant, popular, and authoritative than others. Out of the thousands of the results, search engines ranks them to provide the "best" results first. In Fig.1 [62] web query processing architecture is shown.

The search engines are of two broad categories and the first one is a system of predefined and hieratically ordered keywords and the other is through "inverted index" by analyzing texts it locates. Search engines do not charge the users for a fee but they get revenues through advertisements along the portions of websites and through clicks on them. Information Retrieval is the activity of obtaining information resources through searches over metadata or on full text indexing. Web search engines are also IR System and it is of the type "Automated information retrieval systems". IR query refinements can be applied to web query processing and they can be classified into two categories and the first one is based on mathematical basis and the other is based on using properties of modeling. In Mathematical process of refinement, there are three types' namely set-theoretical models, algebraic models and probabilistic models.

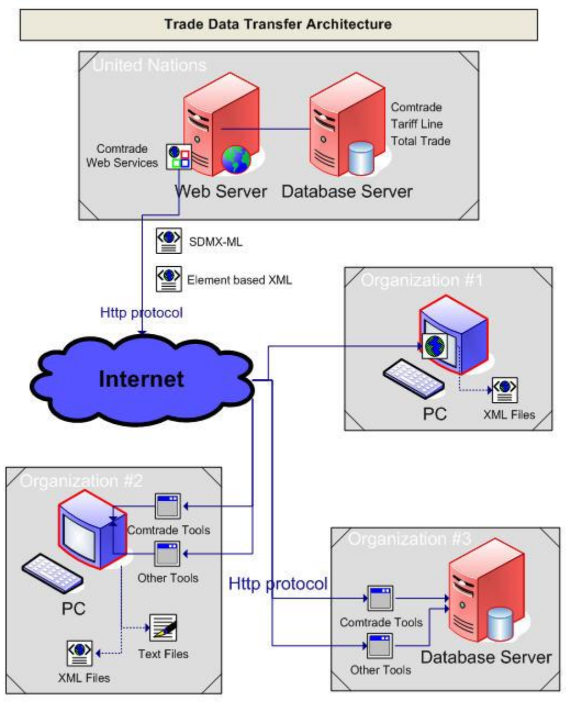

Fig 1: Web query processing architecture

Set theoretical models are Standard Boolean model, Extended Boolean model and Fuzzy retrieval. Algebraic models are vector space model, generalized vector space model, topic-based vector model, extended boolean model and latent semantic analysis. Probabilistic models are Binary Independence Model, Probabilistic relevance model on which is based the okapi (BM25) relevance function, uncertain inference, Language models, Divergence-from-randomness model and Latent Dirichlet allocation. Feature based retrieval models considers features and uses it in ranking. The properties of model method considers two types namely Models without term-interdependencies treat different terms/words as independent and Models with transcendent term interdependencies allow a representation of interdependencies between terms. Further, queries are optimized using performance and correctness measures namely Precision, Recall, Fall-out, F-measure, Average precision, R-Precision, Mean average precision, discounted cumulative gain and some more measures.

With web query processing the following are important concepts.

- **sessions** - changes in queries during a session, number of pages viewed, and use of relevance feedback,
- **queries** - the number of search terms, and the use of logic and modifiers, and
- **terms** - their rank/frequency distribution and the most highly used search terms.

Further, we classify our survey papers into seven categories, namely

1. Learning and Adaptive query processing.
2. Web query through HTML and web search taxonomies.
3. Web search query and search engines.
4. Web query languages and its models.
5. Semantic web and Ontologies.
6. Web query optimizations on distributed web as well as on semantic web.
7. Use Context awareness in web query processing.

2. LITERATURE REVIEW

2.1 Learning and Adaptive query processing

To locate information across the Web, search engines are used. For those who are accustomed to query and retrieve from databases will find that it is difficult to get similar query and result from the web. Also, results from a query in database will be straight forward whereas the web will return several similar web pages and the user has to search through it for the exact result. Information Retrieval (IR) is the area of research by querying the web through building and classifying indices, using better searching strategies and ranking functions, etc. Software agents can be used to learn user profiles by modeling user search behaviors and then using such knowledge to find URLs containing interesting information and providing suggestion to the web user. *Fab* [32] and *InfoSpider* [33] use heuristics to find interesting web query results. *Syskill & Webert* [34] and *WebWatcher* [35], help users in an interactive mode.

For the given web search query and a URL is returned from the search engine. Here the system either returns a web page segment or no result is found from the related Web site as either *Found* or *Not Found* which may either *Right* or *Wrong* to indicate whether the system makes a correct decision. Using these four terms, a query result belongs to one of the four categories. To evaluate *effectiveness*, the following metrics are defined [1]:

Precision = # Right Found / # Found,

Recall = # Right Found / # Sites Containing Queried Segments,

Correctness = # Right Sites / # Sites Processed.

The *absolute path length* is the number of visited pages to locate a queried segment or to conclude that no queried segment can be found. The *relative path length* to locate a queried segment is the ratio between the absolute path length and the level of the queried segment (i.e. the length of the shortest path to locate this segment). The two metrics are presented as:

Absolute Path length = # Visited pages,

Relative Path Length = # Visited pages / Level of the Queried Segment.

The system [1] works well for the first 4 queries used. Accuracy is above 80% and in some queries it reaches 90%. For the last query, precision and recall are not as high as those are in other queries. The system is capable of filtering out irrelevant sites, thus to make the correctness reasonably good. The relative path length to locate a queried segment is close to 1. The absolute path length in an irrelevant site is not more than 2.5 pages.

Adaptive query processing is dealing with unpredictable, dynamic data volumes and transfer rates. Hence [2] explains adaptive query processing as writing queries so that they have the ability to process XML data, dynamic scheduling of operators to adjust to I/O delays, sharing and re-use of data across multiple queries and the ability to output results and update later. Tukwila project at the University of Washington is a query processing system that supports a range of adaptive techniques which is also explained.

Set of dimensions for Internet query processing discussed here [2] include, Data model, Remote vs. local data, First vs. last tuple, Approximate results, Incremental updates, Number of queries and Freshness of results. The basic techniques implemented in Tukwila are illustrated here. The three primary aspects of the Tukwila adaptive framework are an event-condition-action-based rule system, a set of adaptive operators, and the ability to incrementally re-optimize a query plan. The Tukwila system supports a number of adaptive techniques, but [2] extend number of aspects to it. As an extension, XML builds wrappers for every data source, to translate data from source formats into the standard Tukwila data format. Adaptive behavior is specified by including dimensions discussed with the factors that may include whether to optimize for first or last tuple, how fresh the data from each query must be using caching and whether and when user should see approximate or incomplete data. Here query optimizer creates long pipelines and when it runs out of memory, materialization takes place by breaking the pipeline temporarily. This upstream materialization flushes result to the disk and loads the intermediate results and resumes its operation. There are two issues addressed here, first, an interactive query system that ideally provides incremental results for interactive query through browsable window and second, it is found that for a given domain or query, the approximate or partial results are only useful for certain items within the data set.

Highly accurate semantic matches between the attributes of the source query interfaces are required for integrating web sources. Attributes on query interfaces often have no or very few data instances and this lacking reduces the accuracy of current matching techniques. To solve this problem, in [6], WebIQ, a solution that learns from both the Surface Web and the Deep Web to automatically discover instances for interface attributes was discussed. WebIQ uses artificial intelligent techniques through query answering techniques. It was demonstrated that matching accuracy improved from 89.5% F-1 to 97.5%, at only a modest runtime overhead. The solution suggested here consists of the following three components as below.

Discover Instances from the Surface Web [6]: The system employs a two-phased validation process. First, in the *outlier detection* phase, WebIQ detects and removes false instances by performing discordancy tests [36], based on a set of type-specific test statistics. Second, in the *Web validation* phase, WebIQ forms a set of validation queries, using the attribute label, the extracted instance candidates, and a set of validation patterns.

Validate Borrowed Instances via the Deep Web: Sometimes it will be difficult to formulate reliable extraction queries and it may fail to extract instances from Surface Web. To address this problem, WebIQ develops a solution to validate instances via the Deep-Web sources. Hence to verify that b is an instance of attribute A, WebIQ submits a query to the data source of A, with A's value set to b, then observes the response from the source.

Further it is a two phased approach namely The Instance Extraction Phase and The Instance Verification Phase. The first phase analyzes the labels with syntax and formulates next the extraction queries and finally instances are extracted using Surface Web. The second phase removes outliner instance candidates, and validates instances via Surface Web.

WebIQ borrows instances from the other attributes. For example, given an attribute A, WebIQ can also "borrow" instances for A from other attributes. Specifically, suppose b is an instance of attribute B, then WebIQ will try to verify if b can also be an instance of A. A novel approach to learning an instance classifier for an interface attribute was developed [6] using a Validation based Naive Bayes Classifier. Given an attribute A and a borrowed instance x (of attribute B), WebIQ validates instances via the Deep Web by allowing its component proceeds by submitting a formulated query and analyzes for the response. The results of the experiment show that matching time ranges from 1.9 minutes in the auto domain to 4.7 minutes in the airfare domain. Time spent with Attr-Surface was at most 3.5 minutes (in the job domain), time spent with Attr-Deep was at most 5.9 minutes (in the airfare domain). The total overhead ranges from 5.7 minutes in the real estate domain to 11 minutes in the airfare domain [6].

In [11], adaptive query processing trends and foundations of it are discussed. In this survey paper the various issues, themes, and approaches in this work are discussed and it focuses primarily on adaptivity of *intra-query* of long-running, and not on streaming queries. *Adaptive Query Processing* (AQP) addresses the problems of unexpected costs, missing statistics & correlations, and dynamic data by using *feedback* to *tune* execution. It is one of the cornerstones of so-called *autonomic* database management systems, although it also generalizes to many other contexts, particularly at the intersection of database query processing and the Web [11]. Adaptive query processing techniques have been quite broad: they may span multiple query executions or adapt within the execution of a single query; they may affect the query plan being executed or the scheduling of operations within the plan; they have been developed for improving performance of local DBMS, for processing distributed and streaming data and for performing distributed query execution. Query processing in a relational DBMS is to parse the SQL statement and produce a relational calculus-like logical representation of the query, and then to invoke the query optimizer, which generates a *query plan*. The query plan is fed into an execution engine that directly executes it, typically with little or no runtime decision-making. Adaptive query processing through internet is explained in Fig.2[63].

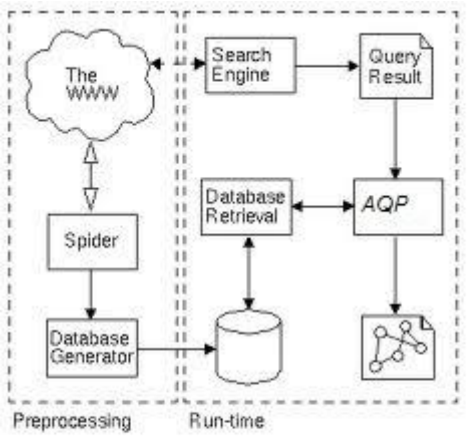

Fig 2: Adaptive query processing through Web.

There are three stages of query processing in general [11]. They are,

1. *Statistics generation. This phase is* offline (typically using the RUNSTATS or UPDATE STATISTICS command) on the

tables in the database. The system profiles the relation instances, collecting information about cardinalities and numbers of unique attribute values, and often generating histograms over certain fields.

2. Next phase normally done at runtime is *query optimization*. Optimization uses a combination of ***cost** estimation***,** pruning heuristics, and exhaustive enumeration.

3. The final stage, *query execution* for the compiled query plan. The important aspects of query execution through pipeline computation, such that each operator processes a tuple at a time from its sub-operators, and also propagates a single tuple to its parent for processing. An alternate approach, so called *data-driven* or *dataflow* scheduling [37], is used in many parallel database systems. A number of adaptive techniques use a hybrid of the iterator and data-driven approaches.

Adaptive query processing has emerged because of the developments in database query processing like Unreliable cardinality estimates, Queries with parameter markers, Dynamically changing data, runtime, and workload characteristics, Complex queries involving many tables, Interactive querying and Need for aggressive sharing.

The following are other interesting research work on query processing. Graefe's survey on query execution techniques [38]. Kossmann's survey on distributed query processing [39] also provides useful context for the discussion, as do Ioannidis and Chaudhuri's surveys on query optimization [40, 41]. Babu and Bizarro [42] also present a survey of Adaptive Query Processing, whether the scheme is plan-based, routing-based, or continuous query-based, from a different means of classification from [11].

Query processing using natural language, i.e., plain English, is used in [15] and for understanding English by the machine, statistical machine translation (SMT) is used. This approach is to bridge the lexical gap between questions and answers. SMT-based query expansion is done by i) using a full-sentence para-phraser to introduce synonyms in context of the entire query, and ii) by translating query terms into answer terms using a full-sentence SMT model trained on question-answer pairs. The experiments conducted shows that SMT-based expansion improves retrieval performance over local expansion and over retrieval without expansion.

In [23], an intelligent distributed query processing method by considering the characteristics of a distributed ontology environment is proposed. Here, the following are the contributions of it.

- Extended models of the distributed ontology query and the semantic mapping.
- Optimization techniques for an efficient query processing on the distributed ontologies.

For the query rewriting and the query optimization, [23] assume that every local site in the distributed environment registers its ontologies on a metadata registry. The registry maintains the schemas and the statistics of all ontologies for query rewriting and scheduling in the distributed environment. The distributed query processing needs to join the results of the local ontology queries is processed in memory. This system has 3 stages namely, Deeper Plan First Order (DPFO), Load balancing for distributed queries and Caching. AIDOS-I prune unnecessary queries before the query scheduling, and uses DPFO heuristic in the scheduling of each query. AIDOS-II supports all optimization techniques like pruning, DPFO, load balancing, and caching.

In Category-specific web search [25], the first step is to *train* a support vector machine (SVM) [43] to classify pages by membership in the desired category. Second, a set of query modifications is learnt. For this experiment, a *query modification* is a set of extra words or phrases added to a user query to increase the likelihood that results of the desired category are ranked near the top. The algorithm first trains an SVM classifier on labeled data and then automatically generates a set of good query modifications, ranked by expected recall. Finally, using the learned classifier to evaluate the query modifications on real search engines, a rank ordering of query modification, search engine tuples is produced. The classifier and the tuples are designed to improve category-specific web search.

2.2 Web query through HTML and web search taxonomies:

User searching for information is inherently predicted in Information Retrieval (IR) [3]. Information retrieval is not only informational but it might be navigational like “give me the url of the site I want to reach” or transactional like “show me sites where I can perform a certain transaction”, e.g. shop, download a file, or find a map. A Classic model involves, a user, driven by an information need, constructs a query in some query language. The query is submitted to a system that selects from a collection of documents (corpus), those documents that match the query as indicated by certain matching rules. A query refinement process is used to create new queries and/or to refine the results.
Web queries are classified according to their intent into 3 classes:

1. **Navigational.** The immediate intent is to reach a particular site.

2. **Informational.** The intent is to acquire some information assumed to be present on one or more web pages.

3. **Transactional.** The intent is to perform some web-mediated activity.

The users are self selected with attitude towards interstitial windows.

1. According to [44] 83% of the users view such pop-ups as interference.
2. The percentage of queries with sexual content is under 1%, whereas the percentage of such queries in the log is about 12%.
3. The percentage of queries identified as navigational was 24.5%, non-navigational queries was 68.4%, and 7.1% of the surveys did not answer Q1. The respondent to Q2, the percentage of navigational queries was 26.4%.
4. We could not find a simple question to distinguish between transactional and informational queries. Most popular transactional queries were identified as shopping on the web (Q3.1) and downloading data (Q3.3). Among people that answered both Q2 and Q3 these type of queries account for 32.3% of the non-navigational queries. The number of transactional queries is at least 23.8%.

2.3 Web search query and search engines

Query on a particular geographic region by user to constrain to search results in a Geographic web search engines. This geographic search technology is called local search and is being executed with significant interest in major search engines. Academic research was conducted for extracting geographic knowledge from the web. Combination of text and spatial data processing is used in such geographic web search engines. This research [5] concentrate on geographic search engines and it is merged with general web query processing. Each page in such search engine also has a geographic area of relevance associated with it, called the *geographic footprint* of the page
It extracts geographic information, such as city names, addresses, or references to landmarks, from the pages and then maps these to positions using external geographic databases. Footprints are represented as polygons and bitmap-based structures. These search engines can be divided into *crawling*, *data mining*, *index construction*, and *query processing.* In this system [5] it focuses on Germany and crawl the "de" domain; in cases where the coverage area does not correspond well to any set of domains, focused crawling strategies that may be needed to find the relevant pages. Overall ranking function might be of the form,

$F(D, q) = g(fD, fq) + pr(D) + F(D, q)$, with a term-based ranking function $F(D, q)$, a global rank $pr(D)$ (e.g., Pagerank), and a geographic score $g(fD, fq)$ computed from query footprint fq and document footprint fD (with appropriate normalization of the three terms). This paper [5] is on how to efficiently compute such ranking functions using a combination of text and spatial index structures.

Main contributions in this paper [5] are as follows:

- Discuss and formally study the query processing problem in geographic web search engines.
- Describe several efficient algorithms for query processing in geographic search engines.
- Integrate the algorithms into an existing high-performance query processor for a scalable search engine, and evaluate them on a large web crawl and queries derived from a real query trace.

There are search engines with immediate query processing and results are also required faster where as other search engines with large batches of queries are submitted for various web mining and system optimization tasks that do not require an immediate response and such search engines are called as batch processing search engines [9]. Here, conclusion is that significant cost reductions are possible by using specialized mechanisms for executing batch queries in Web search engines.

In particular, it describes three important scenarios where such queries arise:

- Cache Updates
- Internal Testing and Mining
- Queries by Outside Parties

Three major sources of savings and approaches in this paper [9]:

(1) Query Reordering: Given a batch of queries, executed in random ordering which results in significantly better caching behavior.
(2) Clairvoyance: Use clairvoyant algorithms for caching lists and partial results are demonstrated.
(3) Reusing Partial Results

The system compares the average time for each query in query stream, using 30% inverted index size as the caching buffer. The algorithm is then better (4.6 ms per query) than the strict admission case (6.05 ms per query) and the naive algorithm (6.86 ms per query). There is a 24% improvement over older algorithms.

Energy-price-driven query processing in a multi-center web search engines is discussed in [10]. Concurrent processing thousands of web queries necessitates maintaining and operating massive data centers which costs high energy consumption and a huge electric bill. This paper [10], takes the challenge to reduce the electric bill of commercial web search engines operating on data centers that are geographically far apart. Experiments on real-life query workloads obtained from a commercial search engine show that significant financial savings can be achieved by this technique.

2.4 Web query languages and its models

A large scale analysis on the extent of the query language differences is lacking and is being addressed in [8]. An extensive study on the issue by examining the language model properties of search queries and the three text streams associated with each web document: the *body*, the *title*, and the *anchor text*. The paper applies web scale n-gram language models to three search query processing (SQP) tasks: query spelling correction, query bracketing and long query segmentation. The n-gram language models in this work are built from different web sources, including the different text fields from the documents such as titles, anchor texts and body texts, as well as web search queries. In the web search setting, SQP tasks distinguish themselves from many traditional IR/NLP tasks in at least two aspects.
(1) SQP tasks deal with the query language instead of the formal natural language. The query language is found to be dominated by noun phrases that are loosely attached.

(2) In the web context, SQP tasks are faced with a living language with an open-domain and dynamic vocabulary, as new words are constantly being created. For instance, the query log of 2009 of a web search engine is drastically different from that of 2007. Even the grammar changes over time: it has been observed that the average query length increases over the years indicating users are becoming more sophisticated.

The paper [8] describes the Web N-gram Language Models Collection (Wnlmc). The collection is built from the high quality English web documents, in the scale of billions of pages and trillions of tokens, served by a popular search engine.

The contributions of this work are as follows:

- It builds web-scale language models from web sources including the search queries, as well as the titles, the anchor texts and the bodies of web documents.
- Unsupervised methods, including a novel query segmentation approach, are applied to efficiently tackle three search query processing tasks from query spelling correction to analyzing sub-query structures.

Three categories of Web query languages can be distinguished [14], according to the format of the data they can retrieve: XML, RDF and Topic Maps. This article introduces the spectrum of languages falling into these categories and summarizes their salient aspects.

A number of formalisms have been proposed for representing Semantic Web meta-data, in particular RDF, Topic Maps, and OWL (formerly known as DAML+OIL). These formalisms usually allow one to describe relationships between data items, such as concept hierarchies and relations between concepts. Semantic Web is an integrated access to the data on the Web that is represented in any of the above-mentioned formalisms.

The following three questions are at the heart of establishing a query language:

1. What are the core data retrieval capabilities of each query language?

2. to what extent, and what forms of reasoning do they offer, and

3. How are they realized?

It focuses on introducing and comparing languages designed primarily for providing efficient and effective access to data on the Web and Semantic Web. In particular, it excludes the following types of languages:

Programming language tools for XML, Reactive languages, Rule languages, OWL query languages.

The different query languages are illustrated using five types of queries against the sample data.

Select queries:

Query 1 “Select all Essays together with their authors (i.e. author items and corresponding names)”

Query 2 “Select all data items with any relation to the book titled ‘Bellum Civile’.”

Reduction queries:

Query 3 “Select all data items except ontology information and translators.”

Restructuring queries:

Query 4 “Invert the relation author (from a book to an author) into a relation authored (from an author to a book).”

Aggregate queries:

Query 5 “Return the last year in which an author with name ‘Julius Caesar’ published something.”

Query 6 “Return each of the subclasses of ‘Writing’, together with the average number of authors per publication of that subclass.”

Combination and inference queries:

Query 7 “Combine the information about the book titled ‘The Civil War’ and authored by ‘Julius Caesar’ with the information about the book with identifier bellum_civile.”

Query 8 “Return the transitive closure of the subclass of relation.”

Query 9 “Return the co-author relation between two persons that stand in author relationships with the same book.”

2.5 Semantic web and Ontologies:

According to the W3C, "The Semantic Web provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries [12]. In semantic web, the three different approaches are used namely, synonyms adding approach, Semantic approach and Linguistic approach. Let us discuss them here one by one through some prominent research work in these areas.

A simple unsupervised learning algorithm for recognizing synonyms was discussed in [45]. A synonym for a word is identified by choosing member from a set of alternative words that is most similar in meaning to the given problem word. This was demonstrated by using unsupervised learning algorithm which issues queries to a search engine and analyzing the replies to the queries. This is applied to measure a (human) student’s mastery of a language while conducting the tests like the Test of English as a Foreign Language (TOFEL) and English as a Second Language (ESL). There are several well-known lexical database systems that include synonym information, such as WordNet [46], BRICO [47], and EuroWordNet [48] which use the database of synonyms constructed by hand and not through automated machine learning system. Here, manual process involved must be repeated for each new language and must be repeated regularly as new terms are added to a language as well as scientific and technical words are difficult to be included through manual system. Also, only about 70% of the authors’ keywords were in WordNet. Statistical approaches to synonym recognition are based on co-occurrence and associations of words that are likely to be used in the same context. Hence, three methods for synonyms adding approach like statistical (unsupervised learning from text) techniques, some using hand-built lexical databases with some hybrid approaches, combining statistics and lexical information. In [45], statistical methods used to measuring semantic similarity is Latent Semantic Analysis (LSA) and data mining with interest calculation and text mining approaches are also used in it. The first step is to use the text to construct a matrix **X**, in which the row vectors represent words and the column vectors represent chunks of text (e.g., sentences, paragraphs, documents). The next step in this work applies a technique namely, SVD to **X**, to decompose **X** into a product of three matrices **ULA***T*, where **U** and **A** are in column orthonormal form (i.e., the columns are orthogonal and have unit length) and **L** is a diagonal matrix of *singular values* (hence SVD). LSA works by measuring the similarity of words using this compressed matrix, instead of the original matrix. The similarity of two words is measured by the cosine of the angle between their corresponding compressed row vectors [45]. Here, the results with the TOEFL questions show the algorithm can score almost 10% higher than LSA. The results with the ESL questions support the view that this performance is not a chance occurrence. Now, let us discuss various trends in web query process as below.

2.5.1 Synonyms adding approach

OBSERVER [49], is brokering across the domain ontologies which uses inter-ontology relationships representation for synonyms, hyponyms, and hyper nyms between different ontologies. User queries are rewritten by using this to translate across ontologies. Precision and Recall are used to estimate loss of information if any. It solves most challenging issue of vocabulary sharing between ontologies using web query processing.

There is no centralized or federated information management as anyone can put up a web page or make other information resources available on the Web independently. It is also impossible for users to be aware of the locations, organization/structure, query languages and semantics of the data in the various information resources because of the dynamic and open nature of such an environment [49]. It presents a method that performs the translation using an extended set of relationships including (but not limited to) synonyms, hyponyms and hyper nyms. Furthermore, the crucial issue of estimating the possible loss of information when using relationships other than synonyms was also discussed. It chose a significant set of ontologies built from three different points of view: linguistics research, knowledge representation research and the individual point of view of some research groups.

- ***Linguistic point of view ontologies.*** Two ontologies belong to this category: *WN* which was based on WordNet 1.5 and *μCosmos* which is a subset of MikroKosmos. These ontologies were created from the point of view of a global ontology: they are knowledge bases used by their creators (and other users) to perform automatic recognition and translation of texts. They are very big in size because they try to represent any kind of knowledge.
- ***Knowledge representation view point ontologies.*** Two other ontologies are subsets of the 'Bibliographic-Data' ontology developed as a part of the ARPA Knowledge Sharing Effort.
- ***Individual point of view ontologies*** Two ontologies were considered more to describe the information (and differing points of view) of two different research groups at different universities.

The query which is used as a running example throughout this paper is: "Get title, number of pages, year of publication and a file containing the publication itself, if available, for books related to Mars where the only author is Carl Sagan."

The following problems need to be solved in order to answer a query such as above in a global information system:

Resource Discovery: There is a need to search for the repositories relevant to the query, (e.g., which repositories are likely to have information about publications related to Mars?).

Structure/Format Heterogeneity: Different data repositories containing relevant data for a query may have different data organizations (e.g., publications are stored in databases, file systems, etc.), formats and media, (e.g., the sample query requests the publication that is stored in a non-textual format such as a Microsoft Word *(* document), and may support different query languages.

Modeling of Information Content: The same information may be modeled at differing levels of abstraction (e.g., "book" vs. "publication"). On the other hand, only part of the information required may be modeled at an information source/ontology. For example, information about books and authors may be modeled at different information sources/ontologies.

Querying of the Information Content: If keywords do not appear in a document, such a document will not be retrieved even though it may be relevant (e.g., the words "Carl Sagan" may not appear in the content of his publications). There is a need a semantic approach to access information rather than a syntactic approach. This should also define a query language expressive enough to precisely describe the information needs in an intentional manner.

The Vocabulary Problem: Current Internet tools and query processing systems are unable to support heterogeneous vocabularies used to describe the same information. In the case of keyword-based systems, if a synonym of the keyword included in the document is used as a part of the query, the document may not be retrieved. Different but related terms may be used to describe similar information at the intentional level (e.g., the term for "subject" may be modeled as "keywords" at a different ontology) as well at the extensional level (two different representations represent the same value: for instance, 'February 10, 1998' and '2/10/98').

Imprecise answers: There is a need to deal with imprecise answers when most of available data repositories do not model our information needs exactly. The loss of information incurred should be measured, controlled and reduced as the system enriches the answer by visiting more repositories. In the example, the system could deal with books with Carl Sagan as the only author even though there may be no way to know if those books are about Mars.

2.5.2 Synonym approach

OBSERVER uses three step query processing approach, namely, Query *Construction,* Access *to Underlying Data* and *Controlled Query Expansion to New Ontologies* are discussed as below.

Query construction.
This step consists of two tasks discussed below with the help of the example introduced earlier. These tasks involve direct user intervention. A GUI facilitates this user involvement.

1. Select User Ontology.
2. Edit Query.

Accessing the underlying data.
The Ontology Server retrieves the information corresponding to the query. That information is retrieved from the different repositories in a common format so that the partial answers can be easily combined, i.e., *correlated*. The answer is returned to the user by the Query Processor. The query processing ends if the user is satisfied with the answer.
Controlled query expansion to new ontologies.
The original query is translated from terms of the user ontology into terms of component ontology, also referred to as target *ontology*. The translation cannot always be exact because all the abstractions represented in the user ontology may not appear in the component ontology. Hence the user can define a limit for the *loss of information* allowed.

For the task of translating the query from terms of the user ontology into terms of the target ontology, the user and the target ontologies are integrated automatically by the Query Processor. If the user query is completely expressed in terms of the target ontology, the translation is called a *full translation*. Otherwise a *partial translation* is obtained. The Ontology Server is the module that provides information about ontology residing on its node as well as about their underlying data repositories. There is only one Ontology Server per node containing ontologies. The Ontology Server maintains a catalog of its ontologies called *Repository Catalog*, which contains information about the data repositories underlying each ontology. OBSERVER deals with the following kinds of interontology relationships, which are a subset of the defined in the other work.
Synonym relationships, when two terms in different ontologies have the same semantics. But need not have the same extension.

Hyponym relationships, when a term is less general than other in ontology, i.e., a term in one ontology subsumes the abstraction represented by another term in a different ontology. Note that this does not mean that the extension of the term is a superset.

Hypernym relationships, when a term is more general than other in ontology, i.e., a term in one ontology is subsumed by another term in a different ontology. Note that this does not mean that the extension of the general term is the superset of the extension of the specific term.

Overlap relationships, when there exists an intersection between the abstractions represented by two given terms. Two numbers can be associated with these relationships to represent the approximate percentage of the two terms participating in the relationship that belong to the intersection.

Disjoint relationships, when there exists no intersection between the abstractions represented by two terms.

Covering relationships, when the abstraction represented by a term in one ontology is the same that the abstraction represented by the union of other given abstractions which are subsumed individually by the term.

Issues related to the correlation of data from multiple repositories:

Logical schemas, data repositories and data sources

From the point of view of mapping expressions, data repositories are seen as a set of entity types and attributes. Thus each repository has an associated logical schema. Mappings act as an intermediate language between Description Logic expressions and the query languages of the local repositories. Each data repository has a specific data organization and may or may not have a data manager.. The different data sources of a repository can be distributed. The simplest data source is a system file. Anything can be a data repository: a set of files of different formats, an HTML page, a database, and their combinations.

Mappings: The key to repository heterogeneity encapsulation
Mapping Information relates terms in the ontology (concepts, roles) with data elements in data repositories.

The advantages of these mappings are as follows. The mappings subscribe to the idea of viewing a data repository as a set of entities and attributes (or relations and attributes in Extended Relational Algebra), independently of the specific organization of the data in the repository. This gives a homogeneous view of the description of the data repositories without capturing any characteristic specific to the individual data repositories.

They are expressive enough to capture the complex associations of concepts and roles with entities and attributes. They act as an intermediate language between the Description Logics (DL) expressions and the concrete query languages of the local repositories.

Main steps in accessing underlying data repositories

Step 1: Translation from DL into mapping.

Step2: Translation from mono-repository mappings into Local Query L and data access.

Step 3: Plan execution and retrieval of the final answer.

2.5.3 Semantic approach

Query Processing in JeromeDL: The search algorithm of JeromeDL consists of three major steps. Each step requires different metadata sources that describe resources in specific ways [55]:
Step A – the first step is the full text index search on the resources' contents and users' annotations on resources,

Step B – the next step is the bibliographic description search consisting of MARC21 and BibTEX formats,

Step C – the last step finally is a user-oriented search with semantics, based on the semantic description of the resources and information about most interested categories regarding the user that issued the query.

Query object: When issuing a query to JeromeDL a reader has to submit a query object , usually using an HTML-form. Each query contains several entries which state what information the search algorithm should look for in the resource description. The reader can choose from Dublin Core Metadata, MARC21-based and BibTeX-based properties. A special property that indicates the content of the resource is additionally provided. Each property contains the list of possible values that the reader expects to find in the description of the resource. Additionally each value may have a ranking value specified, so results containing a desired value are higher ranked. It is possible to define a maximum distance between words consisting the phrase value.

Result object: A result object contains the information about the resources that have been found and additional information on the query process that has been executed. Each of the resources is described with: the URI of the resource, title and authors, categorizations and keywords, summary – digest, information on the type of the resource, and the ranking of the resource in this result set. Additionally some debugging information and history of the query processing is also included in the result object.

JeromeDL's search algorithm was designed based on the following requirements [56]:

- The query should return resources where descriptions do not directly contain the required values.
- The meaning of values provided in the query should be resolved in the context of users' interests.
- These goals can be achieved by combining full text search as well as searching the bibliographic description and semantic descriptions of resources. The semantically enabled search phase includes query expansion based on the user's interests.

In order to measure the improvement of effectiveness of the semantic enabled search algorithm, the database of the prototype system has been filled with 100 resources. After a little time of browsing 50 queries have been processed with and without the semantic query expansion phase. To evaluate the gain in effectiveness produced by the semantic phase of the semantic searching process, tree metrics have been calculated: precision, recall and waste. The results have shown that the semantic query expansion phase in the search algorithm improves the results by 60% compared to the search process without the semantic phase.

The purpose of information retrieval is to assist users in locating information they are looking for. Information retrieval is currently being applied in a variety of application domains from database systems to web information search engines. The main idea is to locate documents that contain terms the users specify in their queries. Retrieval, by classical information retrieval models (eg. Vector Space, Probabilistic, Boolean), is based on plain lexicographic term matching between terms. However, plain lexicographic analysis and matching is not generally sufficient to determine if two terms are similar and consequently whether two documents are similar. Two terms can be lexicographically different but have the same meaning (eg. they are synonyms) or they may have approximately the same meaning (they are semantically similar).

The lack of common terms in two documents does not necessarily mean that the documents are irrelevant and vice-versa. Semantically similar terms or concepts may be expressed in different ways in the documents and the queries, and direct comparison is not effective (eg. Vector Space Model(VSM) will not recognize synonyms or semantically similar terms). It is proposed to discover semantically similar terms using ontology, and specifically WordNet.

Four main categories for determining semantic similarity between terms

1. Edge Counting Methods : These methods measure the similarity between two concepts *c*1, *c*2 by determining the path linking the terms in the taxonomy and the position of the terms in the taxonomy
2. Information Content Methods : In this category, similarity measures are based on the *Information content* of each concept
3. Feature based Methods : Measures that consider also the features of the terms in order to compute similarity
4. Hybrid methods : Those methods combine ideas from the above three approaches in order to compute semantic similarity between *c*1 and *c*2

Given two terms, the user selects a similarity method and the system returns the semantic similarity of the two terms. Several options are available

- Sense selection : The user has the option to select which sense of the term to compare (one specific or all senses)
- Similarity method selection : A list of 10 similarity methods is available from all categories with option to add new methods as easy as inserting a java class File in the system
- Ontology selection: The system is Ontology independent. The methods can be used within any ontology that conforms to a specific schema or that can be mapped to this schema
- Cross ontology similarity : Select two terms from different ontologies and compare them
- API : Furthermore a complete API was developed in order to be used from anyone who wants to make use of the system functionality

2.5.4 Linguistic approach

This paper introduces a simple unsupervised learning algorithm for recognizing synonyms. In order to recognize synonyms, given a problem word and a set of alternative words, it chooses the member from the set of alternative words that is similar in meaning to the problem word. The unsupervised learning algorithm performs this task by issuing queries to a search engine and analyzing the replies to the queries. The algorithm, called PMI-IR, uses Point wise Mutual Information (PMI) [50,51] to analyze statistical data collected by Information Retrieval (IR).Recognizing synonyms is often used as a test to measure a (human) student's mastery of a language. The performance is measured for PMI-IR using 80 synonym test questions from the Test of English as a Foreign Language (TOEFL) [52] and 50 synonym test questions from a collection of tests for students of English as a Second Language (ESL) [53]. PMI-IR obtains a score of 73.75% on the 80 TOEFL questions (59/80) and 74% on the 50 ESL questions (37/50). By comparison, the average score on the 80 TOEFL questions, for a large sample of applicants to US colleges from non-English speaking countries, was 64.5% (51.6/80) [54].Latent Semantic Analysis (LSA) is another unsupervised learning algorithm that has been applied to the task of recognizing synonyms. LSA achieves a score of 64.4% (51.5/80) on the 80 TOEFL questions [54]. PMI-IR may be suitable as a tool to aid in the construction of lexical databases. It might also be useful for improving IR systems. Fig.3 [64] shows the web query processing with natural language interface.

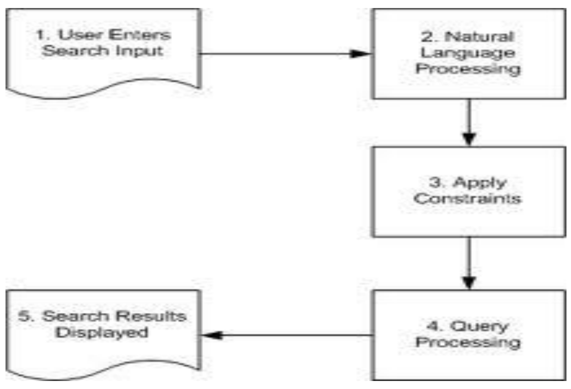
1. User Enters Search Input
2. Natural Language Processing
3. Apply Constraints
4. Query Processing
5. Search Results Displayed

Fig 3: Query processing with Natural Language Processing.

In [51], it particularly interested in applying PMI-IR to automatic keyword extraction [55]. One of the most helpful clues that a word (or phrase) is a keyword in a given document is the frequency of the word. However, authors often use synonyms, in order to avoid boring the reader with repetition. This is courteous for human readers, but it complicates automatic keyword extraction. PMI-IR will help to cluster synonyms together, so that one can aggregate their frequency counts, resulting in better keyword extraction.

This paper has introduced a simple unsupervised learning algorithm for recognizing synonyms. The algorithm uses a well-known measure of semantic similarity (PMI). The new contribution is the observation that PMI can exploit the huge document collections that are indexed by modern Web search engines. The algorithm is evaluated using the Test of English as a Foreign Language. The algorithm is compared with Latent Semantic Analysis, which has also been evaluated using TOEFL.

Also, XML-QL is a data model and query language for XML. XQL is a simpler query language based on single path expressions and is strictly less powerful than XML-QL, Lorel, StruQL, or UnQL.

2.5.5 Integration of the user and target ontologies

Two types of relationships are considered in order to integrate the user and the target ontology.
1. Synonym, hyponym and hypernym relationships between terms in the user and target ontologies.

2. Synonyms, hyponyms and hypernyms within the user and the target ontologies.

Plans with loss of information

After translation of the user query into the integrated ontology, there may be terms of the user ontology for which there did not exist synonyms into the target ontology. Each conflicting term in the user query is then replaced by the intersection of its immediate parents or by the union of its immediate children. This method is applied recursively until a translation of the conflicting term is obtained using only the terms of the target ontology. Note that it is always possible to get at least one full translation of any conflicting term in both the directions since the terms *Anything* and *Nothing* exist in any ontology. This procedure changes the semantics of the query resulting in loss of information in the answer returned to the user.

Example: Generation of plans with loss of information.

To demonstrate the different cases that can arise, let us consider a query by another user— "*Retrieve title and number of pages of all the books written by Carl Sagan*". This user chooses WN as user ontology and allows the system to provide imprecise answers and limits the maximum loss to 50%, i.e., using the query editor the user asserts that at least fifty percent of the objects retrieved should satisfy the constraints in the user query. This is the query in DL:

*Q = [NAME PAGES] for (***AND** *BOOK (***FILLS** *CREATOR "Carl Sagan"))*

After accessing the data underlying WN the user wants more data and the system chooses Stanford-I as target ontology. The user query has to be translated into terms of the Stanford-I ontology. After the process of integrating the WN and Stanford-I ontologies , *Q* was redefined as follows:

*Q = [title number-of-pages] for (***AND** *BOOK (***FILLS** *doc-author-name "Carl Sagan"))*

Hence, [49] attempts to enhance the scalability of query processing in a Global Information System and the same time enables the user to specify an information request in terms of semantic concepts. The novel contributions of this approach are:

- Use of pre-existing domain ontologies to describe information in the underlying data repositories. This enables the user to view the repositories at the level of its relevant semantic concepts. Thus an information request can now be expressed using these concepts and the user can now browse multiple domain ontologies as opposed to individual heterogeneous repositories or concepts based on statistical information.
- Use of brokering across domain ontologies to enhance the scalability of the query processing approach. Semantic relationships across terms in ontologies are stored and utilized to perform the brokering, thus avoiding the need to have a global schema or collection of concepts.
- Management of answers that have an associated loss of information as limited by the user and measurement of such a loss to provide answers with the least loss.

2.6 Web query optimizations on distributed web as well as on semantic web

Much of the data in XML are semi-structured, the data are irregular and incomplete and its structure may change while querying. While all of the usual problems associated with cost-based web query optimization apply to XML-based query languages, and to problems such as new kinds of indexing database statistics, query execution strategies for different databases. In [57], it is defined that appropriate logical and physical query plans, database statistics, and a cost-based model, and describe the plan of enumeration and heuristics for reducing the large search space. To transform a query into a logical query plan, and then explore the (exponential) space of possible physical plans looking for the one with least estimated cost, a number of factors associated with XML data complicate the problem. This paper [57] builds a new overall optimization framework for a number of reasons:

- Previous work has considered the optimization of single path expressions. This query language permits several, possibly interrelated, path expressions in a single query. It is developed an optimization framework that operates on a set of path expressions, as well as path expressions in the context of a complete query with other query constructs.
- The statistics maintained by relational DBMS's (for joins) and object-oriented DBMS's (for path expression evaluation) are generally based on single joining pairs or object references, while for accuracy in the environment it is essential to maintain more detailed statistics about complete paths.
- The capabilities of deployed OODBMS optimizers are fairly limited, and we know of no available prototype optimizer extensible enough to meet our needs.

Web services are becoming a standard method of sharing data and functionality among loosely-coupled systems. This paper [58] propose a general purpose Web Service Management System (WSMS) that enables querying multiple web services in a transparent and integrated fashion. It solves the problem namely, query optimization for Select-Project-Join queries spanning multiple web services. It develops an algorithm for arranging a query's web service calls into a pipelined execution plan that optimally exploits parallelism among web services to minimize the query's total running time.

The thesis [59] formally defines transformations that preserve semantic equivalence for queries in the relational calculus. In addition, it identifies several classes of cost-reducing query transformations for relational database queries, and provides quantitative estimates of the improvements they can produce, based upon widely accepted models of query processing. This can be applied to web query processing discussed here. Thus, semantic query optimization brings together the apparently separate research areas of query processing and database integrity.

A new statistical schema matching has been explored in [60]. Unlike traditional approaches using pairwise-attribute correspondence, given a set of input sources as observed schemas; it finds hidden models that are consistent, in a statistical sense, with the schemas observed. Further, [60] specializes the general framework to develop Algorithm MGS, targeting at synonym discovery, a canonical problem of schema matching, by designing and discovering a model that specifically captures synonym attributes.

This paper performed case studies for over 200 real sources in 4 domains: books (e.g., amazon.com), movies (e.g., blockbuster-.com), music records (e.g., towerrecords.com), and automobiles (e.g., cars.com). The goals are two-fold: (1) Verify the phenomenon's (of proliferating sources and converging vocabularies) that support our motivating hypotheses. (2) Validate the performance of MGS with two suites of metrics: model accuracy and target accuracy.

In summary, this approach takes a new view to cope with schema matching: (1) this is to study and report two distinguishing characteristics of databases on the deep Web, an important frontier for information integration. (2) It presents a new paradigm, hidden model discovery, for large-scale schema matching, realized by a general statistical framework MGS. (3) An Algorithm MGS was developed specifically for synonym discovery across Web query interfaces.

2.7 Use Context awareness in web query processing

A major impediment to accurate information retrieval from the World Wide Web is the inability of search engines to incorporate semantics in the search process. This research [28] presents a methodology, CONQUER (CONtext-aware QUERy processing), that enhances the semantic content of Web queries using two complementary knowledge sources: lexicons and ontologies. The contribution of this research [28] is a methodology to incorporate context into Web query processing that directly addresses part of the grand challenge of semantics for the Semantic Web. Specifically, the research provides: (1) a methodology for processing queries on the Web that incorporates semantics from existing lexical and ontological knowledge sources; (2) a design science artifact, the CONQUER (CONtext-aware QUERy processing) prototype, to improve query processing; and (3) an evaluation of the methodology to identify improvements to to query results and opportunities for future research.

The internal components include an interface, an inference engine that uses a local knowledge base, and a query constructor. The "interface" captures the user's query in natural language, transforms it for processing, and presents the results to the user. The algorithms are described below following the three phases namely,
Phase 1: Identify Query Context,

Task 1_ *Identify noun phrases*:

Task 2_ *Build synonym sets*:

Task 3_ *Decide intended word sense*:

Phase 2: Refine Query Context,

Task 4_ *Exclude unintended word senses*:

Task 5*: Including hypernyms and hyponyms*:

Task 6_ *Resolve inconsistencies*:

Phase 3: Execute Query

Task 7_ *Construct Boolean query*:

Task 8_ *Execute query on search engine*:

Task 9_ *Retrieve and present results*:

The methodology was implemented via J2EE technologies in a prototype (CONQUER) that interfaces with existing search engines.

The three main results are, first, this research demonstrates that it is possible to leverage existing knowledge sources (i.e., WordNet and DAML ontologies) to improve the processing of queries on the Web. Second, The CONQUER methodology is an alternative to such personalization, allowing the users to retain greater control over their personal details. Third, the CONQUER methodology has clear applications for searching within organizations.

In this paper [61], it is proposed a new supervised classification method based on a modified sparse model for action recognition. First, traditional interest-point-based utilizes features of a single interest point, and fails to capture adequate spatial or temporal information. It is sensitive to the noise. Second, it proposed a modified sparse model for classification. Different from traditional sparse representation whose only task is to minimize the reconstruction error, the proposed sparse model targets at classification. Third, it introduced a classification loss function for the class- specific dictionary learning. The dictionaries are trained by minimizing the classification loss function. It presents a modified sparse model and a supervised class-specific dictionary learning method for classification. Use of fuzzy sets theory in web query processing is demonstrated in [31].

3. COMPARISON OF WEB QUERY PROCESSING APPROACHES

In this survey, various important references papers are discussed. In [1], Yanlei Diao et al, discusses about how AI kind of learning intelligently and how the user traverses web pages to get a query result. This is done by first allowing the user to navigate with few URLs and the system is trained by this user behavior. Then the system applies the strategy of the user automatically and presents the user further results of the web query. The adaptive query processing with results through XML as internet data streams along with locally stored data were used in [2]. Data volume in streams and I/O transfer rates were used with optimization. Sharing and reusing of results in multiple queries were considered. Different taxonomies of web searches like navigational, transactional, etc. were discussed in [3].
We already classify the reference papers, which are used for this survey into seven categories in introduction chapter, namely
Category 1. Learning and Adaptive query processing.

Category 2. Web query through HTML and web search taxonomies.
Category 3. Web search query and search engines.

Category 4. Web query languages and its models.

Category 5. Semantic web and Ontologies.

Category 6. Web query optimizations on distributed web as well as on semantic web.
Category 7. Use Context awareness in web query processing.
Based on these seven categories, all the reference papers are tabulated based on the concepts used for query processing and survey categories in Table: 1 given below.

Table1: Tabulation of various web query research works in seven broad categories

Paper Reference Id	Concepts used	Survey Category
1,16,23 31 55,61	Artificial Intelligence Technique	1
2 32 33 42	Adaptive Query Processing, XML	1
3,34,35,44	Different Taxonomies	2
4,24 27 28 29,30,49, 60	Classification of Optimization Techniques	6
5	Geographic location based	2
6	Surface web and Deep Web	2
7 26	Several sources of input	2
8,45-48, 50-54	Language model	4
9	Batch query	3
10	Energy-Price optimization	6
11 37 38 39-41, 57-59	Adaptive Query Processing optimization	1
13	Sub graph and similarity query	5
14,15	Web query languages	4
17	Query with heterogeneous Databases	6
18	Web Query design and Architecture	6
12 19 20,21, 24, 25,56	Semantic web& Ontology	5
22	Federated Query Processing	7

Various classifications of optimization techniques and a framework for evaluating web data integration systems were discussed in [4]. Here, heuristics based, cost based and hybrid of both were focused on optimizing them. Focusing query on particular geographic location and its optimization is discussed in [5]. These search engines are called Geographic web search engines. Discovering automatically the instances for interface in Surface Web as well as Deep Web were considered in [6]. Since the web results are many and are from different sources, the end-user does not trust it when it is from a single source. Hence several sources are checked for results for corroborating evidence of the result [7].The language model used in web query to solve issues in query spelling correction, query bracketing and long query segmentation were discussed in [8]. The advantages of executing a query in batch over interactive mode were explained in [9]. Since energy prices and query workloads are highly varied spatio-temporally, significant financial savings were suggested through the research work [10].

In Database query optimization, optimize-then-execute kind of traditional optimization is not used now-a-days but a different type of query optimization in the form of adaptive query optimization is used [11]. This kind of adaptive query optimization can also be used for web query processing. According to the W3C, "The Semantic Web provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries [12]. The Semantic Web is integration of different content, information applications and systems. We can have semantic web in publishing, blogging, and many other areas. The synonyms for semantic web are "semantics", "metadata" and "ontologies". Web query processing is applied in semantic web or ontology. Graph is a powerful representation of any idea and the web query results can be represented sometimes as a graph and subgraph query and similarity queries in [13] can be applied here. Semantic web query languages in XML, RDF, Topic Map discussed in [14] can be analyzed to apply here. In computational linguistic query answering techniques are used with machine translation and is studied [15]. AI techniques are used in intelligent web query answering in heterogeneous databases in [16] and the semantic relations of heterogeneous databases and its query processed are discussed in [17].

Design and Architecture of improving web query processing is discussed in [18]. Further discussions on query processing and expansion of ontology are explained in [19] and use of Association rules in such processing is further discussed in [20]. The use of semantic knowledge in improving web query processing is given in [21]. Federated query is searching from multiple sources and it is discussed extensively in [22]. The exposure to distributed ontologies and using intelligent query in them is further explained in [23]. A cache based model with three tier architecture has been briefed for query optimization in distributed databases [24]. Semantic web approach to distributed databases was discussed in [25]. In Web query processing, interface design and its integration are discussed with a hierarchical model [26].

When query terms are ambiguous in heuristics methodology's use, the results are given to the users more relevantly with a heuristic approach [27]. Most importantly, sometimes the query answering must be in terms of the context in which is used. This is being discussed in [28]. The same concept is also discussed in [29]. A new method for learning query while modifying that can improve the accuracy of locating search pages with a specific category [30]. Use of soft computing techniques like fuzzy sets for expanding queries in search engines was discussed in [31].

4. CONCLUSION

To locate information across the Web, queries are used through search engines. For those who are accustomed to query and retrieve from databases will find that it is difficult to get similar query and result from the web. Also, results from a query in database will be straight forward whereas the web will return several similar web pages and the user has to search through it for the exact result. This survey has discussed and compared many papers some of them in detail and others in brief. These papers were classified into categories like Learning and Adaptive query processing, Web query through HTML and web search taxonomies, Web search query and search engines, Web query languages and its models, Semantic web and Ontologies, Web query

optimizations on distributed web as well as on semantic web and Use of context-based computing techniques in web query processing. Further we focus our future research into how web query impacts through adding synonyms and using natural language processing with the linguistic approach.
The proposed model described here with following characteristics.

1. Focuses the query in a specific area of interest
2. Expands the Query Vector with other semantic similar terms.
3. Re-weights the Query Terms based on synonyms.
4. Ranks the results according to Linguistic characteristics.
5. Use of soft-computing techniques in web query modeling

5. REFERENCES

[1] Yanlei Diao et. al, "Toward Learning Based Web Query Processing", Proceedings of the 26th International Conference on Very Large Databases, Cairo, Egypt, 2000.

[2] Zachary G. Ives," Adaptive Query Processing for Internet Applications", 2001.

[3] Andrei Broder, "A taxonomy of web search", SIGIR Forum 3 Fall 2002, Vol. 36, No. 2.

[4] MOURAD OUZZANI et. al, Query Processing and Optimization on the Web, Distributed and Parallel Databases, 15, 187–218, 2004.

[5] Yen-Yu Chen et. al, Efficient Query Processing in Geographic Web Search Engines, SIGMOD 2006, June 27–29, 2006, Chicago, Illinois, USA.

[6] Wensheng Wu, AnHai Doan et al, WebIQ: Learning from the Web to Match Deep-Web Query Interfaces, 2006.

[7] Minji Wu et al, Corroborating Answers from Multiple Web Sources, Proceedings of the 10th International Workshop on Web and Databases (WebDB 2007), June 15, 2007, Beijing, China

[8] Jian Huang, Exploring Web Scale Language Models for Search Query Processing, WWW 2010, April 26–30, 2010, Raleigh, North Carolina, USA.

[9] Shuai Ding et al, Batch Query Processing for Web Search Engines, WSDM'11, February 9–12, 2011, Hong Kong, China.

[10] Enver Kayaaslan et al, Energy-Price-Driven Query Processing in Multi-center Web Search Engines, SIGIR'11, July 24–28, 2011, Beijing, China.

[11] A. Deshpande, Z. Ives and V. Raman, Adaptive query processing, Foundations and Trends in Databases, Vol. 1, No. 1 (2007) 1–140.

[12] "W3C Semantic Web Activity". World Wide Web Consortium (W3C). November 7, 2011. Retrieved November 26, 2011.

[13] James Cheng, Efficient Query Processing on Graph Databases, ACM Transactions on Database Systems, Vol. V, No. N, September 2008, Pages 1–44.

[14] James Bailey, Web and Semantic Web Query Languages: A Survey.

[15] Stefan Riezler, Statistical Machine Translation for Query Expansion in Answer Retrieval, Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics, pages 464–471, Prague, Czech Republic, June 2007.

[16] Mohd Kamir Yusof, Implementing of XML and Intelligent Algorithm for Improving Web Query Processing in Heterogeneous Database Access, International Journal of Database Theory and Application Vol. 4, No. 2, June, 2011.

[17] Naphtali Rishe, Semantic Relations: The key to integrating and query processing in heterogeneous databases, Research funded by NASA and NSF.

[18] Mohd Kamir Yusof, Designing an Architecture for Improving Web Query Processing in Heterogeneous Databases Access, WIMS'11, May 25-27, 2011 Sogndal, Norway.

[19] J. Bhogal, A review of ontology based query expansion, Information Processing and Management 43 (2007) 866–886.

[20] Min Song, Integration of association rules and ontologies for semantic query expansion, Data & Knowledge Engineering 63 (2007) 63–75.

[21] Jordi Conesa, Improving web-query processing through semantic knowledge, Data & Knowledge Engineering 66 (2008) 18–34.

[22] Haifeng Jiang, Improving parallelism of federated query processing, Data & Knowledge Engineering 64 (2008) 511–533.

[23] Jihyun Lee, An intelligent query processing for distributed ontologies, The Journal of Systems and Software, June 2009.

[24] Neera Batra, Three tier cache based query optimization model in distributed database, International Journal of Engineering Science and Technology Vol. 2(7), 2010, 3206-3212.

[25] Mohd Kamir Yusof , Ontology and Semantic Web Approaches for Heterogeneous Database Access, International Journal of Database Theory and Application Vol. 4, No. 4, December, 2011.

[26] Eduard C. Dragut, A Hierarchical Approach to Model Web Query Interfaces for Web Source Integration, VLDB '09, August 24-28, 2009, Lyon, France.

[27] Andrew Burton-Jones, A Heuristic-Based Methodology for Semantic Augmentation of User Queries on the Web, ER 2003, LNCS 2813, pp. 476–489, 2003.

[28] Veda C. Storey, CONQUER: A Methodology for Context-Aware Query Processing on the World Wide Web, Information Systems Research, Vol. 19, No. 1, March 2008, pp. 3–25.

[29] Steve Lawrence, Context in Web Search, IEEE Data Engineering Bulletin, Volume 23, Number 3, pp. 25–32, 2000.

[30] Eric J. Glover, Improving Category Specific Web Search by Learning Query Modifications, IEEE, 2001.

[31] Tomohiro TAKAGI, Query Expansion Using Conceptual Fuzzy Sets For Search Engine, lEEE International Fuzzy Systems Conference, 2001.

[32] M. Balabanovic. An Adaptive Web Page Recommendation Service. In Proc. of 1st International Conference on Autonomous Agents, pp. 378-385, 1997.

[33] F. Menczer, R. Belew. Adaptive Retrieval Agents: Internalizing Local Context and Scaling up to the Web. Technical Report CS98-579, University of California, San Diego, 1998. Available: http://www.cse.ucsd.edu/~rik/papers/arachnid/arachnd-mlj.ps.

[34] M. Pazzani, J. Muramatsu, D. Billsus. Syskill & Webert: Identifying Interesting Web Sites. In Proc. of AAAI-96, pp. 54-61, 1996.

[35] T. Joachims, D. Freitag, T. Mitchell. WebWatcher: A Tour Guide for the World Wide Web. In Proc. of the 1997 International Joint Conference on Artificial Intelligence, pp. 770-775, Aug. 1997.

[36] V. Barnett and T. Lewis. Outliers in Statistical Data. John Wiley & Sons, 1994.

[37] A. N. Wilschut and P. M. G. Apers, "Dataflow query execution in a parallel main-memory environment," in PDIS '91: Proceedings of the First International Conference on Parallel and Distributed Information Systems, Fontainebleu Hilton Resort, Miami Beach, FL, pp. 68–77, IEEE Computer Society, 1991.

[38] G. Graefe, "Query evaluation techniques for large databases," ACM Comput. Surv, vol. 25, no. 2, pp. 73–169, 1993.

[39] D. Kossmann, "The state of the art in distributed query processing," ACM Comput. Surv, vol. 32, no. 4, pp. 422–469, 2000.

[40] S. Chaudhuri, "An overview of query optimization in relational systems," in PODS '98: Proceedings of the seventeenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems, (New York, NY, USA), pp. 34–43, ACM Press, 1998.

[41] Y. E. Ioannidis, "Query optimization," ACM Computing Surveys, vol. 28, no. 1, pp. 121–123, 1996.

[42] S. Babu and P. Bizarro, "Adaptive query processing in the looking glass," in CIDR '05: Second Biennial Conference on Innovative Data Systems Research, pp. 238–249, Asilomar, CA, 2005.

[43] J. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Scholkopf, C. Burges, and A. Smola, editors, Advances in kernel methods - support vector learning. MIT Press, 1998.

[44] J. Jackson. Pop Goes the Interstitial. eMarketeer, 7 June 2001, Available at http://www.emarketer.com/analysis/eadvertising/20010607_ead.html

[45] P.Turney, Mining the Web for Synonyms: PMI-IR Versus LSA on TOEFL, Proceedings of the Twelth European Conference on Machine Learning (ECML-2001), Freiburg, Germany. September 3–7, 2001. pp. 491–502. NRC 44893.

[46] Fellbaum, C. (ed.): WordNet: An Electronic Lexical Database. Cambridge, Massachusetts: MIT Press (1998). For more information: http://www.cogsci.princeton.edu/~wn/.

[47] Haase, K.: Interlingual BRICO. IBM Systems Journal, 39 (2000) 589-596. For more information: http://www.framerd.org/brico/.

[48] Vossen, P. (ed.): EuroWordNet: A Multilingual Database with Lexical Semantic Networks. Dordrecht, Netherlands: Kluwer (1998). See: http://www.hum.uva.nl/~ewn/.

[49] Eduardo Mena, OBSERVER-An approach for query processing, Distributed and Parallel Databases, 8, 223-271, 2000.

[50] Church, K.W., Hanks, P.: Word Association Norms, Mutual Information and Lexicography. In: Proceedings of the 27th Annual Conference of the Association of Computational Linguistics, (1989) 76-83.

[51] Church, K.W., Gale, W., Hanks, P., Hindle, D.: Using Statistics in Lexical Analysis. In: Uri Zernik (ed.), Lexical Acquisition: Exploiting On-Line Resources to Build a Lexicon. New Jersey: Lawrence Erlbaum (1991) 115-164.

[52] Test of English as a Foreign Language (TOEFL), Educational Testing Service, Princeton, New Jersey, http://www.ets.org/.

[53] Tatsuki, D.: Basic 2000 Words - Synonym Match 1. In: Interactive JavaScript Quizzes forESLStudents,http://www.aitech.ac.jp/~iteslj/quizzes/js/dt/mc-2000-01syn.html(1998).

[54] Landauer, T.K., Dumais, S.T.: A Solution to Plato's Problem: The Latent Semantic Analysis Theory of the Acquisition, Induction, and Representation of Knowledge. Psychological Review, 104 (1997) 211-240.

[55] Turney, P.D.: Learning Algorithms for Keyphrase Extraction. Information Retrieval, 2 (2000) 303-336.

[56] Sebastian Ryszard Kruk, JeromeDL - Adding Semantic Web Technologies to Digital Libraries, JeromeDL - e-Library with Semantics: http://www.jeromedl.org/

[57] Jason Mchuge, Query optimizations in XML, Rome Laboratories, Stanford University.

[58] Utkarsh Srivastava, Query Optimization over Web Services, VLDB '06, September 12-15, 2006, Seoul, Korea.

[59] King, Jonathan Jay, Query Optimization by Semantic Reasoning, Doctoral Thesis, STANFORD UNIV CA DEPT OF COMPUTER SCIENCE, 1981.

[60] Bin He, Statistical Schema Matching across Web Query Interfaces, SIGMOD 2003, June 912, 2003, San Diego, CA.

[61] Haoran Wang Supervised class-specific dictionary learning for sparse modeling in action recognition, National Laboratory of Pattern Recognition, Institute of Automation, CAS, 2012, Beijing, China.

[62] Figure 1 referred from TempVars in Web Query social.msdn.microsoft.com

[63] Figure 2 referred from WebQuery: Searching and Visualizing the Web Through Connectivity www.ra.ethz.ch

[64] Figure 3 referred from Oracle ATG Web Commerce Query Processing Overview docs.oracle.com