CHAPTER 2

LINGUISTIC MODELS IN WEB CRAWLING

2.1 INTRODUCTION

Crawling has been the subject of widespread research and presently web crawling in diverse aspects has been studied. In this chapter, the web crawling techniques are reviewed. The seed page is the most important page for extracting the relevant topic information. Initially, the query words are input to the search engines such as Google, Yahoo and MSN search, so that the seed URL is extracted. Then, the outgoing links are obtained from the seed URL utilizing hypertext analysis. An assumption is made that the resulting URLs that are common in all three search engines are more relevant for the query and thus these common search result URLs are considered as seed URLs and they belong to most relevant group seed URLs. The resulting URLs that are common in any two search engines are not most relevant but they are relevant for topics and hence they are also placed in relevant group (Debashis and Amritesh 2010). Subsequently, it is assumed in the proposed algorithms that the resulting URLs in any two search engines are also placed in the relevant group and are applied to find the relevancy of the pages.

2.2 REVIEW OF WEB CRAWLING TECHNIQUES

Ample researches are proposed by researchers for the categorization of web documents and web crawling processes. Information retrieval (IR) finds most of the documents relevant to a user query in a given collection of documents. The most essential component of an IR system is that which addresses the problem of
effectively and efficiently searching and retrieving information from the Web. Retrieval algorithms use not only the words in the documents but also the information like hyperlink structure of the web or the markup language tags. Information retrieval is concerned with the question of how to identify relevant documents from a document collection and to produce structured data for post-processing. Information extraction is becoming an important task due to the vast growth of the online information. The process of selectively structuring and combining data that are explicitly stated or implied in one or more documents represents extraction of information. It addresses the problem of extracting specific information from a collection of documents. The detailed steps of web information extraction, such as web page organization and rule generation are described by Yan and Chenyue (2011). It is good to judge the relevance of the crawl by human inspection, even though it is subjective and inconsistent. One way to help in the search is by grouping HTML pages together that appear in some way to be related. In order to better understand the task, an initial study of human clustering of web pages was performed which provides some insight into the difficulty of automating the task (Macskassy et al 1998).

For information retrieval systems, the quality of searching is most important. There are many techniques to improve the quality of search. Among them the query expansion is considered in order to improve the search result in terms of recall and user satisfaction. Maleerat et al (2013) proposed the query expansion technique using keyword-based query. The semantic terms are added to the original query in order to reformulate query before sending to the searching process. The experiments are tested on twitter data collection. The results show that the retrieval effectiveness is considerably higher than the use of only original query and the method provides higher performance in terms of recall and precision. Query expansion analysis is separated into local analysis and global analysis. In local analysis technique, the results set of initial query is extracted in order to get an expansion term or set document as relevant. Global analysis technique is built by adding a suitable word into the original query to assist the users in reformulating their
query. In order to search with partial matching, not just for exact matching with query expression, document representation as the vector space model is considered.

One of the first dynamic search heuristics was the “fish search” (De Bra et al 1994). The method assumes that the browsing session is always large enough to benefit most from a depth-first strategy. The school moves on, rather than stays in the same neighborhood for too long and risks dying of starvation. Generally, in a certain water area, the area most abundant of nutriment lies where most fish alive. Based on this character, the fish-search algorithm optimizes the feeding behavior of fish school by simulating. Feeding behavior and flocking behavior are the basic behaviors of fish school. Feeding behavior of fish school, while swimming freely, could be regarded as moving at random. Once they find the food, they will move quickly in the direction of food. Flocking behavior of fish school, fish would gather automatically while swimming, will guarantee fish school survival and escaping from danger. There are foods for a number of fish, not just a single one, meaning relevant nodes are usually clustered. This capitalizes on the intuition that relevant documents often have relevant neighbors. Flocking character of fish school searches the parallel hypertext, which could increase search speed enormously. The algorithm retrieves the new webpage and hyperlink from “seed” webpage collected by user. Internet is treated as a directed graph, webpage as node and hyperlink as edge, so the search operation could be abstracted as a process of traversing directed graph. The algorithm maintains a list, which keeps URL of page to be searched. The URLs have different priority, the URL with superior priority is located at the front of the list, and is searched sooner than others. The key point of the algorithm lies in the maintenance of URLs order.

The key principle of the fish-search algorithm are: it takes as input a seed URL and a search query, and dynamically builds a priority list initialized to the seed URL of the next URLs called nodes. At each step the first node is popped from the list and processed. As each document’s text becomes available, it is analyzed by a scoring component evaluating whether it is relevant or irrelevant to the search query (1-0
value) and based on the score, a heuristic decides whether to pursue the exploration in that direction or not. Whenever a document source is fetched, it is scanned for links. The nodes pointed by these links denoted as children are assigned a depth value. If the parent is relevant, the depth of the children is set to some predefined value. Otherwise, the depth of the children is set to be one less than the depth of the parent. When the depth reaches zero, the direction is dropped and none of its children is inserted into the list.

Searching for information on the web requires three types of actions (De Bra et al 1994):

- The first step is finding a good starting point for the search. One possibility is to use a well-connected document.
- Documents are retrieved over the internet and scanned for relevant information at the receiving (client) end.
- The retrieved documents are scanned to find links to URLs of other web documents.

The effectiveness of a search algorithm depends largely on the ability to find relevant documents and to avoid downloading irrelevant documents. The best possible result of the search algorithm is a subset of the documents in web that contain the requested information. Typically, some relevant documents can be found, provided a “good” starting point is chosen.

Based on the value of potential score, Fish-search algorithm changes the order in the list. As a result, fish-search algorithm changes the order of page searching, realizes the preferential treat instead of searching one by one according to the location order of URLs in the page. A famous dynamic web search algorithm, the fish-search is analyzed and its merits contrasted with general depth-first algorithm, and it points out that the random of search range could lead to repeated search or
overlong search time (Fang-fang et al 2005). The analysis of flocking behavior of fish school, a parameter “dist” is defined to control the search range and time. The Fang-fang et al method presents an improved fish-search algorithm, in which the search process could adjust the search range adaptively. Simulation shows some improvements over the original fish-search algorithm. The reformatory fish-search algorithm focuses on the fuzzy reformation of two value (0, 1) setting. In this way, it accords with the real-world all the more and the relative of searching result is improved. Based on the analysis of the fish school’s behavioral character, this method introduces the graph theory to control the direction of every fish in the searching process. Two fish moving in different direction can be regarded as two directed graphs. The search direction is controlled with the distance between the two directed graphs, the distance stands for the searching range where the two fish had searched. There should be reasonable distance between the two fish, neither too close not too far; if it is too close it will lead to crossover repeated search, if it is too far it will make the search algorithms run slowly because of the wide searching range. The searching time and range can be controlled efficiently, by setting the distance between the two fish. The distance of the two directed graphs can be regarded as the distance between the centers of two directed graphs. The setting of distance differs according to the extension of query cluster. The experiments indicated that the reformatory algorithm improves the searching efficiency to a certain degree. The algorithm is implemented with vc6.0. In order to increase searching speed, the multi-thread technology is adopted to search different site simultaneously, this way can make full use of the hardware resource and net source. User interface has an inter-communicate environment, user can tailor some search parameter freely.

The metrics most commonly used by the crawler are precision, recall and accuracy determination (Powers and David 2011). These metrics are based on the following facts namely; total number of pages crawled, number of relevant pages retrieved, number of irrelevant pages retrieved, number of relevant pages not retrieved and the number of irrelevant pages not retrieved. Precision is the number of relevant
documents retrieved by a search divided by the total number of documents retrieved by the search; Recall is the number of relevant documents retrieved by a search divided by the total number of existing relevant documents. Accuracy is the proportion of true results in the total number of pages crawled. High precision is that an algorithm returned substantially more relevant results than irrelevant, while high recall is an algorithm returned most of the relevant results. Accuracy is the overall correctness of the model.

2.2.1 Web Crawler

Web crawler also known as a web spider or web robot is a program or automated script which browses the World Wide Web in a methodical and automated manner. Many legitimate sites, in particular search engines, use spider as a means of providing up-to-date data. Web crawlers are mainly used to create a copy of all the visited pages for later processing by a search engine, which will index the downloaded pages to provide fast searches. Crawlers can also be used for automating maintenance tasks on a web site, such as checking links or validating HTML codes. A web crawler is one type of bot, or software agent. In general, it starts with a list of URLs to visit, called the seeds. As the crawler visits these URLs, it identifies all the hyperlinks in the page and adds them to the list of URLs to visit, the crawl frontier (Shalin 2006). A typical web crawler starts by parsing a specified web page, noting any hypertext links on that page that point to other web pages. The crawler then parses those pages for new links, and so on, recursively. It supports the crawler to collect as many relevant web pages as possible. A crawler is a software or script or automated program which resides on a single machine. The crawler sends HTTP requests for documents to other machines on the internet, just as a web browser does. There are two important characteristics of the web that generate a scenario in which web crawling is difficult:
- A large volume of web page implies that web crawler can only download a fraction of the web pages and hence it is very essential that web crawler should be intelligent enough to prioritize download.

- The web pages on the internet change very frequently, as a result, by the time the crawler is downloading the last page from a site, the page may change or a new page may be placed / updated at the site.

So, it is becoming essential to crawl the web in not only a scalable, but efficient way, if some reasonable amount of quality or freshness of web pages is to be maintained. This ensures that a crawler must carefully choose at each step which pages to visit next. Thus the implementer of a web crawler must define its behavior. Defining the behavior of a web crawler is the outcome of a combination of the strategies: selecting the right algorithm to decide which page to download, strategizing how to re-visit pages to check for updates and strategizing how to avoid overloading websites.

The number of pages on the web is infinite, and a web crawler cannot download all the pages. Normally, a document having its content related to a topic is composed of a set of keywords which frequently appear in the topic. Topic keywords support the crawler to recognize appropriate keywords matching the topic. This helps the crawler to capture the most important pages as early as possible and a crawler should be intelligent enough to prioritize the downloaded pages. The most important pages have many links to them from numerous hosts, and those links will be found early, regardless of on which host or page the crawl originates. Increasing the coverage of existing search engines would pose a number of technical challenges, both with respect to their ability to discover, download and index the web pages, as well as their ability to serve queries against an index of that size. Hence, search engines should attempt to download the best pages and include them in their index. Web crawler creates a collection which is indexed and searched. Discovering high
quality pages early on a crawl is desirable for public web search engines such as Yahoo or Google, given that none of these search engines is able to crawl and index more than a fraction of the web.

The connectivity-based metric pagerank is used to measure the quality of a page (Marc Najork and Janet Wiener 2001). The algorithm shows that traversing the web graph in breadth-first search order is a good crawling strategy, and it tends to discover the high quality pages early in the crawl. The method extends the results of Cho et al (1998) regarding the effectiveness of crawling in breadth-first search order, and the work was based on a crawl of more numbers of pages from Stanford.edu domain. The connectivity-based page quality metrics is used, namely Brin and Page’s pagerank and variations of it, to measure the quality of downloaded pages over the life of the crawl. Not only the breadth-first search downloads the hot pages first, but also the average quality of the pages decreased over the duration of the crawl. It also increases the overall download rate and to avoid overloading any given web server. The authors modified pagerank slightly so that pages with no outgoing links contribute their weight equally to all pages. That is, the random surfer is equally likely to jump to any page from a page with no outgoing links. Two tools are used, namely: Mercator to crawl the web, and the Connectivity Server2 to provide fast access to the link information downloaded from the crawl. Breadth-first search is speculated as a good crawling strategy because the most important pages have many links to them from numerous hosts, and those links will be found early, regardless of on which host or page the crawl originates. On the other hand, crawling in breadth-first search order provides a fairly good bias towards high quality pages without the computational cost.

Issues related to crawler design are (Sotiris et al 2009):

- **Input:** Crawlers take as input a number of starting (seed) URLs and the topic description. This description can be a list of keywords.
• **Page downloading:** The links in downloaded pages are extracted and placed in a queue. A non focused crawler uses these links and proceeds with downloading new pages in a first in, first out manner. A focused crawler reorders queue entries by applying content relevance or importance criteria or may decide to exclude a link from further expansion.

• **Content processing:** Downloaded pages are lexically analyzed and reduced into term vectors, all terms are reduced to their morphological roots by applying a stemming algorithm and stop words are removed. Each term in a vector is represented by its term frequency-inverse frequency vector (tf-idf) according to vector space model.

• **Priority assignment:** Extracted URLs from downloaded pages are placed in a priority queue where priorities are determined based on the type of crawler and user preferences. They range from simple criteria such as page importance or relevance to query topic.

• **Expansion:** URLs are selected for further expansion and the above steps from page downloading are repeated until some criteria (e.g., the desire numbers of pages have been downloaded) are satisfied or system resources are exhausted.

Cho et al (1998) suggested using connectivity-based document quality metrics directs a crawler towards high quality pages. They used different ordering metrics: breadth-first, backlink count, PageRank and random to direct the different crawls. Under the breadth-first ordering, pages are crawled in the order they are discovered. In the backlink ordering, the pages with the highest number of known links to them are crawled first. In PageRank ordering, pages with the highest page rank are crawled first. Under the random ordering, the crawler selects the next page to download at random from the set of uncrawled pages. Cho et al evaluated the
effectiveness of each ordering metric by examining how fast it led the crawler to all
the “hot” pages. Here, a hot page is a page with either a high number of links pointing
to it, or a page with a high page rank. They found that using the PageRank metric to
direct a crawler works extremely well. However, they also discovered that performing
the crawl in breadth-first order works almost as well, in particular if hot pages are
defined to be pages with high pagerank.

2.2.2 Hyperlink Analysis on the Web

Hyperlink analysis significantly improves the relevance of the search
results, so that all major web search engines claim to use some type of hyperlink
analysis. A hyperlink (or link) points to a whole document or to a specific element
within a document. Hyperlink analysis provides a means for judging the quality of
pages. Hyperlink analysis algorithms make either one or both of the following
simplifying assumptions:

Assumption 1: A hyperlink from page A to page B is a recommendation
of page B by the author of page A.

Assumption 2: If page A and page B are connected by a hyperlink, then
they might be on the same topic.

The two main uses of hyperlink analysis in web information retrieval are
crawling and ranking. First, to distill a large web search topic to a size that makes
sense to a human user, need a means of identifying the topics most definitive or
authoritative web pages. The need is to locate not only a set of relevant pages, but also
those relevant pages of the highest quality. Second, the web consists not only of
pages, but hyperlinks that connect one page to another. This hyperlink structure
contains an enormous amount of latent human annotation that can help automatically
infer notions of authority (Chakrabarti et al 1999c). A collection V of hyperlinked
pages is viewed as a directed graph G = (V, E); the nodes correspond to the pages,
and a directed edge \((p,q) \in E\) indicates the presence of a link from \(p\) to \(q\). Currently, page importance is calculated by using the link graph of the web and such a process is link analysis. Link analysis algorithms have been successfully applied to web search. Link analysis algorithms that are robust against link spam have been proposed.

TrustRank (Zoltan Gyongyi et al 2004) is a link analysis technique which takes into consideration the reliability of web pages when calculating the importance of pages. The term web spam refers to hyperlinked pages on the web that are created with the intention of misleading search engines. Web spam pages use various techniques to achieve higher-than-deserved rankings in a search engine results. While human experts can identify spam, it is too expensive to manually evaluate a large number of pages. Instead, they propose techniques to semi-automatically separate reputable, good pages from spam. Here, first a small set of seed pages to be evaluated by an expert are selected. Once manually the reputable seed pages are identified, the link structure of the web is used to discover other pages that are likely to be good. The algorithm discusses possible ways to implement the seed selection and the discovery of good pages. Since the algorithmic identification of spam is very difficult, the schemes do not operate entirely without human assistance. The algorithm receives human assistance as follows: The algorithm first selects a small seed set of pages whose spam status needs to be determined. A human expert then examines the seed pages, and tells the algorithm if they are spam (bad pages) or not (good pages). Finally, the algorithm identifies other pages that are likely to be good based on their connectivity with the good seed pages. The trust of the seed pages is propagated to other pages on the web link graph. Since the propagation in TrustRank starts from the reliable pages, TrustRank can be more spam-resistant than PageRank. The results of experiments run on the web indexed by AltaVista are presented. The results show that they effectively filter out spam from a significant fraction of the web, based on a good seed set. Two commonly used metrics namely; precision and recall metrics are used for a certain threshold value to evaluate the retrieved web pages. In a search engine,
TrustRank can be used either separately to filter the index, or in combination with PageRank and other metrics to rank search results.

The advances in graph-based search techniques derived from Kleinberg’s work (1998) are impressive. The Persona web search (Francisco and Lik 2002) improves the graph based search algorithm in two dimensions. The variants of Kleinberg techniques do not take into account the semantics of the query string nor of the nodes being searched. As a result, polysemy of query words cannot be resolved. This method presents an interactive query scheme utilizing the simple web taxonomy provided by the ODP to resolve meanings of a user query. Simulation results are presented to illustrate the sensitivity of the technique. The authors outline the implementation of the algorithm in the Persona personalized web search system.

Document categorization is one of the foundational problems in web information retrieval. Even though the web documents are hyperlinked, the most proposed classification techniques take little advantage of the link structure and rely primarily on text features (Gyongyi et al 2006), as it is not immediately clear how to make link information intelligible to supervised machine learning algorithms. The method introduces a link-based approach to classification, which can be used in isolation or in conjunction with text-based classification. Various large-scale experimental results indicate that link based classification are on par with text based classification, and the combination of the two offers the best of both worlds. The two sources of data for the experiments were the Yahoo! web graph and the Open Directory. The open directory is a hierarchy of topics organized as a tree, where tree nodes contain links to web pages relevant for the corresponding topic.

Most traditional web search engines are based on keyword matching and return results containing the query terms in the document. In recent years, web-based hyperlink topology of the web search algorithms attracted extensive attention of many scholars, become more popular web search technology. Among the many findings, the
better known algorithms are PageRank and HITS algorithm. Although the calculation of pagerank algorithm is simple and very efficient, to complete a hyperlink analysis can be used to sort all the results, but it completely ignores the content of the website. HITS algorithm is to use the page number and number of links referenced to determine the value of different pages. This method can get a higher recall rate, but ignoring the text, prone to drift them. An algorithm based on HITS algorithm, combined with hyperlinks and content relevance strategy proposed a new Clonal Selected Hperlink Induced Topic Search (CSHITS) (Lili Yan 2012) which performs well in topic-specific crawling.

Measuring similarity between objects is a fundamental task in information retrieval. Link-based similarity measures have attracted the attention of many researchers and have been widely applied in recent years. However, most previous works mainly focus on introducing new link-based measures, and seldom provide theoretical as well as experimental comparisons with other measures. A comprehensive analysis and critical comparison of various link-based similarity measures and algorithms are presented by Liu et al (2013). Their strength and weakness are discussed. The goal of the link based crawling is to predict the category of a web page based on the number of keywords on the outgoing link.

The different algorithms used for link analysis like PageRank, Weighted PageRank, HITS and Clever algorithms are discussed and compared by Ashish Jain et al (2013). The method discovers an efficient and better system for mining the web topology to identify authoritative web pages. The mentioned algorithms provide satisfactory performance in some cases but many times the user may not get the relevant information. The problem when the user search a topic in the web using a search engine like Google is that the user is presented with millions of search results. First of all it is not practically feasible to visit all these millions of web pages to find the required information. Second, when the user visits few initial links shown in the search results, he may not get the relevant information. To overcome these problems
is to include the Intelligent Search Method, which means that there is a need for interpreting the inherent meaning of the query and indexing should be based on that. This new method “Intelligent Search Method” (ISM) to be proposed in future is on research by Ashish et al. This method can be integrated with any of the page ranking algorithms to produce better and relevant search results.

The algorithms given by Jeffrey and Monika (1999) discusses a different approach to web searching where the input to the search process is not a set of query terms, but instead is the URL of a page, and the output is a set of related web pages. A related web page is one that addresses the same topic as the original page. They describe two algorithms to identify related web pages. These algorithms use only the connectivity information on the web (i.e., the links between pages) and not the content of pages or usage information. This algorithm describes the Companion and Cocitation algorithms, two algorithms which use only the hyperlink structure of the web to identify related web pages. These algorithms use only the information about the links that appear on each page and the order in which the links appear. They neither examine the content of pages, nor do they examine patterns of how users tend to navigate among pages. The Companion algorithm is derived from the HITS algorithm. The algorithm is extended to exploit not only links but also their order on a page and present the results of a user-study. The Cocitation algorithm finds pages that are frequently cocited with the input URL u that is, it finds other pages that are pointed to by many other pages that also point to u. For finding related pages is to examine the siblings of a starting node u in the web graph. Two nodes are cocited if they have a common parent. The number of common parents of two nodes is their degree of cocitation. For a given URL u, the average precision for u of an algorithm is the sum of all the average precisions for the entire query URLs divided by the total number of query URLs. The average precision of the companion algorithm is 57% better and the cocitation algorithm is 51% better than that of Netscape algorithm (Netscape Corpn.). The ‘What’s Related’ algorithm in Netscape webpage computes
its answers based on connectivity information, content information and usage information.

While the search engine could retrieve information on the web for a specific topic, users have to step a long ordered list in order to locate the needed information, which is often tedious and less efficient. A link-based clustering approach (Yitong and Masaru 2002) is proposed to cluster search results returned from web search engine by exploring both co-citation and coupling. Unlike document clustering algorithms which are based on common words / phrases shared among documents, this approach is based on common links shared by pages. It also extends standard clustering algorithm, k-means, to make it more natural to handle noises and apply to web search results. By filtering some irrelevant pages, the approach clusters high quality pages in web search results into semantically meaningful groups to facilitate users accessing and browsing. The idea is, pages that share common links with each other are very likely to be tightly related. Here, common links for two web pages p and q mean common out-links (point from p and q) as well as common in-links (point to p and q). Three metrics namely, precision, recall and entropy are used to evaluate quality of final clusters. They manually check 200 web pages for each of six topics and mark each one page with ‘relevant’ or ‘irrelevant’ to indicate whether it is relevant to the corresponding query topic. Entropy provides a measure of goodness or purity for un-nested clusters by comparing the groups produced by the clustering technique to known classes.

The importance of a web page is an inherently subjective matter, which depends on the readers’ interests, knowledge and attitudes. But there is still much that can be said objectively about the relative importance of web pages. Lawrence Page et al (1998) describes pagerank, a method for rating web pages objectively and mechanically, effectively measuring the human interest and attention devoted to them. They show how to efficiently compute pagerank for large numbers of pages. The method considers the link structure of the web to produce a global importance ranking
of every web page based on the graph of the web. Pagerank has applications in search, browsing and traffic estimation. They implemented two search engines which use pagerank for searching. The first one is the simple title-based search engine. The second is a full text search engine called Google. Google utilizes a number of factors to rank search results including standard retrieval measures, proximity, anchor text and pagerank. The original goal of pagerank was a way to sort back links so if there were a large number of back links for a document, the best back links could be displayed first. Pagerank is a global ranking of all web pages, regardless of their content, based solely on their location in the web’s graph structure. Using pagerank, search results are ordered so that more important and central web pages are given preference. In experiments, this tends to provide higher quality search results to users.

Haveliwala (1999) discusses efficient techniques for computing pagerank which is a ranking metric for hypertext documents. They show that pagerank can be computed for very large sub graphs of the web, up to hundreds of millions of nodes on machines with limited main memory. The method discusses several methods for analyzing the convergence of pagerank based on the induced ordering of the pages and also presents convergence results helpful for determining the number of iterations necessary to achieve a useful pagerank assignment, both in the absence and presence of search queries. When computing pagerank, either single-precision or double-precision values are used for the source and destination arrays. Using double-precision, however, would adversely affect the running time by doubling the sizes of the two vectors. Hence, it is shown that sing-precisions vectors are sufficient for computation and does not lead to significant numerical error. It is shown that an accurate pagerank vector can be computed in as few as 10 iterations, if accuracy is measured in the context of the induced ordering of results to conjunctive search queries. This convergence rate implies that a useful pagerank vector can be computed on a modestly equipped machine in roughly an hour, demonstrating the feasibility of client-side computation of personalized pagerank.
The pagerank and HITS algorithms tend to over-rank tightly interlinked collections of pages, such as well organized message boards. This effect can be alleviated using a number of modifications to the underlying Markov process. Specifically, rather than weight all out links from a given page equally, greater weight is given to links between pages that are in other respects, further off in the web and less weight is given to links between pages that are nearby (Ziyang Wang 2003). This was experimented with a number of variants of this idea, using a number of different measures of “distance” in the web, and a number of different weighting schemes. These revised methods often do avoid the over-ranking problem and give better overall rankings. To overcome the effect of local aggregation and improve the effectiveness of ranking systems, they propose the idea of hyperlink evaluation and evaluation-based web ranking.

2.2.2.1 Page Importance in Retrieval

Page importance, which represents the ‘value’ of an individual page on the web, is a key factor for web search, because for contemporary search engines, the crawling, indexing and ranking are usually guided by this measure. Because the scale of the web is extremely large and the web evolves dynamically, accurately calculating the importance scores of web pages becomes critical and also poses a great challenge to search engines. Most of the algorithms assume that if many important pages link to a page on the link graph, then the page is also likely to be important, and they calculate the importance of the page on the basis of a model defined on the link graph. Link analysis algorithms have been successfully applied to web search.

In general, a page’s importance must be measured in order to rank it. Three approaches help with this process: link, content (similarity) and anchor (Jaroslav 2004). In terms of information retrieval, these measures reflect a model of web documents. The best-known link-based technique used on the web today is a variant of the pagerank algorithm implemented in the Google search engine. It tries to
infer a web pages importance from just the topological structure of a directed graph associated with the web. A pages rank depends on the ranks of all the pages pointing to it, with each rank divided by the number of out-links those pages have. The page rank of a page is a non-negative real number. Another technique is HITS – is used at query time and processed on a small subset of relevant documents, but not all of them. It computes two scores per document namely authoritative pages and hub pages. In the content-based approach, the similarity score is computed between a page and a predefined topic in a way similar to the vector model. Topic vector q is constructed from a sample of pages, and each web page has its own vector p. The similarity score Sim(p,q) is defined by the cosine similarity measure. Another text is the visible hyperlinked text on the web page. In the anchor-based approach, page quality can be judged by pattern matching between the query vector and the URLs anchor text, the text around the anchor text and the URLs string value. The approach relying on text near anchors seems to be the most useful for web similarity-search tasks. Most search engines perform their tasks by using important keywords. Moreover, the user might submit a query with additional constraints such as searching a specific web page or finding the pages within a web graph structure.

Pant et al (2004) discuss the basic issues related to developing an infrastructure for crawlers. It is followed by a review of several topical crawling algorithms and evaluation metrics that are used to judge the performance. Some of the methods that are used to measure page importance are enumerated as follows:

- **Keywords in document**: A page is considered relevant if it contains some or all of the keywords in the query. Also, the frequency with which the keywords appear on the page may be considered.

- **Similarity to a query**: Often a user specifies an information need as a short query. In some cases a longer description of the need may be
available. Similarity between the short or long description and each crawled page may be used to judge the page’s relevance.

- **Similarity to seed pages:** The pages corresponding to the seed URLs are used to measure the relevance of each page that is crawled. The seed pages are combined together into a single document and the cosine similarity of this document and a crawled page is used as the page’s relevance score.

- **Classifier score:** A classifier may be trained to identify the pages that are relevant to the information need or task. The training is done using the seed pages as positive examples. The trained classifier will then provide Boolean or continuous relevance scores to each of the crawled pages.

- **Retrieval system rank:** N different crawlers are started from the same seeds and allowed to run until each crawler gathers P pages. All of the N.P pages collected from the crawlers are ranked against the initiating query or description using a retrieval system. The rank provided by the retrieval system for a page is used as its relevance score.

- **Link-based popularity:** A simpler method would be to use link-based to the crawled page to derive similar information. Many variations of link-based methods using topical weights are choices for measuring topical popularity of pages.

### 2.2.2 Web Crawling

Crawling is the process of collecting web pages. The crawling process usually starts from a set of source web pages. The web crawler follows the source page hyperlinks to find more web pages. Search engines use “crawling” along the web
creating hyperlinks. This process is repeated on each new set of pages and continues until no more new pages are discovered or until a predetermined number of pages have been collected. The crawler has to decide in which order to collect hyperlinked pages that have not yet been crawled. The crawlers of different search engines make different decisions, and so collect different sets of web documents. Web search engines handle hundreds of millions of queries each day. When a user submits a query, usually in the form of a list of textual terms, an internal scoring function is applied to each web page in the repository. Applying the proposed method to the page produces a numerical score, representing the best available estimate of the usefulness of the page to the user. Query results are usually presented in the form of a sequential list of links to pages arranged in descending order of score. A study examining the overlap among three major web search engine (Amanda et al 2006) results retrieved for the same queries. They used three major web search engines with different crawling and indexing technology - Ask Jeeves, Google and Yahoo! These results are then compared with results retrieved for the same queries by the meta-search engine Dogpile.com. The goal of their work was to measure the overlap across three major web search engines on the first results page across a wide range of search queries, to determine the differences in the first page of search results and their ranking across the three single-source web search engines, and to measure the degree to which the meta-search web engine provides the most highly-ranked search results from three major single source web search engines.

In what order a crawler should visit the URLs it has seen, in order to obtain more “important” pages first is studied by Junghoo Cho et al (1998). Obtaining important pages rapidly can be very useful when a crawler cannot visit the entire web in a reasonable amount of time. Several importance metrics, ordering schemes and performance evaluation measures are defined for this problem. The results show that a crawler with a good ordering scheme can obtain important pages significantly faster than one without. Not all pages are of equal interest to the crawler’s client. For instance, if the client is building a specialized database on a particular topic, then
pages that refer to that topic are more important, and should be visited as early as possible. Given a web page, the importance of the page can be defined in one of the following metrics: similarity to a driving query, back link count, pagerank, forward link count and location metric. Pages that have relevant content and many back links will be ranked high. The crawler is so designed that, if possible it visits high pages before lower ranked ones. This can be stated more precisely in three ways: crawl and stop – the crawler starts at its initial page and stops after visiting k pages; crawl and stop with threshold – the crawler visits k pages and stops when the percentage of the hot pages have been visited; limited buffer crawl – considers the impact of limited storage on the crawling process. A crawler keeps a queue of URLs it has seen during the crawl, and must select from this queue the next URL to visit. The ordering metric is used by the crawler for this selection, i.e., it selects the URL u which has the highest value among all URLs in the queue. The method was evaluated experimentally with several combinations of importance and ordering metrics, using the Stanford web pages.

An efficient crawling algorithm for retrieving the most important pages remains a challenging issue. Ali Mohammad et al (2009) proposed an intelligent crawling algorithm based on reinforcement learning, called FICA (Fast Intelligent Crawling Algorithm) that models a random surfing user. The priority for crawling pages is based on a concept called logarithmic distance. FICA outperforms in discovering highly important pages. Furthermore, it computes the importance (ranking) of each page during the crawling process. In this method, the aim is to minimize the sum of received distance by the crawler agent so that a page with a low distance will have the highest priority for crawling. The breadth-first crawling algorithm traverses the crawling tree by following its links i.e., level by level. The FICA algorithm is based on the breadth-first algorithm with a new definition of distance between web pages, called logarithmic distance. If page i points to page j then the weight of the link between i and j is equal to \( \log_{10} O(i) \) where \( O(i) \) denotes i’s
out-degree, number of outward links. The distance between pages i and j is the weight of the shortest path (the path with the minimum value) or sum of link weights in the shortest path from i to j, called logarithmic distance. Here, pages considered for crawling are those with lower logarithmic distances from the root pages (starting URLs). In other words, the priority of pages for crawling is dependent on their logarithmic distance from the root.

Google is designed to crawl and index the web efficiently and produce much more satisfying search results than existing systems. It makes especially heavy use of the additional structure present in hypertext to provide much higher quality search results. The search engine makes especially heavy use of the additional structure present in hypertext to provide much higher quality search results. Google has several other features. First, it has location information for all hits and so it makes extensive use of proximity in search. Second, Google keeps track of some visual presentation details such as font size of words. Words in a larger or bolder font are weighted higher than other words. Third, full raw HTML of pages is available in a repository. The text of links is treated in a special way in Google search engine. Most search engines associate the text of a link with the page that the link is on. In addition, Brin and Page (1998) associate it with the page the link points to. This has several advantages. First, anchors often provide more accurate description of web pages than the pages themselves. Second, anchors may exist for documents which cannot be indexed by a text-based search engine, such as images, programs and databases. This makes it possible to return web pages which have not actually been crawled.

As the collection of the pages grow, very high precision, the number of relevant documents returned, is required. The notion “relevant” is to only include the very best documents since there may be tens of thousands of slightly relevant documents. This very high precision is important even at the expense of recall, the total number of relevant documents the system is able to return. The Google search engine has two important features that help it produce high precision results. First, it
makes use of the link structure of the web to calculate a quality ranking for each web page which is called PageRank (Lawrence Page et al 1998). Second, Google utilizes link to improve search results. In particular, link structure and link text provide a lot of information for making relevance judgments and quality filtering. Google makes use of both link structure and anchor text. In designing Google, both the rate of growth of the web and the technological changes are considered. Google is designed to scale well to extremely large data sets. It makes efficient use of storage space to store the index. Its data structures are optimized for fast and efficient access. Further, the cost of the index and store text or HTML will eventually decline relative to the amount that will be available. The major operations of Google are crawling, indexing and sorting. This way the information can be kept up to date and major changes to the system can be tested relatively quickly.

2.2.2.3 Ranking the Documents

When a user sends a query to a search engine, the search engine returns the URLs of documents matching all or one of the terms, depending on both the query operator and the algorithm used by the search engine. Ranking is the process of ordering the returned documents in decreasing order of relevance, so that the best answers are on the top. Ranking that uses hyperlink analysis is called connectivity-based ranking (Monika 2001). These ranking techniques usually assume the most straightforward representation: each web page in the collection is modeled by a node in the graph. If page A contains a hyperlink to page B, then there exists a directed edge (A, B) in the graph. If A does not have a hyperlink to B, there is no directed edge. This directed graph is called the link graph. Connectivity-based ranking can be partitioned into two classes:

- Query-independent ranking, which assign a fixed score to each page in the collection.
- Query-dependent ranking or topic-sensitive ranking, which assign a score to each page in the collection in the context of a given query.

2.2.2.3.1 Query-independent Ranking

Query-independent ranking schemes assign a score to a document once and then use this score for all subsequent queries. This aims to measure the intrinsic quality of a page. A score is assigned to each page independent of a specific user query. At query time, this score is used with or without some query-dependent ranking criteria to rank all documents matching the query. Brin and Page (1998) computed the Pagerank of a page by weighting each hyperlink to the page proportionally to the quality of the page containing the hyperlink. Here, they present Google, a prototype of a large-scale search engine which makes heavy use of the structure present in hypertext. Despite the importance of large-scale search engines on the web, very little academic research has been done on them. It addresses the question of how to build a practical large-scale system which can exploit the additional information present in hypertext. Academic citation literature is applied to the web, largely by counting citations or backlinks to a given page. This gives some approximation of a page’s importance or quality. PageRank extends this idea by not counting links from all pages equally, and by normalizing by the number of links on a page. It can be calculated using a simple iterative algorithm, and corresponds to the principal eigenvector of the normalized link matrix of the web. To determine the quality of a referring page, the Pagerank was used recursively, with an arbitrary initial setting of the pagerank values. The Pagerank measure effectively distinguishes high-quality web pages from low-quality web page, and is currently an important part of the ranking function used by the Google search engine.

Although PageRank has its advantages, it also has certain limitations as a model for representing page importance.
The link graph, which PageRank relies on, is not a very reliable data source, because hyperlinks on the web can be easily added or deleted by web content creators.

PageRank only models a random walk on the link graph, but does not take into consideration the lengths of time which the web surfer spends on the web pages during the random walk.

To overcome these drawbacks, the BrowseRank algorithm (Yuting Liu et al. 2008) computes the page importance by using a user browsing graph created from user behavior data. In this graph, vertices represent pages and directed edges represent transitions between pages in the user's web browsing history. Furthermore, the lengths of staying time spent on the pages by users are also included. Many algorithms have been developed in order to further improve the accuracies and efficiencies of PageRank. Some work focuses on speed-up of the computation (Haveliwala 1999), while others focus on refinement and enrichment of the model. For example query-dependent PageRank proposed by Richardson and Domingos (2002). The propagation in TrustRank starts from the reliable pages; TrustRank can be more spam-resistant than PageRank.

New versions of PageRank are introduced using alternative document models by Mike and Liwen (2004). They introduce several new versions of PageRank (the link based web page ranking algorithm), based upon an information science perspective on the concept of the web document. Although the web page is the typical indivisible unit of information in search engine results and most web information retrieval algorithms, other research has suggested that aggregating pages based upon directories and domains gives promising alternatives, particularly when web links are the object of study. The algorithms based upon these alternatives were used to rank four sets of web pages. The ranking results were compared with human subjects rankings. The approach worked well for the set that includes pages from different web
sites; however, it does not work well in ranking pages that are from the same site. The effectiveness of this algorithm is compared against the standard PageRank. Human ranking judgment is used as the benchmark against which to compare different algorithms. The PageRank can be improved by incorporating rankings of a page based upon its hosting site, domain and directory.

2.2.2.3.2 Query-dependent Ranking

Query-dependent ranking schemes require a hyperlink analysis for each query but tailor their hyperlink analysis specifically to the query. Here, the algorithm assigns a score that measures the quality and relevance of a selected set of pages for the given user query. It builds a query-specific graph, called a neighborhood graph, and performs hyperlink analysis on it. This graph contains only pages on the query topic. Jeromy and Rick (1997) build this graph by starting with a set of pages containing the query terms; this set can be the list of results given by a full-text search engine. This root set is augmented by its neighborhood that comprises all or a large sample of the pages directly pointing to, or directly pointed by the pages in the root set.

The idea of limiting the number of pages added to the neighborhood set by following back links was introduced by Bharat and Henzinger (1998). The method addresses the problem of topic distillation on the web, namely, given a typical user query to find quality documents related to the query topic. The process of finding quality documents on a query topic is called topic distillation. Connectivity analysis has been shown to be useful in identifying high quality pages within a topic specific graph of hyperlinked documents. The essence of the approach is to augment a previous connectivity analysis based algorithm with content analysis. Three problems are discovered with connectivity analysis as suggested by Kleinberg (1998), i.e., a link-only approach: mutually reinforcing relationships between hosts, automatically generated links and non-relevant documents. Mutually reinforcing relationships
between hosts give undue weight to the opinion of a single person. Ideally, it is
decided that all the documents on a single host to have the same influence on the
document they are connected to. To achieve this fractional weights are assigned to
edges. The authors combine content analysis using traditional information retrieval
techniques with improved connectivity analysis to tackle topic drift. There are two
basic approaches to determine the relevance of a node to the query topic: (i)
eliminating non-relevant nodes from the graph, and (ii) regulating the influence of a
node based on its relevance. The combinations of these techniques solve the
remaining two problems of HITS method.

The WebQuery (Carriere and Kazman 1997) system offers a powerful
method for searching the web based on connectivity and content. This is examined by
links among the nodes returned in a keyword-based query. Then the nodes are ranked,
giving the highest rank to the most highly connected nodes. They explore several
techniques for visualizing the returned information – including cone trees, 2D graphs,
3D graphs, lists etc. They propose the following approach for building a
neighborhood graph:

- A start set of documents matching the query is fetched from a search
  engine.
- The start set is augmented by its neighborhood, which is the set of
documents that either hyperlinks to or is hyperlinked to by documents
in the start set. Since the indegree of nodes can be very large, in
practice a limited number of these documents are included.
- Each document in both the start set and the neighborhood is modeled
by a node. There exists an edge from node A to node B if and only if
document A hyperlinks to document B. Hyperlinks between pages on
the same web host can be omitted.
Various ranking schemes are now be used on the neighborhood graph. As with query-independent ranking schemes, an indegree-based approach ranks the nodes in the neighborhood graph by the number of documents hyper linking to them. Neighborhood graphs typically consist of thousands of nodes. Computing the pagerank on this graph produces a ranking similar to that produced by indegree-based ranking.

The network structure of a hyperlinked environment can be a rich source of information about the content of the environment, provided the effective ways of understanding it. A set of algorithmic tools (Jon Kleinberg 1998) is developed for extracting information from the link structures of such environments. The central issue is the distillation of broad search topics, through the discovery of “authoritative” information sources on such topics. An algorithm formulated on the notion of authority, based on the relationship between a set of relevant authoritative pages and the set of hub pages that join them together in the link structure. The formulation has connections to the eigenvectors of certain matrices associated with the link graph; these connections in turn motivate additional heuristics for link-based analysis. A more complex idea is the HITS algorithm is based on considering that relevant pages can be either “authority pages” or “hub pages”. An authority page is expected to have relevant content for a subject, and a hub page is expected to have many links to authority pages. The HITS algorithm produces two scores for each page, called “authority score” and “hub score”. These two scores have a mutually-reinforcing relationship: a page with high authority score is pointed to by many pages with a high hub score, and a page with a high hub score points to many pages with a high authority score. In particular, the algorithm focuses on the use of links for analyzing the collection of pages relevant to a broad search topic, and for discovering the most authoritative pages on such topics. The work originates in the problem of searching on the WWW, which could be defined roughly as the process of discovering pages that are relevant to a given query. The quality of a search method necessarily requires human evaluation, due to the subjectivity inherent in notions such as relevance. Here,
a link-based model is proposed for the conferral of authority, and show that it leads to a method that consistently identifies relevant, authoritative web pages for broad search topics. They observe that a certain natural type of equilibrium exists between hubs and authorities in the graph defined by the link structure, and this exploit to develop the algorithm that identifies both types of pages simultaneously. The algorithm operates on focused sub graphs of the web that are constructed from the output of a text-based web search engine, produces small collections of pages likely to contain the most authoritative pages for a given topic.

In the original pagerank algorithm for improving the ranking of search query results, a single pagerank vector is computed, using the link structure of the web, to capture the relative importance of web pages, independent of any particular search query. To yield more accurate search results, Taher Haveliwala (2002) proposed computing a set of pagerank vectors, biased using a set of representative topics, to capture more accurately the notion of importance with respect to a particular topic. For ordinary keyword search queries, they compute the topic-sensitive PageRank scores for pages satisfying the query using the topic of the query keywords. For searches done in context, the topic-sensitive pagerank scores is computed using the topic of the context in which the query appeared. In this model, a set of pagerank vectors is computed each biased with a different topic to create for each page a set of importance scores with respect to particular topics. At query time, these importance scores are combined based on the topics of the query to form a composite pagerank score for those pages matching the query. This biasing process involves introducing artificial links into the web graph during the online rank computation. To generate a set of biased pagerank vectors using a set of basis topics, the URLs present in the various categories in the ODP is used.

An improvement to HITS is probabilistic HITS, a model that has clear statistical representations (Cohn and Chang 2000). They describe a model of document citation that learns to identify hubs and authorities in a set of linked
documents, such as pages retrieved from the web, or papers retrieved from a research paper archive. Unlike the popular HITS algorithm, which relies on dubious statistical assumptions, this model provides probabilistic estimates that have clear semantics. An advantage of the probabilistic factored model is that it provides a foundation for building a unified probabilistic model of the content and connections of linked documents.

A Co-Recommendation algorithm (Mingjun Lan et al 2002), consists of the features of the recommendation rule and the co-citation algorithm. The algorithm addresses some challenges that are essential for further searching and recommendation algorithms. It does not require users to provide a lot of interactive communication. Further, it supports other queries, such as keyword, URL and document investigations. The high online performance can be obtained as well as the repository computation, which can achieve a high group forming accuracy using only a fraction of web pages from a cluster. The purpose of the method is to propose an algorithm called co-recommendation, which involves in indexer and ranking modules. The algorithm is based on the index term weighting. The term weights are ultimately used to compute the degree of similarity between stored web pages in the system and the user query. These index terms are noun groups from the web pages, because most of the semantics comprise nouns in a sentence using natural language text. The index term weighting is calculated using a TF-IDF scheme. Then, a correlation-based similarity computation is conducted to assess the similarity between two web pages. The cocitation+ algorithm, which differentiates between pages that are just a collection of links and pages that have more content than links, incorporates the information into the web ranking module.

One of the most important drawbacks is that not all the documents in the neighborhood set are about the original topic (“topic drifting”), as while expanding the root set; it is common to include popular pages that are highly-linked, but unrelated to the query topic. The solution is to use analysis of the contents of the
documents, and pruning the neighborhood graph by removing the documents that are too different from the query. This is done by using a threshold for the standard TF-IDF measure of similarity (Gerard and Christopher 1988) between documents and queries. A different variation of the HITS algorithm, designed specifically to avoid “topic drifting”, was presented by Soumen Chakrabarti et al (1998). They describe the design, prototyping and evaluation of a system for automatically compiling a list of authoritative web resources on any topic. The system is built on an algorithm that performs a local analysis of both text and links to arrive at a global consensus of the best resources for the topic. In the approach, for each link, the text near it in the origin page and the full text of the destination page are compared. If they are similar, the link is given a high weight, as it carries information about semantic similarity between the origin and destination pages. As this heuristic keeps the pages in the neighborhood set more closely related, a more relaxed expansion phase can be done. The authors propose to follow two levels of links forward and backward from the root set, instead of just one link.

The original PageRank algorithm for improving the ranking of search query results computes a single vector, using the link structure of the web to capture the relative importance of web pages independent of any particular search query. To yield more accurate search results, Taher (2003) proposes computing a set of pagerank vectors, based using a set of representative topics to capture more accurately the notion of importance with respect to a particular topic. For ordinary keyword search queries, the topic-sensitive pagerank scores are computed for pages satisfying the query using the topic of the query keywords. For searches done in context, e.g., when the search query is performed by highlighting words in a web page, the score is computed using the topic of the context in which the query appeared. By using linear combinations of these precomputed biased pagerank vectors to generate context-specific importance scores for pages at query time, more accurate rankings can be generated than with a single, generic page rank vector. The method computes multiple importance scores for each page; computes a set of scores of the importance of a page
with respect to various topics. At query time, these importance scores are combined based on the topics of the query to form a composite pagerank score for those pages matching the query. The first step is to generate a set of biased pagerank vectors using a set of basis topics. This step is performed once, offline, during the preprocessing of the web crawl. There are many possible sources for the basic set of topics. Here, the freely available hand constructed ODP is used as a source of topics. The second step is performed at query time. If the query was issued by highlighting the term in some web page, then the context is in the web page. Using a text index, the URLs are retrieved for all documents containing the original query terms. Finally, the query-sensitive importance score is computed for each of the retrieved URLs. The results are ranked according to the score.

A strict personalized Pagerank coverage guarantee on the downloaded pages during and after the crawl is specified in RankMass crawler (Junghoo Cho and Uri Schonfeld 2007). Here, given a set of trusted pages, the algorithm developed (1) provides a theoretical guarantee on how much of the important part of the web it will download after crawling a certain number of pages and (2) give a high priority to important pages during a crawl, so that the search engine can index the most important part of the web first. The two issues related to these points are mentioned as ‘Crawler coverage guarantee’ and ‘Crawler efficiency’. This makes search engine operators judge exactly how much of the web they have covered so far, and thus makes an informed decision on when it is safe to stop crawling the web.

### 2.2.2.4 Text-based Ranking

Although the physical characteristics of web information are distributed and decentralized, the web can be viewed as one big virtual text document collection. In this regard, the fundamental questions and approaches of traditional IR research are likely to be relevant in web IR (Yang 2002). The web is rich with various sources of information that go beyond the contents of documents, such as document
characteristics, hyperlinks, web directories (e.g., Yahoo) and user statistics. In topic relevance task, they examined the effects of combining result sets as well as those of various evidence source parameters for text and link-based methods. Analysis of results suggests that index source, query length and host definition are the most influential system parameters for retrieval performance.

In a document retrieval environment, where stored entities (documents) are compared with each other or with incoming patterns (search requests), it appears that the best indexing space is one where each entity lies as far away from the others as possible. In these circumstances the value of an indexing system may be expressible as a function of the density of the object space. An approach based on space density computations is used to choose an optimum indexing vocabulary for a collection of documents. The language independent vector space model representation of documents has proved effective for text classification (Salton et al 1975 and Gerard 1971). This vector space model is the standard technique for ranking documents according to a query. This model is described with indexing terms that are considered to be coordinates in a multidimensional space where documents and queries are represented as binary vectors of terms resulting in a term-document two-dimensional \( m \times n \) matrix, where \( m \) is the number of terms and \( n \) is the number of documents in the collection. The similarity of a query to a document is specified by a formula that transforms each vector using certain weights and then calculates the cosine of the angle between the two weighted vectors. For text-based information retrieval systems, documents are viewed to the users in decreasing order using this similarity measure. A weighting scheme uses statistical properties from the text and the query to give certain words more importance when doing the similarity calculation. The most used scheme is the TF-IDF weighting scheme (Gerard and Christopher 1988), that uses the frequency of the terms in both queries and documents to compute the similarity.

Most web pages are linked to others with related content. This idea, says that text in, and possibly around, HTML anchors describe the pages to which they
point, is the foundation for a usable web. Davidson (2000) finds that the likelihood of linked pages having similar textual content to be high; the similarity of sibling pages increases when the links from the parent are close together; titles, descriptions and anchor text represent at least part of the target page; and that anchor text may be a useful discriminator among unseen child pages. The method uses the TF-IDF vector space model to calculate the comparability among the web pages in an internet subgraph. He proposed the concept ‘topical locality’ of web pages to mean that two pages linked through hyperlinks have higher comparability than any two random web pages. The primary experiments performed include measuring the textual similarity:

- of the title to its page, and of the description to its page
- of a page and one of its children
- of a page and a random page
- of two pages with the same direct ancestor and with respect to the distance in the parent document between referring URLs
- of anchor text and the page to which it points
- of anchor text and a random page
- of anchor text and a page different from the one to which it points, but still linked from the parent page

Additionally, they measured lengths of titles, descriptions, anchor texts and page textual contents. This discovery pointed out a direction for focused crawling. Many research activities are conducted by this discovery to predict the relatedness between web pages and some specific topics. That is, if a page is relevant to a specific topic, the pages hyperlinked by it would be more likely to be related to this topic. So, it is more probable to find other topic-specific pages with the help of links from webs relevant to the specific topic.

TF represents term frequency that specifies if a term appears several times in a document it is better for describing the contents of that document. The TF is
usually normalized with respect to document length, that is, the frequency of term divided by the frequency of the most frequent term in the document. IDF represents inverse document frequency and reflects how frequent a term is in the whole collection. The rationale is that a term that appears in a few documents gives more information than a term that appears in many documents (Carlos 2004). This weighting scheme is one of the methods used by several web crawling algorithms and gives good results.

The web pages are indexed based on text content and link analysis. Numerous algorithms are based on these two categories. Almpanidis et al (2007) have proposed a latent semantic indexing classifier which combines link analysis with text content for the purpose of retrieving and indexing the domain-specific web documents. A different approach has been presented in the implementation to focused crawling and this is intended to overcome the restrictions of the initial training data, while maintaining a high standard of recall / precision ratio and efficiency. The main goal of this work is to provide an efficient topical information resource discovery algorithm when no previous knowledge of link structure is available except that found in web pages already fetched during a crawling phase. The breadth-first policy uses a simple FIFO queue for the unvisited documents and provides a fairly good bias towards high quality pages without the computational cost of keeping the queue ordered. The focused or topic-driven crawler is assigned the task of automatically classifying crawled pages to existing category structures and simultaneously discovering web information related to the specified domain while avoiding irrelevant regions of the web.

A popular approach for focused resource discovery is the best-first search algorithm where two URL queues are maintained; one containing the already visited links and another having the yet unvisited references of the first queue, also called crawl frontier (Arasu et al 2001). The challenging task is periodically reordering the links in the crawl frontier efficiently. The importance metrics can be either interest
driven where the classifier for document similarity checks the text content and popularity / location driven where the importance of a page depends on the hyperlink structure of the crawled document. They cover crawling, local web page storage, indexing and the use of link analysis for boosting search performance. The indexer module builds two basic indexes: a text (or content) index and a structure (or link index).

Lixin Han et al (2006) proposed a Hybrid Web Search (HWS) architecture, which combines text search with semantic search to enhance the precision and recall. With the emergence of the semantic web, the information retrieval can be improved if machines could understand the content of web pages. The existing information retrieval technologies can be classified mainly into three classes:

- The traditional information retrieval technologies (Mei and Koichi 2000) are almost based merely on the occurrence of words in documents. It is only limited to string matching.
- Search engines are limited to string matching and link analysis. The most widely used algorithms are PageRank and HITS algorithm.
- The widespread availability of machine understandable information on the semantic web offers some opportunities to improve traditional search.

The major difficulty in implementing is the extra time spent to annotate web pages. It is possible that not all of the relevant web pages are annotated and therefore only part of them can be returned. As a result, there is no sufficient page volume. Moreover, in contrast to a typical web search interface for keyword entering, the semantic search systems are more complicated. There has to be a way to choose detailed information such as ontology, category etc.
2.2.3 Web Characterization

The web crawling has been studied in the context of search engines, web characterization, web archiving and the deep web. The method (Balamurugan et al 2008) designs a new model and architecture for a web crawler that tightly integrates the crawler with the rest of the search engine, providing access to the metadata and links of the documents that can be used to guide the crawling process effectively. The number of pages on the web can be considered infinite, and given that a web crawler cannot download all the pages, it is important to capture the most important ones as early as possible during the crawling process. They tend to model and study user browsing behavior in web sites, concluding that it is not necessary to go deeper than five levels from the home page to capture most of the pages actually visited by people, and support this conclusion with log analysis of several web sites. Fang Liu et al (2004) propose a novel technique to learn user profiles from users search histories. The user profiles are then used to improve the retrieval effectiveness in web search. A user profile and a general profile are learned from the users search history and a category hierarchy, respectively. Web search is conducted based on both the user query and the set of categories.

Classification of web pages into relevant categories helps the search engine to get relevant results. Web page classification involves the classification of web pages under some predefined categories that may be organized in a tree or other structures. Web clustering involves the grouping of web pages based on the similarities among them. Each resultant group should have similar web pages while web pages from different resultant groups should be dissimilar (Ee-Peng and Aixin 2005). The search result clusters are ranked according to salience phrase and thus the more likely clusters required by users are ranked higher (Hua-Jun et al 2004). They reformalize the clustering problem as a salient phrase ranking problem. Given a query and the ranked list of documents, typically a list of titles and snippets returned by a certain web search engine, the method first extracts and ranks salient phrases as...
candidate cluster names, based on a regression model learned from human labeled training data. The algorithm is to cluster search results into different groups, and to enable users to identify their required group at a glance. They first get the webpage of search results returned by a certain web search engine. These web pages are analyzed by an HTML parser and result items are extracted. Generally, there are only titles and query-dependent snippets available in each result item. These contents are assumed as informative enough because most search engines are well designed to facilitate users relevance judgment only by the title and snippet, thus it is able to present the most relevant contents for a given query. Each extracted phrase is in fact the name of a candidate cluster, which corresponds to a set of documents that contain the phrase. Meanwhile, several properties for each distinct phrase are calculated during the parsing. One of the properties is TF-IDF method.

There are two types of web page classification. They are subject-based and genre-based classifications. In subject-based classification (also called topic-based classification), web pages are classified based on their contents or subjects. This approach, which defines numerous topic categories, is used in the proposed methods.

Pandey and Olston (2005) discussed the prioritizing of web pages for selective re-downloading into a search engine repository. To improve the quality of the user experience for those who query the search engine, they initially performed a quantitative characterization in which the inconsistency between the content of the repository and the present content of the live web has influenced the quality of the user experience. This characterization produces a user centric metric of the quality of a search engine’s local repository. A policy has been derived by using this metric for prioritizing web page re-downloading which is driven by search engine usage. They gave importance to the problem of scheduling the refreshing of web pages previously present in the repository, and demonstrated the way for calculating the priorities efficiently. They compile responses to user queries by accessing a local repository that mirrors the web. When a user submits a query, usually in the form of a list of
textual terms, an internal scoring function is applied to each web page in the repository. Applying this function to a page produces a numerical score, representing the best available estimate of the usefulness of the page to the user who submitted the query. Query results are usually presented in the form of a sequential list of links to pages arranged in the descending order of score. When the user clicks on a link in the query result list, the web browser fetches the current copy of the linked page from the web.

Automatic document categorization plays a key role in the development of future interfaces for web based search. Clustering algorithms are considered as a technology that is capable of mastering this task. The method Sven Meyer and Benno (2002) presents results of a comprehensive analysis of clustering algorithms in connection with document categorization. The contributions relate to exemplar-based, hierarchical and density-based clustering algorithms. As analyzed by Simon Dennis et al (2002) there is a plethora of web search technology, which can broadly be classified into four categories:

- **Unassisted Keyword search**: One or more search terms are entered and the search engine returns a ranked list of document summaries.
- **Assisted Keyword search**: The search engine produces suggestions based on the user’s initial query.
- **Directory-based search**: The information space is divided into a hierarchy of categories, where the user navigates from broad to specific classes.
- **Query-by-Example**: The user selects an interesting document snippet, which is then used as the basis of a new query.

The web based search interface should model the search process within three phases: an initialization phase according to the plain unassisted keyword search paradigm, a categorization phase similar to the directory-based search paradigm and a refinement
phase that may combine aspects from assisted keyword search and query-by-example paradigm.

2.2.4 Indexing and Querying Web Pages

Web pages come in many different formats such as plain text, HTML pages, PDF documents, audio, image and/or video information associated with them. Not all of the web pages are of uniform size. The web pages are dynamic and volatile in nature. There is no unique format for the web page. Some web pages may be unstructured (full text), some pages may be semi-structured (HTML pages) and some pages may be structured (databases). To find the category of the web page, the textual information in the web page serves as a hint. As the web pages contain many irrelevant, infrequent and stop words that reduce the performance of the classifier, extracting or selecting representative features from the web page is an essential preprocessing step (Indra Devi et al 2008). The goal of the method is to find minimum number of highly qualitative features by integrating feature selection techniques. Selecting representative features from the web page reduces the time and resource requirement to find the category of the web page. The basic idea of feature selection algorithms is searching through all possible combinations of features in the data to find which subset of features works best for prediction. The selection is done by reducing the number of features of the feature vectors, keeping the most meaningful discriminating ones and removing the irrelevant or redundant ones. The web pages are indexed to extract a standard logical view from the documents. Each document is in the form of an unordered group of words. In modern web search engines, the document is viewed with word frequencies, text formatting attributes, meta-information about web pages and explicit keywords in the HTML markup. There are several text normalization operations (Ricardo 2004) which are executed for extracting the keywords from the document. The method explores the challenges to effectively use Natural Language Processing (NLP) for information retrieval. The
most common text mining processes used are: stop word removal, stemming and tokenization and are known as preprocessing steps of web crawling.

When parsing a web page to extract content information or in order to score new URLs suggested by the page, it is often helpful to remove commonly used words or stop words. In a web document, there are several, most common, short function words that have little discriminating power which are named as stop words. Hence it is necessary and beneficial to remove these words (Narayana et al 2009). This process of removing stop words from text is called stoplisting. Some of the common stop words are “it”, “can”, “an”, “and”, “by”, “for”, “from”, “of”, “the”, “to” etc. These words are removed from multiple word queries in order to improve the search performance. The removing of stop words is necessary because it reduces indexing file size and provides spin-off advantage for index compression.

The automatic removal of suffixes from words in English is of particular interest in the field of information retrieval. Stemming is the process which strips away the affixes to leave the stem of the word. It extracts the root of the word. Several stemming algorithms or stemmers have been proposed in order to decrease a word to its stem or root form. In information retrieval, stemming enables a matching of query and document terms which are related to a semantic similarity but which can appear in different morphological variants. It effectively works by treating complex suffixes as compounds made up of simple suffixes, and removing the simple suffixes in a number of steps (Porter 1980). In each step the removal of the suffix is made to depend upon the form of the remaining stem, which usually involves a measure of its syllable length.

In information retrieval environment, there are a collection of documents, each described by the words in the document title and possibly the words in the document abstract. Ignoring the issue of precisely where the words originate, a document is represented by a vector of words, or terms. Terms with a common stem
will usually have similar meanings. Bacchin et al (2002) propose and evaluate a statistical graph-based algorithm for stemming. Considering that a word is formed by a stem (prefix) and a derivation (suffix), the key idea is that strongly interlinked prefixes and suffixes form a community of sub-strings. Discovering these communities means searching for the best word splits which give the best word stems. The stems are used to describe the key terms of a query or document instead of the original word. The main use of stemming is indexing keywords for search. For example, stemming will ensure that the word “engineering”, “engineered” are reduced to “engineer”. Tokenization involves dividing the stream of text into words. This is performed by splitting the text into a sequence of tokens using spaces, punctuation and special characters, where each token is considered as a word. Subsequent processing produces document surrogates that can range from full text to just a list of keywords. The keywords that are taken by the above specified method are termed as tokens.

2.2.5 Related Works in Focused Crawling

At present, the total web in the internet is composed of several billion pages and it is impossible to collect all of relevant pages on time. Focused crawlers aim to search and retrieve only the subset of the World Wide Web that pertains to a specific topic of relevance without having to explore all web pages. The ideal focused crawler retrieves the maximal set of relevant pages while simultaneously traversing the minimal number of irrelevant documents on the web. The importance of a page for a crawler can also be expressed as a function of the similarity of a page to a given query. This hypertext resource discovery system is called “focused crawling” and was introduced by Chakrabarti et al (1999b). The goal of a focused crawler is to selectively seek out pages that are relevant to a pre-defined set of topics. The topics are specified not using keywords, but using exemplary documents. Rather than collecting and indexing all accessible web documents to be able to answer all possible ad-hoc queries, a focused crawler analyzes its crawl boundary to find the links that are
likely to be most relevant for the crawl, and avoids irrelevant regions of the web. This leads to significant savings in hardware and network resources and helps keep the crawl more up-to-date. To achieve such goal directed crawling, they designed hypertext mining programs that guide the crawler: a classifier that evaluates the relevance of a hypertext document with respect to the focus topics and a distiller that identifies hypertext modes that are great access points to many relevant pages within a few links. The experiment was conducted using several topics at different levels of specificity.

Focused crawling acquires relevant pages steadily while standard crawling quickly loses its way, even though they are started from the same root set. Focused crawling is robust against large perturbations in the starting set of URLs. It discovers largely overlapping set of resources in spite of these perturbations. It is also capable of exploring out and discovering valuable resources that are dozens of links away from the start set, while carefully pruning the millions of pages that may lie within the same radius. Focused crawlers are usually evaluated by ‘harvest rate’ which is the ratio between the number of relevant pages and all of the pages retrieved. ‘Loss rate’ is equal to 1 minus harvest rate. A page from which a link was extracted is called a parent page and the one to which the link points is a child page or a target page. Harvest rate can be improved by utilizing search engines as a source of seed URLs and back-references.

The focused crawler seeks, acquires, indexes and maintains pages on a specific set of topics that represent a relatively narrow segment of the web. To run a specific instance, initial human input has to be provided in two forms. The user has to select and / or refine specific topic nodes in the taxonomy (such as Yahoo!, ODP, etc.), and may also need to provide additional example URLs which serve as starting points for the crawl. The user inspects the system regularly. The system reports the most popular sites and resource lists, and the user can give feedback by marking them
as useful or not. This feedback goes back to the classifier and distiller. Two modes of focusing are possible with the classifier:

- **Hard focus rule:** While fetching a document d, then to find the leaf node c* with the highest probability. If some ancestor of c* has been marked good, the future visitation of URLs found on d are allowed, otherwise the crawl is pruned at d.

- **Soft focus rule:** The probability that a page is relevant to the focused crawl is sum of the probabilities of the good nodes for each document, because a good node is never the ancestor of another. In case of multiple paths leading to a page, the maximum of their relevance is considered.

The most crucial evaluation of focused crawling is to measure the rate at which relevant pages are acquired, and how effectively irrelevant pages are filtered off from the crawl. This harvest ratio is high for the focused crawler. Thus, web content can be managed by a distributed team of focused crawlers, each specializing in one or a few topics. Each focused crawler will be far more nimble in detecting changes to pages within its focus than a crawler that crawls the entire web.

The major problem in focused crawling is performing appropriate credit assignment to different documents along a crawl path, such that short term gains are not pursued at the expense of less obvious crawl paths that ultimately yield larger sets of valuable pages. Diligenti et al (2000) present a focused crawling algorithm that builds a model for the context within which topically relevant pages occur on the web. This context model can capture typical link hierarchies within which valuable pages occur, as well as model content on documents that frequently co-occur with relevant pages. The algorithm further leverages the existing capability of large search engines to provide partial reverse crawling capabilities. The algorithm shows significant
performance improvements in crawling efficiency over standard focused crawling. The specific implementation of such a context model is called a Context Graph. The crawler, Context Focused Crawler (CFC) uses the limited capability of search engine like AltaVista or Google to allow users to query for pages linking to a specified document. This data can be used to construct a representation of pages that occur within a certain link distance of the target documents. This representation is used to train a set of classifiers, which are optimized to detect and assign documents to different categories based on the expected link distance from the document to the target document. During the crawling stage the classifiers are used to predict how many steps away from a target document the current retrieved document is likely to be. This information is used to optimize the search. There are two distinct stages to use the algorithm when performing a focused crawl session: an initialization phase when a set of context graphs and associated classifiers are constructed for each of the seed documents and a crawling phase that uses the classifiers to guide the search and performs online updating of the context graphs. The implementation uses keyword indexing of each document using a modification of TF-IDF measure. The major limitation of this approach is the requirement for reverse links to exist at a known search engine for a reasonable fraction of the seed set documents. In practice, this does not appear to be a problem. However, even when no seed documents have yet been indexed by search engines, the approach can be bootstrapped. In this case a content model of the seed set is extracted and other high-confidence target data can be found using query modification on a search engine. The indexed target data pages returned by the search engine can then be used to build context graphs.

The performance of a focused crawling depends mostly on the richness of links in the specific topic being searched, and a focused crawling usually relies on a general web search engine for providing starting points. Focused crawlers are considered to tackle the scalability problem of topic-oriented or personalized search engines. A method using a decision tree on anchor texts of hyperlinks is discussed by Jun Li et al (2005). They propose a method to utilize anchor texts for determining the
priorities. Two assumptions about the web space are made in which the crawler is designed for crawling. First, the crawler is crawling in a limited URL domain, e.g., the web sites of a university or a company. Second, there exists an entry page to the URL domain, e.g., the home page of a university. So, the crawler is supposed to crawl in the limited URL domain starting from the entry page. Here, \( G = (V, E, r) \) denote the web graph of a limited URL domain, where \( V \) is the set of web pages, \( E \) is the set of hyperlinks between these web pages, and \( r \) is the entry page. To effectively exploit the information contained in anchor texts, they employ a decision tree to predict the relevance of the target pages. For a web graph \( G \), the crawler first crawl all the pages in \( V \) and identify the relevant pages by using a properly trained SVM (Support Vector Machine) classifier. The user needs to prepare some relevant and irrelevant example pages of topic in mind for the classifier. Second, for each page the shortest path from the entry page to the target page is computed by Dijkstra’s algorithm. Third, a function returning the anchor text associated with the hyperlink is computed with positive examples and negative examples of the decision tree learning. The crawler, Decision Tree Crawler (DTC) determines the priority of unvisited URLs. The efficiency of DTC is compared with two crawlers; a standard breadth-first crawler and a traditional focused crawler. The DTC crawler gives high priorities to all the children of relevant fetched pages.

Sotiris Batsakis et al (2009) have addressed issues related to the design and implementation of focused crawlers and also outperform classic focused crawlers in searching for specialized topics. Several variants of state-of-the-art crawlers relying on web page content and link information for estimating the relevance of web pages to a given topic are proposed. Particular emphasis was given to crawlers capable of learning not only the content of relevant pages but also paths leading to relevant pages. All crawlers achieve their maximum performance when a combination of web page content and (link) anchor text was used for assigning download priorities to web pages.
Several algorithms have been proposed based on topic-specific web crawlers. Some examples are: focused crawler (Diligenti et al 2000) shark-search (Hersovici et al 1998) and an intelligent crawler (Aggarwal et al 2001). All these works present algorithms that enable their crawlers to select web pages related to desired topics. The main purpose of those algorithms is to gather as many relevant web pages as possible. They assigned starting URLs defined by the user, which are relevant to an interested topic to the crawler. The architecture of a hypertext resource discovery system using a relational database is described by Chakrabarti et al (1999b). This system can answer questions that combine page contents, metadata and hyperlink structure in powerful ways. A key problem in populating the database in such a system is to discover web resources related to the topics involved in the queries. They exploit the properties that pages tend to cite pages with related topics, and given that a page u cites a page about a desired topic, it is very likely that u cites additional desirable pages. These properties are exploited by using a crawler controlled by two hypertext mining programs: a classifier that evaluates the relevance of a region of the web to the user’s interest and a distiller that evaluates a page as an access point for a large neighborhood of relevant pages. The implementation uses IBM’s Universal database, not only for robust data storage, but also for integrating the computations of the classifier and distiller into the database. This results in significant increase in I / O efficiency; a factor of ten for the classifier and a factor of three for the distiller. The goal is to go beyond representation and basic querying, to discover properties that combine the topical content of web pages and the linkage relationship between them.

The main contribution of the author is the design of a novel example-driven, goal-directed resource discovery system. The system uses two basic properties of the web: pages cite pages on related topics and a page that points to one page with a desired topic is more likely than a random page to point to other pages with desired topics. Another contribution is the implementation of the system on a relational database. The three modules crawler, classifier and distiller, together with monitoring and administering utilities, run concurrently as clients. The important distinction of
the system is the integration of topical content into the link graph model. The most important indicator of the success of the system is the harvest rate, or the average fraction of crawled pages that are relevant. Human judgment, although subjective and even erroneous, would be best for measuring relevance. The classifier is used to estimate the relevance of the crawl graph. For a close, self-contained data set, precise measurements of recall or coverage is made. They build a reference crawl by selecting a random set of start URLs from a set of sources, e.g., Yahoo!, Infoseek and Excite. The crawler ran starting from this set for some fixed time. Then, another random set of start sites was collected from AltaVista, making sure that both the start sets are disjoint. The crawler was monitored with the second start site. The results show that, within an hour of crawling, the test crawler collects up to 83% of the relevant URLs collected by the reference crawler.

Aggarwal et al (2001) proposes the novel concept of intelligent crawling which actually learns characteristics of the linkage structure of the web while performing the crawling. Specifically, the intelligent crawler uses the inlinking web page content, candidate URL structure, or other behaviors of the inlinking web pages or siblings in order to estimate the probability that a candidate is useful for a given crawl. This is a much more general framework than the focused crawling technique which is based on a pre-defined understanding of the topical structure of the web. The techniques discussed in the method are applicable for crawling web pages which satisfy arbitrary user-defined predicates such as topical queries, keyword queries or any combinations of the above. Unlike focused crawling, it is not necessary to provide representative topical examples, since the crawler can learn its way into the appropriate topic. This technique is referred as intelligent crawling because of its adaptive nature in adjusting to the web page linkage structure. The learning crawler is capable of reusing the knowledge gained in a given crawl in order to provide more efficient crawling for closely related predicates.
The crawler of this algorithm starts at a few general starting points and collects all web pages which are relevant to the user-specified predicate. Initially, the crawler behavior is as random as a general crawler, but it gradually starts autofocus as it encounters documents which satisfy the predicate. The crawler keeps track of the nodes which it has already visited, as well as a potential list of candidates. A web page is said to be a candidate when it has not yet been crawled, but some web page which links to it has already been crawled. For each such candidate, considerable amount of information is available and it is based on the web pages which link to it, their content, and the exact tokens in the URL address of the candidate itself. This information may be used by the crawler to decide the order in which it visits the web pages. In addition, an aggregate level of self learning information is collected during the crawl which models the relationship between the features in the candidates to the actual predicate satisfaction probability. Thus, at each point the crawler maintains candidate nodes which it is likely to crawl and keeps calculating the priorities of the nodes using the information about which nodes are most likely to satisfy the predicate. The pages are crawled in this order, and the appropriate statistical information is collected based on the relationship between the features in the candidate URLs and the predicate satisfaction of the crawled web page. For a candidate page which is about to be crawled, the value of \( P(C) \) is equal to the probability that the web page will indeed satisfy the user-defined predicate if it is crawled. The value of \( P(C) \) can be estimated by the fraction of web pages already crawled which satisfy the user defined predicate. Several factors are used during the crawl in order to evaluate its effectiveness, including the content of the inlinking web pages into the candidate URL, the set of tokens in the string representing the URL, the link based learning and the sibling based learning. Based on these factors, a composite crawler is created which perform robustly across different predicates. The aggregate interest ratio is a weighted product of the interest ratios for each of the individual factors. Equivalently, the preferences are combined by summing the weighted logarithms of the individual factors. The weights are used to normalize the different factors. By increasing the weight of a given factor, the importance of the
corresponding priority can be increased. In particular implementation, the weights are chosen with equally balanced. This algorithm does not need any suitable starting URLs, and the crawler is able to find a direction to target pages by starting at non-related web pages. Though their work is interesting, it is believed that the crawler supplied with good starting URLs is capable of gathering more relevant web pages. Then, the efficiency of topic-specific web crawling is measured in terms of a proportion of the number of relevant web pages and the total number of downloaded web pages. If this proportion is high, it means that the crawler can collect more relevant web pages than irrelevant ones during the early period of the crawling attempt.

The shark-search algorithm (Hersovici et al 1998) is a refined version of one of the first dynamic web search algorithms, the fish-search. The algorithm has been embodied into a dynamic web site mapping that enables users to tailor web maps to their interests. Web search services typically use a previously built index that is actually stored on the search service server, which is static. The dynamic search actually fetches the data at the time the query is issued and guarantees valid results. The shark-search overcomes the limitations of fish-search, by better estimating the relevance of neighboring pages, even before they are accessed and analyzed. The Fish-search algorithm, while being attractive because of the simplicity of its paradigm, and its dynamic nature, presents some limitations. More generally, the key problem of the fish-search is the very low differentiation of the priority of pages in the list. When many documents have the same priority, and the crawler is restricted to a fairly short time, arbitrary pruning occurs – the crawler devotes its time to the documents at the head of the list. Documents which are further along the list whose scores may be identical to some further along may be more relevant to the query. In addition, cutting down the number of addressed children by using the width parameter is arbitrary, and may result in losing valuable information. Clearly, the main issue that needs to be addressed is a finer grained scoring capability.
The improved version of Fish-search is developed, known as Shark search algorithm. One immediate improvement is to use, instead of the binary (relevant / irrelevant) evaluation of document relevance, called similarity engine is used to evaluate the relevance of documents to a given query. Such an engine analyzes two documents dynamically and returns a “fuzzy” score, i.e., a score between 0 and 1 (0 for no similarity, 1 for perfect conceptual match) rather than a binary value. A straightforward method for building such an engine is to apply the vector space model (Salton and Gill 1983). For any pair query, document (q, d), the algorithm returns a similarity score $\text{sim}(q, d)$ between 0 and 1. A more significant improvement consists of refining the calculation of the potential score of the children not only by propagating ancestral relevance scores deeper down the hierarchy, but also by making use of the meta-information contained in the links to documents. They propose to use the hints that appear in the parent document, regarding a child. The anchor text of the link is the author’s way to hint as to the subject of the linked document. A surfer on the web, encountering a page with a large amount of links, will use the anchor text of the links in order to decide how to proceed. The algorithm proposes to use a measure for getting as many relevant documents in the shortest delays. The measure is defined as the sum of similarity scores of all documents forming the maps, referred as “sum of information” measure. Preliminary experiments show significant improvements over the original fish-search algorithm.

The key of focused crawling is to accurately predict the relevance of the unvisited web pages pointed by known URLs to a given topic. Four policies (Zhumin Chen et al 2007) are proposed to predict the relevance of unvisited web pages to a topic. Further the combinations of these policies are used to improve the shark-search, which is a classic focused crawling algorithm mainly based on the textual information of web pages. A large number of experiments were carried out to identify the optimized combination and verify that the improved shark-search is more effective than the original shark-search. However, the original shark-search evaluate the relevance of the candidate URLs only by taking into account a linear combination of
ancestral pages content, anchor text and anchor context. In addition, shark-search describes the topic just as one or several keywords. In this method, some limitations of original shark-search are overcome by deriving topics from a hierarchical index of concepts – the Open Directory Project (ODP) in order to improve the relevance computation. ODP is organized as a tree where topics are internal nodes and example web pages relevant to its parent node are leaf nodes. The algorithm formalize the process of relevance prediction and look for specific features including textual information, namely page content and anchor text, URL address and link type to predict the relevance more accurately. Textual information is the base of focused crawling used to predict the relevance. Original shark-search evaluates the relevance based on the text. In general, textual information includes two kinds: page content and anchor text. URLs of pages are not randomly created but associate semantic meanings with the page content. The tokens in a known URL are used to predict the relevance of the unvisited page pointed to the URL on the topic. The source and target are the uncrawled URLs. The link relations are built from the source, relevant crawled URLs to the target relevant uncrawled URLs. The aggregate relevance score is determined by summing the weighted individual relevance score. The weights are used to normalize the different relevance scores. By increasing the weight, the importance of the corresponding individual can be increased. There are two measures for the performance of a focused crawler: precision rate (harvest rate) and sum of information. Precision rate measures the query result at page level. To simplify the evaluation, a notion of “relevant page” – a page is a relevant page if its relevancy is greater than a certain threshold. So, precision rate is the number of relevant pages retrieved to the number of all pages retrieved. The sum of information evaluates the result regarding all collected pages as a whole. To analyze the data further, they study the dynamic performance during different stages of the crawling process. The total of 5,000 pages is divided into 5 equal portions, each containing 1,000 consecutive pages according to the original visiting order of each crawler. Within each portion, the number of relevant pages and the sum of information of the search results are calculated and the performance was predicted.
Zhumin Chen, Jun Ma et al (2007) used link analysis technology to improve the shark-search. Here, they classify the URL type into five groups namely: downward, sibling, crosswise, outward and upward. Then, five heuristic rules are presented to infer the relation of an unvisited page to its parent page based on link types. According to these rules, different hyperlink relevance is assigned to the candidate URL.

Focused crawling is designed to traverse a subset of the web to gather documents on a specific topic (Cheng et al 2008). It aims to identify the promising links that lead to target documents, and avoid off-topic searches. The focused crawler of a special-purpose search engine aims to selectively seek out pages that are relevant to a pre-defined set of topics, rather than to exploit all regions of the web (Pal et al 2009 and Zhang et al 2007) and offer a potential solution to the problem. Debashis and Amritesh (2010) calculate the link score based on average relevancy score of parent pages and division score. The division score means how many topic keywords belong to a division in which a particular link belongs. The topic-specific search engine is constructed and optimized in accordance with domain knowledge. It can provide the information with higher precision than a general or directory search engine does. Focused (topical) crawlers are a group of distributed crawlers that specialize in certain specific topics. Each crawler will analyze its topical boundary when fetching web pages. To check the similarity of web pages with respect to topic keywords, a similarity function is used and the priorities of extracted out links are calculated based on meta data and resultant pages generated from focused crawler (Pal et al 2009). The work also uses a method for traversing the irrelevant pages that met during crawling to improve the coverage of a specific topic.

Menczer et al (2001) propose three different methods to evaluate crawling strategies. The three methods differ in how they assess the value of crawled pages: the first builds text classifiers, the second uses an independent retrieval system, and the
third applies a simple similarity metric to the dynamic set of crawled pages. They apply the proposed metrics to compare three types of crawlers. The first is best-first search, which prioritizes links in the frontier based on the similarity between the query and the page where the link was found. The second crawler is based on pagerank, the well known link-based algorithm used for ranking by the Google search engine. The third crawler, InfoSpiders, is based on search agents that evaluate links via neural nets. They use topics instead of queries, each represented by a collection of seed URLs.

Filippo Menczer et al (2003) presented the way to find the starting URLs using a web directory instead of a search engine. This way gives URLs categorized by a group of specialists. However, there is a disadvantage when the user’s topic of interest is not in any category of the web directory. In this case, using the search engine seems to be a useful way. A specialized search engine NECI (NEC Research Institute) via the domain specific modifying of queries and query processing is given by Steve Lawrence and Gilles (1998). The search engine improves the efficiency and precision of web search by downloading and analyzing each document and then displaying results that show the query terms in context. This helps users more readily determine if the document is relevant without having to download each page. Results are returned progressively after each page is downloaded and analyzed, rather than after all pages are downloaded. Pages are downloaded in parallel and the first result is typically displayed in less time than a standard search engine takes to display its response. Search engines do not index sites equally, not index new pages for months and no engine indexes more than about 16% of the web (Steve and Giles 2000). As the web becomes a major communications medium, the data on it must be made more accessible.

There are many avenues for the improvement of the web, in the areas of locating and organizing information. Several techniques for access to both general and scientific information on the web provide more improvement – search engines do not
provide comprehensive indices of the web and have difficulty in accurately ranking the relevance of results. Lawrence and Giles (1999) discuss the effectiveness of web search engines, including results that show that the major web search engines cover only a fraction of the publicly indexable web. Current research into improved searching of the web is discussed, including new techniques for ranking the relevance of results, and new techniques in metasearch that can improve the efficiency and effectiveness of web search. The creation of digital libraries incorporating autonomous citation indexing is discussed for improved access to scientific information on the web.

Topical crawlers (Filippo Menczer et al 2003) are increasingly seen as a way to address the scalability limitations of universal search engines, by distributing the crawling process across users, queries or even client computers. The context available to such crawlers can guide the navigation of links with the goal of efficient location of highly relevant target pages. The article is to examine the algorithmic aspects of topical crawlers. They have designed and implemented two new classes of crawling algorithms. These proposed crawler classes allow focusing on two crucial machine learning issues in the domain of web crawling strategies: the role of exploration versus exploitation and the role of adaptation versus static approaches. Seed pages are chosen so as to guarantee that from each seed there leads at least one path to some target page, within a distance of atmost links. The seed pages are identified by first considering the in links of each target page. The target pages represent a known subset of relevant pages. If the target set for any given topic is identified, the crawlers would reach these target pages, and / or pages similar to the targets, from whatever seed pages are used to begin the crawl. Each tested crawler visit up to maximum pages specified per topic, starting from a seed set.

One could theoretically generate topics and target sets using frequent queries from search engines and user assessments. The researchers used Yahoo! in previous due to its popularity (Menczer et al 2001) and in the current work Open
Directory Project (ODP) is used because it has less of a commercial bias, it is an open resource and it is maintained by a very large and diverse number of volunteer editors. They collected topics by running randomized breadth-first crawls starting from each of the main categories on the open directory site. A topic is represented by three types of information derived from the corresponding leaf page. First, the words in the ODP hierarchy form the topic’s keywords. Second, the external links form the topic’s targets. Third, the text descriptions and anchor text of the target URLs are concatenated to form a topic’s description. The difference between a topic’s keywords and its description is that the former are models of (short) query-like topics; and the latter, which is a much more detailed representation of the topic. The experiments described in this article use 50 such topics. This article employs only two of the performance metrics, focusing on the effect of different machine learning techniques on the performance of crawling algorithms over time. The first metric is the simple recall level based on the target pages. This measure allows us to determine how well a crawler can locate a few highly relevant pages. The second metric measure the mean similarity between the topic description and the set of pages crawled. Here, the TF-IDF weight, the inverse document frequency is computed from the crawl set. The crawls considered are very short, visiting 1,000 pages and longer crawls of 50,000 pages.

Pant et al (2005) presents a general framework to evaluate topical crawlers. Topical crawlers, also known as topic driven or focused crawlers, are an important class of crawler programs that complement search engines. They identify a class of tasks that model crawling applications of different nature and difficulty. Then they introduce a set of performance measures for fair comparative evaluations of crawlers along several dimensions including generalized notions of precision, recall and efficiency that are appropriate and practical for the web. The framework also specifies a procedure for defining crawling tasks of variable difficulty by selecting seed pages at appropriate distances from targets. The script selects topics from the Open Directory based on a number of parametric specifications, and generates files
containing topic keywords, descriptions, target URLs at various depths and seed URLs.

An algorithm that covers the detail of both the first and next crawling are given by Niran and Arnon (2005). For efficient result of the next crawling, they keep the log of previous crawling to build some knowledge bases such as: seed URLs, topic keywords and URL prediction. These knowledge bases are used to build the experience of the topic-specific web crawler to produce the result of the next crawling in a more efficient way. For seed URLs, the crawler needs some good URLs which point to many relevant web pages. It supports the crawler to collect as many relevant web pages as possible. For topic keywords, the crawler needs some appropriate keywords of topics of interest. It supports the crawler to recognize the keywords matching the topic. To build the knowledge base of topic keywords, the method extract some texts within the <Title></Title> tag and the <A></A> tag from each relevant page. Keywords extracted from <Title></Title> tag frequently occur in the relevant pages while keywords extracted from <A></A> tag frequently occur in the pages which point to relevant pages. For URL prediction, the crawler needs to predict the undownloaded pages. It supports the crawler to predict the relevancy of the pages of unvisited URLs. To predict the content of a URL, the page similarity is computed between its content of the previous crawled pages and the interested topic. The problem concentrated here is whether the next crawling process can be done in a more efficient way, or how they keep track in the change of web pages. The algorithm adds a learning ability from previous crawling to improve the efficiency of consecutive crawling processes. Although, the previous crawling may not be efficient, the next crawling may be more efficient when the crawler uses the knowledge bases as its experience. The algorithm has been separated into two parts: crawling with no knowledge base and with knowledge bases. During the consecutive crawling, the learning tendency of the crawler is examined by its learning curves: good crawling curve and bad crawling curve. Thus, the positive and negative learning are defined as
the type of learning ability. Although the efficiency of the crawler is discussed, the method does not show any result of the experiment.

Motivated by fish-search, De Bra and Post (1994) used topic similarity of a vector space model technique as a parameter in URL ordering process. This similarity is derived from a comparison between topic keywords and the content of web pages. The similarity score is composed of two parts: content similarity and anchor text similarity. Content similarity informs which web page has its content related to a topic, while anchor text similarity indicates which URLs found in a web page are related to a topic. Two types of search tools have been developed for the WWW: Gateways, offering limited search operations on small or large parts of the web, using a pre-compiled database and a client-based search tool that does automated navigation, thereby working more or less like a browsing user, but much faster and following an optimized strategy. This method highlights the properties, implementation and possible future developments of a client-based search tool called the fish-search, and compares it to other approaches. The fish search, implemented on top of Mosaic for X, offers an open-ended selection of search criteria. It allows the search to start from the current document or from the documents in the user’s hotlist. It turns the browser into an information retrieval tool for the web, based on searching the contents (body) of the documents in the web. As such, it complements the existing databases that provide indexes based on titles or headers of documents.

A number of factors (De Bra and Post 1994) determine how effective and efficient the search tool can be:

- The first is finding a starting node, from which relevant nodes for the search can be found by following only a few links.
- The algorithm must try to find nodes that are well spread over the WWW. The navigation algorithm that resembles breadth-first search performs well in this respect.
• The algorithm must try to avoid downloading irrelevant nodes. By favoring links to neighbors of relevant nodes, and also favoring links to nodes on different sites, the chance of finding more relevant nodes as well as penetrating into different parts of web increases.

• Retrieving nodes should be fast. Since most users of a web browser are likely to start searching from the same nodes many times before finding a cluster of relevant nodes. Using a cache reduces the time used on these popular nodes.

A general-purpose search engine, such as Google or AltaVista, usually generates thousands of hits, many of them irrelevant to the user query. Vertical search engines (Michael et al 2003) solve part of the problem by keeping indexes only in specific domains. Spiders are the software agents that search engines use to collect content for their databases. Most spiders use simple graph search algorithms, such as breadth-first search, to collect web pages. Good spidering algorithms can improve the precision of search results, however, by predicting whether a URL points to a relevant web page before downloading it to the local search engine database. Such predictions depend on representing web content and structure in ways that are meaningful to machines. The research in this area falls into one of two categories: content-based or link-based.

• **Content-based web analysis:** Spiders can apply indexing techniques for text analysis and keyword extraction to help determine whether a page’s content is relevant to a target domain. They can incorporate domain knowledge into their analysis to improve the results. For example, they can check the words on a web page against a list of domain-specific terminology and assign a higher weight to pages that contain words from the list. Assigning a higher weight to words and
phrases in the title or headings is also standard information retrieval practice that spiders can apply based on appropriate HTML tags.

- **Link-based web analysis:** Most of the research has used web link structure to infer important information about pages. Intuitively, the author of a web page A, who places a link to web page B, believes that B is relevant to A. The term in-links refers to the hyperlinks pointing to a page. Usually, the larger the number of in-links, the higher a spider will rate a page.

Anchor text is the word or phrases that hyperlink to a target page. Anchor text can provide a good source of information about a target page because it represents how people linking to the page actually describe it. It is also reasonable to give a link from an authoritative source, such as Yahoo, a higher weight than a link from a personal homepage. Vertical search engines offer more opportunity to apply domain knowledge in spider applications. The three spiders developed address different ways of combining content and link based web analyses and integrating them with graph search algorithms.

- The breadth-first search spider follows a basic graph-traversal algorithm. The BFS approach assumes that a URL relevant to a target domain is likely to have other relevant web pages in its neighborhood, and many commercial search engines have used this spidering technique. BFS has proved effective in discovering high-quality pages early in a spidering process (Najork and Weiner 2001).

- The PageRank spider performs a best-first graph search using PageRank as the heuristic. In each step, the spider gets the URL with the highest PageRank score, fetches the content, and extracts and enqueues all the pages outgoing links. It runs until it has collected a
predetermined number of pages. The spider calculates scores iteratively. Two priority queues were established namely, hot_queue and normal_queue, and ordered the URLs within each queue in descending order by pagerank score.

- Hopfield net spider, modeling the web as a neural network is a graph of many active nodes that are connected with each other by weighted links. It activates its nodes in parallel and combines activation values from different sources for each individual node until the node activation scores across the network converge to a stable state.

Starting with a set of seed URLs each represented as a node; the spider fetches and analyzes the seed web pages in iteration. After the spider calculates the weights of all nodes in the current iteration, it activates (visits) that set of nodes (URLs) and fetches them from the web in descending order of weight. To filter out low quality URLs, the spider bypasses nodes with a weight below a threshold value. After the spider has visited and downloaded all the pages with a weight greater than the threshold value, it updates the weight of each node in the new iteration to reflect the quality and relevance of the downloaded page content. The process iterates until the spider has collected a predetermined number of web pages or until the average weight of all nodes in iteration is smaller than a maximum allowable number. Using the notion of good page, the precision and recall rates of the spiders are determined.

Rungsawang et al (2004) proposed an algorithm which covers the discussion of both the initial and the successive crawling. For efficient result of the next crawling, they derive the information of previous crawling attempts to build some knowledge bases such as: starting URLs, topic keywords and URL prediction. The experience of the learnable topic-specific web crawler has been built using these knowledge bases in order to produce superior result for the next crawling. Their initial evaluation has demonstrated that the proposed web crawler can learn from experience
to better gather the relevant web pages during the early stage of successive crawling attempts. The knowledge bases are categorized into three types as follows:

- **Starting URLs:** The crawler must collect as many relevant web pages as possible; it needs a set of good URLs, which point to a large number of relevant web pages as the starting point. Using the search engine seems to be a useful way to determine the starting URLs.

- **Topic keywords:** The crawler must compare a topic of interest with the content of collected web pages; it should have proper keywords to describe the topic of interest for comparison.

- **URL prediction:** The crawler must perform URL ordering for a set of un-downloaded web pages; it should have an ability to predict the relevancy of those pages before downloading. The crawler will calculate a prediction score by using the seen content of web page obtained from previous crawling attempts.

These knowledge bases are considered as the experience of the crawler. Then, the crawler provides more efficient result during the next crawling attempts. The author performs a search on the Google and selects the top 10 resulting URLs to be the initial set of starting URLs. The learnable web crawler employs three techniques. Firstly, the topic similarity indicates how similar web pages are compared to an interested topic. Secondly, the hub computation is used to tell how good hub web pages are and whether they should be appropriate starting URLs. Finally, the relevance judgment is used to decide which web pages are related to an interested topic. The learnable crawling process consists of three following phases:
- **First crawling phase:** It is a phase of gathering web pages without any prior knowledge base. During this phase, the crawler only has keywords describing an interested topic supplied by the user. The crawler then sends those keywords to a search engine to constitute the candidate set of starting URLs. Results of this phase are a collection of web pages and a crawling record.

- **Learning phase:** In this phase, the crawler tries to learn how to better collect relevant pages using previous experience. Starting URLs knowledge base is built by calculating a hub score for every collected web page, and choosing only the high hub ones. Topic keyword knowledge base is built by extracting keywords from relevant web pages, as well as from anchor texts of links, which point to relevant web pages. Finally, URL prediction knowledge base is built by computing topic similarity scores of the content of all collected web pages, and employing those scores in URL prediction process.

- **Consecutive crawling phase:** In this phase, the crawler collects web pages by using all knowledge bases. During this phase, good starting URLs and topic keywords are learned, URL ordering process is additionally done online using URL prediction knowledge base.

The algorithm is implemented in C language and uses the ODP directory as relevance judgment and evaluates the crawling performance by the ratio of relevant web pages found to the total number of web pages collected. There are some limitations in the study, namely; the chosen topics must only appear in the ODP directory, the volume of examined web pages is too small, etc.

Ahmed Patel and Nikita Schmidt (2011) have enhanced the performance of a focused (topic-specific) web robot by considering the structure of the documents.
downloaded by the robot. In the case of HTML, document structure is tree-like, defined by nested document elements (tags) and their attributes. By analyzing this structure, the text of some HTML elements has been used by the robot to schedule documents for downloading and hence the speed of convergence to a topic has been improved considerably. Three groups of standards are relevant to web crawling namely: document standards such as HTML, protocol standards such as HTTP, and the robot exclusion protocol, well known as “robots.txt”. In order to realize the topic-driven search, a more intelligent search system, based on the technologies of Peer-to-Peer is proposed by Hiroyuki (2004). They developed Mondou web search engine, which is based on the emerging technologies of data mining. The search engine provides associative keywords which are tightly related to focusing web pages. In order to improve the performance of various search strategies by using characteristics of web systems, they implement the advanced web information systems with data mining and information technologies.

The creation of specialized search engines (Steele 2001) search just for information on a particular topic or category on the web. Such search engines have smaller and more manageable indexes and have a powerful domain-specific search interface. The major search engines such as Google and AltaVista provide a very valuable resource for searching the web. However they are not always able to provide the interface to thoroughly search a specialized topic. The technical hurdle is that it is becoming increasingly difficult, if not impossible, for one search engine to index the entire contents of the web. This is also an economic hurdle – it is not cost-effective given the revenue a search engine can receive to build such an index. The two ways in which specialized search engines can add more to the searching experience are to (1) allow searching of pages that are currently not searchable from the major search engines at all and (2) provide more functionality and search power in the searching of already searchable pages. Each specialized search engine allows searching of just a subset of the pages currently not searchable from the major search engines. The two broad ways by which a search engine can be made specialized are: The first is to build
or make use of an index that is itself focused on some particular topic or category of information and the second is to present a search interface to users that supports and implements specialized searches within some category.

Yunming Ye (2004) presents a focused web crawling system iSurfer for information retrieval from the web. Different from other focused crawlers, the crawler uses an incremental method to learn a page classification model and a link prediction model. It employs an online sample detector to incrementally distill new samples from crawled web pages for online updating of the model learned. Other focused crawling systems use classifiers that are built from initial positive and negative samples and cannot learn incrementally. The performances of these classifiers depend on the topical coverage of the initial positive and negative samples. However, the initial samples, particularly the negative ones, with a good coverage of target topics are difficult to find. Therefore, the iSurfer’s incremental learning strategy has an advantage. It starts from a few positive samples and gains more integrated knowledge about the target topics over time. The experiments on various topics have demonstrated that the incremental learning method improves the harvest rate with a few initial samples.

In the search engine, keyword extraction is an important technique. It has important applications in web page retrieval, clustering and text mining etc. Aiming at the defects of the traditional keyword extraction algorithm namely; TF-IDF and pagerank algorithm, an improved weight computation strategy is proposed by Shexuebing (2013). The correctness of the keyword extraction is ensured by the improved weight algorithm. If a word appears frequently in a document, the characteristics of the entries can be a good representative of the class text; such a translation should give a higher weight to them, and as a feature of this type of text to distinguish with other types of documents. This is the deficiency of IDF. The shortcomings of pagerank are based on: algorithm stresses on the old web pages and easily leads to topic drift. In order to get better keyword extraction, here an improved
PageRank algorithm is improved. Because each word weight in the text is not the same, to the weight of each word has flexible assignment. For a word in an article, the word has the weight, refers to the maximum entropy word product to the number of words in the article appeared. In word segmentation to extract keywords article before, the article by division into sentence division, then word of each sentence, after the word weight is divided, with maximum entropy prior calculation word is calculated for each word in the text. This algorithm is based on the improved weight of the extracted keyword. The proposed algorithms in this research are analyzed and verified with the improved weight algorithm and the comparisons of the results are made to prove the efficiency of the proposed algorithms.

Yunhua et al (2005) proposed a method concerned with automatic extraction of titles from the bodies of HTML documents. It is desirable to conduct automatic extraction of titles from the bodies of HTML documents. They utilize format information such as font size, position and font weight as features in title extraction. This method significantly outperforms the baseline method of using the lines in largest font size as title. Titles are the ‘names’ of documents and thus are very useful information for document processing. In HTML documents, authors can explicitly specify the title fields marked by <title> and </title> tags. The method consists of two phases: training and extraction. There is pre-processing for training and extraction and also post-processing for extraction. The input of pre-processing is a document. The output of pre-processing is a sequence of instances. An instance corresponds to a line in the HTML document. It contains not only content information but also format information. In extraction, the input is a sequence of instances from one document. In post-processing of extraction, titles are extracted using heuristics. The output is the extracted titles of the document. Precision, recall, F1-score and accuracy are used in evaluation of title extraction results.
2.2.6 Snippet based Retrieval

Extracting a query-oriented snippet (or passage) and highlighting the relevant information in a long document can help reduce the result navigation cost of end users. While the traditional approach of highlighting matching keywords helps when the search is keyword-oriented, finding appropriate snippets to represent matches to more complex queries requires novel techniques that can help characterize the relevance of various parts of a document to the given query, succinctly. Qing Li et al. (2007) present a language-model based method for accurately detecting the most relevant passages of a given document. They focus on query-informed segmentation for snippet extraction. Qing Li et al. (2008) present language-model based methods for extracting variable length document snippets by real-time processing of documents using the query issued by the user. When searching the web for specific information, however, long documents returned as atomic answers may not be always an effective solution. First, a single document may contain a number of sub-topics and only a fraction of them is relevant to user information needs. Second, long documents returned as matches increase the burden on the user while filtering possible matches.

Many works focused on the post processing search results to facilitate users to examine the results. One of the common ways of post processing search result is clustering. Term-based clustering appears as the first way to cluster the results. But this method is suffering from the poor quality while the processed pages have little text. Link-based clustering can conquer this problem. But the quality of clusters heavily depends on the number of in-links and out-links in common. Nan Yang (2011) proposes that the short text attached to in-link is valuable information and it is helpful to reach high clustering quality. To distinguish them with general snippet, they named it as in-snippet. In this method, similarity between pages consists of two parts: link similarity and term similarity. They study the combination of link analysis and in-snippet information and clustering of web search results. They do parsing and preprocessing for each in-snippet as follows: (1) extract all unique words
from a in-snippet, (2) eliminate non-content-bearing stop words, (3) each word is transformed using stemming algorithm and (4) count the number of occurrences of each word.

In a federated web search setting (Thomas et al 2013), caching large amounts of result snippets is more feasible than caching entire pages. The experiments reported make use of result snippets and pages from a diverse set of actual web search engines. A linear classifier is trained to predict the snippet-based user estimate of page relevance, but also, to predict the actual page relevance, again based on snippets alone.

2.3 CONCLUSION

In this chapter, survey is made on selected publications from the related works that are relevant for this thesis. The thesis focuses on link analysis in web crawling. In the literature, link analysis is an active research topic in the information retrieval community. The web is very important today because it is the cornerstone of the information age, and is used by millions of pupil every day. Link analysis is, in a sense, the most important new component of the web in relation to previous document collections and traditional information retrieval, and probably this explain why the field of link analysis has been so active.

The following chapters show the main part of this thesis by presenting new crawling algorithms. The proposed algorithms are based on topic-specific crawling using hyperlink analysis to fetch the most relevant pages at the earlier stage of crawling.