International Journal of Advanced Computer Research (ISSN (print): 2249-7277 ISSN (online): 2277-7970)
Volume-2 Number-4 Issue-6 December-2012

A Review of Focused Web Crawling Strategies

Bireshwar Ganguly[1], Rahila Sheikh[2]
Department of Computer Science & Engineering[1], Department of Computer Science &Engineering[2]
RCERT, Chandrapur, R.T.M. Nagpur University, India[1,2]

Abstract

Modern world with tons of competition also brings a sense of responsibility of preserving the valuable time of user in case of searching for information around the web. But the abundance of data indexed is quite huge and with different user perspective, searching has a significant impact using a standard exhaustive crawling. A standard crawler starts well with a promising set of initial seed URLs but the amplitude of its graph decline in between the process. This is major reason why researches place heavy emphasis on the relevancy and robustness of the data found. Also the users' perspective differs from time to time from topic to topic. i.e. ones' want is others unnecessary. This is where the importance of Focused crawling comes into play. Focused crawlers aim to search and retrieve only the subset of the world-wide web that pertains to a specific topic of relevance. The ideal focused crawler retrieves the maximal set of relevant pages while simultaneously traversing the minimal number of irrelevant documents on the web. In this paper we review the researches on several focused web crawling strategies and propose a new technique which focuses on the assignment of credits to the web pages as per its semantic contents. We also give emphasis to prioritize the frontier queue so that the higher credit page URLs are given priority to crawl over lower one.

Keywords

Web crawling algorithms, search engine, focused crawling algorithm survey, page rank, Information Retrieval.

1. Introduction

The size of the world-wide-web has provably surpassed 9.38 billion pages documents [24] and as yet growth shows no sign of leveling off. A web surfer starts searching with the use of an internet search engine for his requirement around the web submitted in the form of keywords or *Search Query*. Search engine in turn search its database looking for the match and produces hundreds and thousands of results in front of the user. Now it's users' persistence to go through all this results and find the most relevant information as per his set criteria. But this imposes a bigger challenge for search engine for sorting the results in order of interestingness of user within the first page of appearance.

Now a question arises, who populates the database of search engine with predefine set of web pages? The answer is an automated program called Crawler [20] which does this job on the behalf of search engine behind the scene.

This paper is organized as follows; Section 2 introduces the fundamentals of web crawling, Section 3 introduces the concept of Focused crawling; Section 4 overview the existing techniques in crawling, Section 5 discusses our proposed work and finally we draw some conclusion in Section 6.

2. Fundamentals of Web Crawler

Crawlers [20] form the crucial component of search engine with a primary job of traversing the web & retrieving web pages to populate the database for later indexing and ranking .More specifically the crawler iteratively performs the following process:

1. Download the Web page.
2. Parse through the downloaded page and retrieve all the links.
3. For each link retrieved, repeat the process.

Now let's look at each step of the process in more detail.
In the ***first step***, a Web crawler starts with a list of URLs to visit called *Seeds* and downloads the respective page from the Internet at the given URL. Oftentimes the downloaded page is saved to a file on disk or put in a database. Saving the page allows the crawler or other software to go back later and manipulate the page, be it for indexing words or for archiving the page for use by an automated archiver.
In the ***second step***, a Web crawler parses through the downloaded page and retrieves the links to other pages. Each link in the page is defined with an HTML anchor tag similar to the one shown:
<A
HREF="http://www.host.com/directory/file.html">Li

nk</A>.After the crawler has retrieved the links from the page; each link is added to the list of URLs to visit called the *Queue or Frontier*.
The ***third step*** of Web crawling repeats the process. Crawlers work in a recursive or loop fashion.

3. Focused Web Crawling

Given the current size of the Web, even large search engines cover only a portion of the publicly-available Internet; a study by Lawrence and Giles [25] showed that no search engine indexes more than 16% of the Web. As a crawler always downloads just a fraction of the Web pages, it is highly desirable that the downloaded fraction contains the most relevant pages and not just a random sample of the Web. The most prominent challenge faced by the current web crawlers is to select important pages for downloading. The crawler cannot download all pages from the web. It is important for the crawler to select the pages and to visit "important" pages first by prioritizing the URLs in the queue properly.
Other challenges are the proper refreshing strategy, minimizing the load on the websites crawled and parallelization of the crawling process This is the reason why the concept of ***Focused Crawling*** was first coined by S.Chakrabarti in [1].

A *focused crawler* tries to identify the most promising links, and ignores off-topic documents. If the crawler starts from a document which is *i* steps from a target document, it downloads a small subset of all the documents that are up to *i*-1 steps from the starting document. If the search strategy is optimal the crawler takes only *i* steps to discover the target.

In contrast of *standard crawler*, which follows each link, typically applying a breadth first strategy? If the crawler starts from a document which is *i* steps from a target document, all the documents that are up to *i*-1 steps from the starting document must be downloaded before the crawler hits the target.

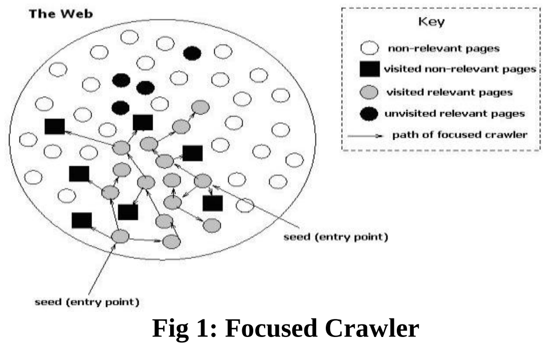

Fig 1: Focused Crawler

In order to achieve topic specialization of high quality, the logic behind focused crawling tries to imitate the human behavior when searching for a specific topic. The crawler takes following features into account of the web that can be used for topic discrimination including:

- the relevance of the parent page to the subject
- the importance of a page
- the structure of the web
- features of the link and the text around a link
- the experience of the crawler

4. Literature Review

The first generations of crawlers [22] on which most of the web search engines are based rely heavily on traditional graph algorithms, such as *breadth-first* or *depth-first* traversal, to index the web. A core set of URLs are used as a seed set, and the algorithm recursively follows hyper links down to other documents. Document content is paid little heed, since the ultimate goal of the crawl is to cover the whole web. However, at the time, the web was two to three orders of magnitude smaller than it is today, so those systems did not address the scaling problems inherent in a crawl of today's web.
Depth-first crawling [22] follows each possible path to its conclusion before another path is tried. It works by finding the first link on the first page. It then crawls the page associated with that link, finding the first link on the new page, and so on, until the end of the path has been reached. The process continues until all the branches of all the links have been exhausted.
Breadth-first crawling [2] checks each link on a page before proceeding to the next page. Thus, it crawls each link on the first page and then crawls each link on the first page's first link, and so on, until each level of links has been exhausted.

In contrast to traditional approaches, ***a focused crawler*** [1] efficiently seeks out documents about a specific topic and guides the search based on both the content and link structure of the web .It implements a strategy that associates a score with each link in the pages it has downloaded. The links are sorted according to the scores and inserted in a queue. A best first search is performed by popping the next page to analyze from the head of the queue. It is administrated by a classifier and a distiller. Based on topic taxonomy of the web and respective examples, as well as the user feedback, the classifier learns to

estimate the relevance of the crawled pages while the distiller identifies pages which are the main nodes of a specific topic. To begin, the system is fed with a topical taxonomy of the web such as the Yahoo! or the ODP and is trained by a set of examples. The user also provides a set of pages of his interest such as the links of his bookmark files. The system automatically classifies these pages and through an interactive process, the user can refine the categories and correct the systems decisions. Additionally, the user chooses the categories of his interest and marks them as '*good*'. After refinement, an interactive exploration of the web takes place. The system proposes some pages in the neighborhood of the examples that are considered to be similar and the user may include some of these pages to the examples. The system's classifier is trained upon the examples and crawls the web looking for pages of the 'good' categories
Throughout the page discovery process the system distils the pages, identifying the ones that link to many relevant pages. It then increases the visiting priority of the hub pages and their neighborhood. The user has the capability of monitoring the pages that are considered popular and give his positive or negative feedback to the system. This feedback is then used for the further refinement of the classifier and distiller.

However, the concept of prioritizing unvisited URLs on the crawl frontier for specific searching goals is not new, and Fish-Search [3] by De Bra et al. and Shark-Search [4] by Hersovici et al. were some of the earliest algorithms for crawling for pages with keywords specified in the query.
In ***Fish-Search***, the Web is crawled by a team of crawlers, which are viewed as a school of fish. If the ''fish'' finds a relevant page based on keywords specified in the query, it continues looking by following more links from that page. If the page is not relevant, its child links receive a low preferential value.
Shark-Search is a modification of Fish-search which differs in two ways: a child inherits a discounted value of the score of its parent, and this score is combined with a value based on the anchor text that occurs around the link in the Web page.

Naive Best First method proposed by Pant, G [19] [20] exploits the fact that relevant pages possibly link to other relevant pages. Therefore, the relevance of a page *a* to a topic *t*, pointed by a page *b*, is estimated by the relevance of page *b* to the topic *t*. Each page is represented as a vector of weights corresponding to the normalized frequencies of the document's terms according to the tf-idf scheme. The term frequency (tf) part, which calculates the frequency of a term within a document and the inverse term frequency (idf) part, which determines the importance of the term throughout the whole collection.

As already mentioned, beyond relevance, one of the main desirable features a focused crawler should have is the ability to download 'important' pages first. The crawler should fetch not just relevant pages, but high quality relevant pages. To pursue this, measuring the importance of a page is very necessary.

Page rank algorithm proposed by Brin, S. and Page, L., [5] [6] determines the importance of the web pages by counting citations or back links to a given page. The page rank of a given page is calculated as:
$PR(A) = (1-d) + d\,(PR(T1)/C(T1) + \ldots + PR(Tn)/C(Tn))$
PR (A) -Page Rank of a Website,
d -damping factor
T1….Tn –links.

The ***HITS algorithm***, proposed by Kleinberg [7] is another method for rating the quality of a page. It introduces the idea of *authorities* and *hubs*. An authority is a prominent page on a topic. They are the target of the crawling process since they have high quality on a topic. A hub is a page that points to many authorities. Their characteristic is that it's out links are suggestive of high quality pages. Hubs do not need to have high quality on the topic themselves or links from *'good'* pages pointing to them. The idea of the 'hub' is a solution to the problem of distinguishing the '*popular*' pages, from the authoritative pages. Therefore, hubs and authorities are defined in terms of mutual recursion.

Info Spiders, a dynamic web search multi agent system proposed by Pant, G. and Menczer [24]. Info Spiders complement traditional index based search engines using agents at the user side. These agents act autonomously with each other and they try to achieve a good coverage of the relevant documents. When the user submits a query, Info Spiders obtain a set of seed links which are the search results of a traditional search engine. An agent is initialized for every link and analyses the corresponding page's links looking for the next one to follow. The agent analyses the links by computing the similarity of the text around the link with the query, with the help of a neural net. The next link to be followed is chosen with a probability proportional to the similarity score.

The neural net weights are adjusted by the relevance of the new page's content so that the agent updates its knowledge.

Intelligent crawling proposed by Aggarwal et al. with arbitrary predicates is described in [8]. The method involves looking for specific features in a page to rank the candidate links. These features include page content, URL names of referred Web page, and the nature of the parent and sibling pages. It is a generic framework in that it allows the user to specify the relevant criteria. Also, the system has the ability of self-learning, i.e. to collect statistical information during the crawl and adjust the weight of these features to capture the dominant individual factor at that moment.

Ontology based focused crawling proposed by Ehrig, M. and Maedche [9] utilizes the notion of ontologies in the process of crawling. It consists of two main processes which interact with each other, the *ontology cycle* and the *crawling cycle.* In the ontology cycle, the crawling target is defined by ontologies (provided by the user) and the documents that are considered relevant as well as proposals for the enrichment of the ontology are returned to the user. The crawling cycle retrieves the documents on the web and interacts with the ontology to determine the relevance of the documents and the ranking of the links to be followed.

Metadata based focused crawling was proposed by Zhuang, et al. [10]. The purpose of the crawler was to harvest missing documents of digital library collections. The crawler could therefore be used to build a complete collection of documents of a given venue i.e. a journal or a conference. The document's metadata are used to locate the home pages of the authors, which are then crawled in order to find the target.

Language focused crawling proposed by Medelyan, O et al. [11] uses a language classifier which determines whether a page is worth preserving, is incorporated into the crawling process. The crawler is build for the creation of topic specific corpora of a given language in two steps. During the first step, a training set of documents which satisfy the language and topic requirements is created in order to extract the most distinguishing ngrams (the ngrams with the highest tf-idf values).
These ngrams are used as queries to a standard search engine and the results are used as the seed links. In the second phase, a classifier is incorporated in the crawler. The classifier is trained by the training set and domain models are created. A page is considered relevant if it belongs to the desired language and domain model.

The Document Object Model (DOM) crawler proposed by Pant, G.et al. [12] Uses the information of the tag tree representation of a page to determine the priority of its out links. The crawler builds *aggregation nodes* for each link of the page and estimates the possibility that the link leads to a relevant page, by the textual information of the aggregation node. When a page is downloaded, the crawler constructs its tag tree representation. A node that is on the path from the root of the tree to an out link of a page is considered to be the aggregation node and all the text that appears to the aggregation node sub tree is treated as the context of the link.

The Context Graph Crawling method, proposed by Diligenti et al. [13] uses backlinks to estimate the link distance from a page to target pages. Their method starts from a set of seed documents, follows backlinks to a certain layer, and then builds up a context graph for the seed pages. A classifier is constructed for each layer using the Naive Bayes algorithm. As a new document is found, it is classified into a layer. Documents classified into layers closer to the target are crawled first. The experiments showed that this approach maintained a higher level of relevance in the retrieved Web pages. However, the assumption that all pages in a certain layer from a target document belong to the same topic described by a set of terms does not always hold.

Reinforcement Learning (RL) Crawler proposed by Rennie and McCallum [17] [18] used to train a crawler on specified example web sites containing target documents. The web site or server on which the document appears is repeatedly crawled to learn how to construct optimized paths to the target documents. However, this approach places a burden on the user to specify representative web sites. Initialization can be slow since the search could result in the crawling of a substantial fraction of the host web site. Furthermore, this approach could face difficulty when a hierarchy is distributed across a number of sites.

Neural Networks extends the ***RL*** method for focused crawling is proposed by Grigoriadis, A. [19]. In their approach, each web page is represented by a set of 500 binary values, and the state of each page is determined by Temporal Difference Learning, in order to minimize the state space. The relevance of the page depends on the presence of a set of keywords within the page. A neural network is used for the estimation of the values of the different stages. During training session, the crawler randomly follows pages for a defined number of steps or until it reaches a relevant page. Each step represents the implementation of action αt which moves the agent from state st to state st+1. The respective reward rt+1 and the features of the state st are used as input to the neural network, which is trained to evaluate a state's potential of belonging to a successful path. During crawling mode, the crawler maintains a priority list of links to be followed, where the priorities are computed by the neural network. Since it is ineffective to download the children pages of the current page the crawler is at, the state value of a children page is inherited by the value of its parent (the current page) or by the average value of its parents, in case the page is pointed by more than one page.

Concept graph focused crawling proposed by Liu, H., Milios, E. & Janssen, J., [15] states the analysis of the link structure and the construction of a concept graph of the semantic content of relevant pages with the help of Hidden Markov Models is proposed. Here, the user of the search engine has a more active role, since he is required to browse the web and train the system, by providing the set of the pages he considered interesting. The system aims at analyzing and detecting the semantic relationships that exist within the paths leading to a relevant page. It has three main steps: *user modeling, pattern learning* and *crawling*.
During user modeling, the user is asked to browse the web for pages he is interested in and mark them. The browsing paths are then analyzed and transformed into web graphs in order to exploit not only the content of the pages but the link structure as well. Each node is a page and the links between pages are the edges of the graph. It is worth noting that the user is asked to mark the interesting pages, which are not restricted to only the targeted topic.

Tunnelling proposed by Bergmark et al. [27] where the patterns existing within the document relation paths are exploited. The crawling phase starts with the definition of the topic hierarchy and the construction of the respective *centroids*. A centroid is a vector with weighted terms, describing a category. In order to build the centroid, the system queries Google and collects the first *k* first results on each category, for a number of categories. Since the representation of a subject can be done with a small set of representative documents but for categories with sufficient presence, *k* is bounded between [27] (Google should return at least 4 documents while the results after the 7th document are ignored). The vectors are constructed by using the tf-idf scheme, concatenating the k documents of each category and comparing the term frequencies in a category with the term frequencies within all the categories.

Decision trees another method taking into account the anchor text of the links is proposed by] Li, J., Furuse, K. & Yamaguchi, K.[16]. Their method is not suitable for large scale focused crawling; instead, it is designed to crawl into a limited portion of the web, e.g. the website of a university. The link prioritization is done with the help of a decision tree which is trained by a web graph provided by the user. The user also indicates positive and negative examples (relevant and irrelevant pages) which are used for the training of a Support Vector Machine classifier. This classifier is used for computing the relevance of the pages in the graph. The decision tree construction is done by the ID3 method, analyzing the terms of the anchor text of all the links in the graph and the shortest paths of an entry page to the positive examples. Therefore, the tree distinguishes the '*promising*' anchor text, which is the anchor text of both relevant and irrelevant pages that lead to relevant pages and classifies the links into promising and not promising. Crawling is done in a best first manner, by examining the promising links first.

After studying the various approaches in literature we find that the major open problem in focused crawling is that of properly assigning credit to all pages along a crawl route that yields a highly relevant document. In the absence of a reliable credit assignment strategy, focused crawlers suffer from a limited ability to sacrifice short term document retrieval gains in the interest of better overall crawl performance. In particular, existing crawlers still fall short in learning strategies where topically relevant documents are found by following off-topic pages. Because of obvious disadvantages we propose a new technique to overcome the overall crediting system of focused crawling in the following section.

5. Proposed Work

Our work mainly focuses on the assignment of credits to the web pages as per its semantic contents. We also give emphasis to prioritize the frontier queue so that the higher credit page URLs are given priority to crawl over lower ones. Our proposed algorithm is as follows:

```
1)   Start to Select a set of seed URLs (Selected & Prioritized Manually) & insert in the frontier queue;
2)   If the frontier queue is non empty and Max URL s <Limit ,Download the Web page New() pointed by the topmost URL in the queue Else Stop;
3)   Initialize the PageScore() =0;
4)   Enter the Keyword(s) to be searched;
5)   Assign the Page Score() as follows:-
i)   If the Keyword(s) is present in the <Head> of the New() Webpage, increment PageScore() by 5 , Else PageScore () is unchanged;
ii)  If the Keyword(s) is present in the <Href > ( inside the hyperlink/URL) of the New() Webpage,, increment PageScore() by 2 , Else PageScore is unchanged;
iii) If the Keyword(s) is present in the <Body/Text> of the New () Webpage, increment PageScore () by 1 and repeat the process for every occurrence of the Keyword(s).
6)   The final PageScore () of the New () Webpage will be the cumulative score of Step (5).
7)   If the final PageScore () <=1, reject New () (irrelevance threshold), and Go to Step 2.
     Else if PageScore () is >= 2,
     If the PageScore () of New () >= Previous (), then extract all the hyperlinks of New () and insert at the top of the frontier queue
     Else append at the rear of the queue.

     Previous () ==New (); Go To Step 2.
```

In the proposed algorithm we give maximum emphasis on the <Head> of an document as it actually depicts the most significant part <Title> of the web page and if a keyword is present in the <Head> then surely the webpage we are searching is a relevant to our query. Second importance is given to the Hyperlink contents<Href> and finally to the frequency of the keyword appearing in the document. And last we are also focusing on improvisation of frontier queue so that the order of crawling can also be prioritized with higher relevant links are crawled over lower ones.

6. Conclusion

Thus we have studied various focused web crawling strategies, their mechanisms & issues. We also have discussed about the major drawback of all mentioned techniques regarding the overall performance of credit system and relevancy factor. Therefore we require an efficient technique to eradicate this drawback from the system, Hence we propose a new technique which are trying to develop and is expected to overcome limitation of existing Crawling techniques.

References

[1] Chakrabarti, S., van den Berg, M. & Dom, B., 1999a Focused crawling: a new approach to topic-specific Web resource discovery. In Proceeding of the 8th International conference on World Wide Web, Toronto, Canada, pp.1623.

[2] Najork, M. and Wiener, J., L., 2001. Breadth-First Search Crawling Yields High-Quality Pages. In 10th International World Wide Web Conference, pp. 114-118.

[3] De Bra, P. and Post, R., 1994. Information Retrieval in the World−Wide Web: Making Client−based searching feasible. In Journal on Computer Networks andISDN Systems, 27, pp. 183-192, Elsevier Science BV.

[4] Hersovici, M., Jacovi, M., Maarek, Y. S., Pelleg, D., Shtalhaim, M. & Ur, S., 1998. The shark-search algorithm an application: tailored Web site mapping. InProceedings of the seventh international conference on World Wide Web 7,Brisbane, Australia pp. 317 – 326.

[5] Brin, S. and Page, L., 1998. The Anatomy of a Large-Scale Hypertextual Web Search Engine. In Proceedings of the seventh international conference on World Wide Web 7.Brisbane, Australia pp. 107 - 117.

[6] Page, L., Brin, S., Motwani, R. & Winograd, T., 1998. The PageRank Citation Ranking: Bringing Order to the Web. Stanford Digital Library Technologies Project.

[7] Kleinberg, M. J., 1997. Authoritative Sources in a Hyperlinked Environment, In Proceedings of the 9th ACM-SIAM Symposium on Discrete Algorithm.

[8] Aggarwal, C., Al-Garawi, F. & Yu, P., 2001. Intelligent Crawling on the World Wide Web with Arbitrary Predicates, In Proceedings of the 10th international conference on World Wide Web, Hong Kong, Hong Kong, pp. 96 – 105.

[9] Ehrig, M. and Maedche, A., 2003. Ontology-Focused Crawling of Web Documents. In Proceedings of the Symposium on Applied Computing 2003 (SAC 2003), Melbourne, Florida, USA, pp. 1174-

[10] Zhuang, Z., Wagle, R. & Giles, C. L., 2005. What's there and what's not? Focused Crawling for Missing Documents in Digital Libraries. In Joint Conference on Digital Libraries, (JCDL 2005) pp. 301-310.

[11] Medelyan, O., Schulz, S., Paetzold, J., Poprat, M. & Markó, K., 2006. Language Specific and Topic Focused Web Crawling. In Proceedings of the Language Resources Conference LREC 2006, Genoa, Italy.

[12] Pant, G. and Menczer, F., 2003. Topical Crawling for Business Intelligence. In Proceedings of the 7th European Conference on Research and Advanced Technology for Digital Libraries.

[13] Diligenti, M., Coetzee, F.M., Lawrence, S., Giles, C.L. & Gori, M., 2000. Focused Crawling Using Context Graphs. In 26th International Conference on Very Large Database.

[14] Castillo, C., 2004. Effective Web Crawling. Ph. D., University of Chile.

[15] Liu, H., Milios, E. & Janssen, J., 2004. Focused Crawling by Learning HMM from User's Topic-specific Browsing. In Proceedings of 2004 IEEE/WIC/ACM international Conference on Web Intelligence, Beijing, China, pp. 732-735.

[16] Li, J., Furuse, K. & Yamaguchi, K., 2005. Focused Crawling by Exploiting Anchor Text Using Decision Tree. In Special interest tracks and posters of the 14th international conference on World Wide Web, Chiba, Japan.

[17] J. Rennie and A. McCallum, "Using Reinforcement Learning to Spider the Web Efficiently," In proceedings of the 16th International Conference on Machine Learning(ICML-99), pp. 335-343, 1999.

[18] Sutton, R. and Barto, A. 2002. Reinforcement Learning. An Introduction. MIT Press, Cambridge, MA.

[19] Grigoriadis, A. and Paliouras, G., 2004. Focused Crawling using Temporal Difference-Learning. In Proceedings of the Panhellenic Conference in Artificial Intelligence (SETN), Samos, Greece, pp. 142-153.

[20] Pant, G. and Menczer, F., 2003. Topical Crawling for Business Intelligence. In Proceedings of the 7th European Conference on Research and Advanced Technology for Digital Libraries.

[21] Pant, G., 2004a. Learning to crawl: Classifier-guided topical crawlers. Ph. D. The University of Iowa.

[22] Alexander Shen "Algorithms and Programming: Problems and solutions" Second edition Springer 2010, Pg 135.

[23] Pant, G. and Menczer, F., 2002. MySpiders: Evolve Your Own Intelligent Web Crawlers. In Autonomous Agents and Multi-Agent Systems, 5(2): pp. 221-229.

[24] www.worldwidewebsize.com/.

[25] Lawrence, Steve; C. Lee Giles (1999-07-08). "Accessibility of information on the web". *Nature* 400 (6740).

[26] http://en.wikipedia.org/wiki/Web_crawler.

[27] Bergmark, D., Lagoze, C. & Sbityakov, A., 2002. Focused Crawls, Tunneling, and Digital Libraries. In Proceedings of the 6th European Conference on Research and Advanced Technology for Digital Libraries, pp. 91-106.

Prof. B.S. Ganguly was born on 20Aug.1984.He has completed his B.E from Nagpur University and is currently pursuing his M.Tech from the same university. His research interest is Data Mining and Information Retrieval. He is currently working as Asst. Professor in RCERT; Chandrapur. He is also life member of Institution of Engineers & CSI.

Prof. Mrs. Rahila Sheikh was born on 06 May 1970.She has completed her M.Tech from Nagpur University and is currently pursuing her Ph.D from the same university. She has over 14 years of teaching experience as Asst.Professor in RCERT, Chandrapur. Her research interest includes Genetic Algorithms, Optimization Techniques & Data Mining.