A Frame Work for Topical Collections Make with Focused and Accelerated Focused Crawlers

Saturi Rajesh
Asst.prof, Dept of CSE
NNRESGI

D.Raju
Assoc.Prof & HOD
NNRESGI

P.Ajay Kumar
Lecturer,Dept of CSE
JNTUH-CEH

P.Srikanth
Asst.prof, Dept of CSE
NNRESGI

ABSTRACT
The rapid growth of the World-Wide Web poses unprecedented scaling challenges for general-purpose crawlers and search engines. In the personalized search domain, an alternative to general purpose crawler called focused crawlers are receiving increasing attention. The goal of these crawlers is to selectively seek out pages that are relevant to a pre-defined set of topics or theme. Rather than collecting and indexing all accessible Web documents to be able to answer all possible ad-hoc queries, these crawlers analyzes their 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. This paper presents and compares two focused crawlers called traditional focused crawler and accelerated focused crawler. Accelerated focused crawler takes offline lessons from traditional focused crawler. It emulates human surfer by trying to predict the relevance of a 'HREF' target page based on words around the link on the source page. The topics are specified using exemplary documents in these experiments. Naive Bayesian classifier is used to guide the crawlers. The crawlers were evaluated for different number of pages crawled, for different number of features gathered from different distances from the link and with different feature selection methods.

Index Terms
Focused Crawler, World Wide Web, and Accelerated focused crawlers.

1. INTRODUCTION
The World Wide Web is an ever expanding source of information that reflects the interests and views of a diverse global society. Many individuals, organizations, and institutions describe their interests, mission, activities, future plans, and research on their Web sites. There are sites that review companies or provide financial information about them. Others post customer reviews of products and services. Securities and Exchange Commission filings and other public documents are also available on the Web. Online digital libraries, tutorials, and white papers provide access to professional and technical information [1-3]. A characteristic of the Web is that it is not a simple collection of documents and pages. It is an interconnected collection. The connections between pages that are made through hyperlinks facilitate manual navigation or surfing over the Web. Hence, in the abstract, the Web can be seen as a graph with pages forming its nodes and hyperlinks forming its edges as shown in Fig 1. Since hyperlinks have a source page and a destination page, the edges of the Web graph have directions as shown in Fig 1. The Web is a constantly changing entity. The content of Web pages may change; new pages are added to the Web while many others disappear from it over time. The hyperlinks between pages may also vary with time. This dynamism of the Web is largely uncontrolled. This lack of control over the Web also brings up the issue of information quality and reliability. Sometimes we find pages with little or no information. At other times, they contain unreliable or incorrect information. On other occasions we find very valuable and accurate information. Hence, the large size, dynamism, and uncontrolled nature of the Web offer new challenges for information handling, retrieval, and usage [4-6].

This paper is on the design and evaluation of a method for building collections from the Web that are about a given topic. We shall begin the study of these methods by exploring the contributions of search engines and orienting the reader to Web crawlers.

1.1 Search Engines
General purpose search engines like Google and Yahoo! are popular tools for information search over the Web. A search engine captures and updates a snapshot of a portion of the Web by automatically downloading pages at regular intervals. At any given time, a search engine has a set of previously downloaded pages that it can use to approximate the current state of the Web. Based on this approximation, the search engine attempts to provide relevant results to diverse user queries. As mentioned earlier, the Web has pages with varying quality and reliability of information. This creates the problem of irrelevant results from search engines [7-8].

Google [9-12] pioneered an innovative technique of ranking search engine results based on the graph structure of the Web that not only considered the content in pages but also their hyperlinks. This successful mechanism allows Google to more effectively handle the Web's uncontrolled nature. Due to the large number of pages that appear relevant to a given user query, it is important to provide the most reliable pages as the top few results. Google displays the relevant results in decreasing order of their estimated popularity. One way to estimate popularity of a given page is to count the number of pages that have a hyperlink to the given page. Each hyperlink is seen as a vote that vouches the popularity of the given page. Google's popularity measure extends this simple idea through a more sophisticated algorithm based on the graph built from the downloaded pages. These generic search engines attempt to download an ever increasing fraction of Web pages so that they can cater to a diverse user base. While for many user queries such a technique will work well, one may question the effectiveness of this one-size-fit-all technique in catering to specific information needs of a user, a community, or an organization. When the information needs are specific,

possibly requiring in-depth research, there is a need to build **niche search engines**. There is also growing interest in knowledge discovery systems that typically rely on topical collections [13-15]. A collection of Web pages is said to form a **topical collection** when it is primarily focused on a topic or theme. Description of the need for topical collections, and the techniques involved in effectively building them from the Web are discussed in the next sections.

1.2 Topical Collections

There are several advantages to building topical collections. In a large collection of billions of pages the problems of polysemy and synonymy will tend to be far greater when compared to a focused topical collection. For example, if the word "chips" appears on a page within a general collection it has several potential meanings. But this space of possible meanings would be reduced if the collection is primarily focused on electronics. Similarly, the words "robots", "crawlers", and "spiders" may not appear to be synonyms in a generic collection. However, in a topical collection that has pages about Web crawler technology; these words would be considered synonyms. Hence, topical collections are more likely to provide homogeneous contexts which can help reduce the space and complexity of information search strategies. Topical collections can also facilitate knowledge discovery. For example, a topical collection may capture information associated with a research area. Such a collection may then be used to find serendipitous connections and relationships between researchers, concepts, and communities that were previously unknown or implicit [16-18]. The relationships may be found even if the content providers made no proactive attempt to convey the same. In a large generic collection such a hypotheses generation process would be extremely hard because of the large nature of the search space. For the same reasons, a topical collection would also facilitate testing of hypotheses as for example, "Is University Z associated with research topic W?" One may conceptualize similar knowledge discovery tasks in various other domains such as homeland security and business intelligence. Search engines can also benefit from topical collections if they cater to individual users or communities that have focused interests. Such niche search engines will have the advantage of scalability since they need to capture a much smaller portion of the Web, one that is likely to be easier to maintain and update

1.3 Crawlers

Web crawlers are programs that exploit the graph structure of the Web to move from page to page and to download them. In their infancy such programs were also called wanderers, robots, spiders, fish, and worms, words that are quite evocative of Web imagery. The noun 'crawler' is not indicative of the speed of these programs, as they can be considerably fast. In its simplest form a crawler starts from a seed page and then uses the hyperlinks within it to attend to other pages. The process repeats with the new pages offering more hyperlinks to follow, until a sufficient number of pages are identified or some higher level objective is reached. Behind this simple description lies a host of issues related to network connections, spider traps, parsing HTML pages, and the ethics of dealing with remote Web servers. In fact a current generation Web crawler can be one of the most sophisticated yet fragile parts [9] of the application in which it is embedded. The crawlers that are developed in this project follow ethical spider behavior, consume a small fraction of the available bandwidth, and are faster than a sequential crawler. As noted earlier, general purpose search engines serving as entry points to Web pages strive for coverage that is as broad as possible. They use Web crawlers to maintain their index databases amortizing the cost of crawling and indexing over the millions of queries they receive. These crawlers are blind and exhaustive in their approach, with comprehensiveness as their major goal. In contrast, crawlers that are selective about the pages they fetch are referred to as preferential or heuristic based or focused crawlers. Among other things, these may be used for building topical collections. There is a vast literature on preferential crawling applications, including preferential crawlers built to retrieve topical collections are called topical or focused crawlers. Synergism between search engines and topical crawlers is certainly possible with the latter taking on the specialized responsibility of identifying subspaces relevant to particular communities of users. Techniques for preferential crawling that focus on improving the "freshness" of a search engine has also been suggested in literature [19, 21].

2. LITERATURE SURVEY

Information on the web can be found in two ways. One is by surfing and the other is by the use of search engines. Web surfing is feasible due to the fact that most pages link to similar pages. Some recent work by Menczer [3, 4] provides interesting insights into the relationship between content similarities and relatedness among Web pages. He finds that both content and links provide a weak yet significant signal about the (semantic) relatedness of Web pages [18-19]. Another aspect that makes Web surfing possible is that the contextual information found around the hyperlink is suggestive of the destination page. For example, words within a hyperlink or around it tend to describe some of the aspects of the destination page. Not many would use the Web if these properties were not by and large true. Search engines are second method for retrieving information from the web. They mine the data available on the web for displaying search results. The main components of any search engine are crawler, Indexer and Query processor. Web crawlers are programs that exploit the graph structure of the Web to move from page to page and to download them. Crawlers start from a given set of URL's progressively fetch and scan them for new URLs (out links) and then fetch these pages in turn in an endless cycle. New URL's found thus represent potentially pending work for the crawler. The set of pending work expands quickly as the crawl proceeds, and the data that is fetched by crawlers are written to disk to relieve main memory as well as guard against data loss in the event of a crawler crash, the actual architecture is shown in Fig 1.

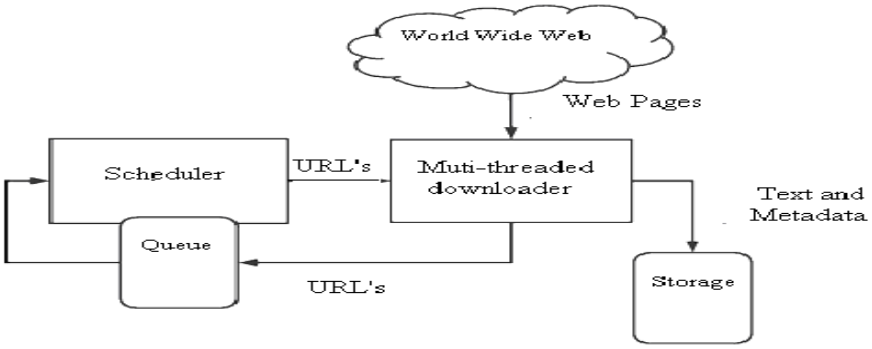

Fig. 1. Simplified architecture of a web crawler

2.1 Review of the State of the Art

The crawlers used by generalized search engines uses the breadth first (BRFS) or exhaustive crawling strategy. The **BRFS** 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 [17].

Systems on the other hand that require high precision and targeted information must seek new pages in a more intelligent way. The crawler of such a system called focused or topic-driven or preferential 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. The importance metrics can be either *interest driven* where the classifier for document similarity checks the text content or *popularity/location* driven where the importance of a page depends on the hyperlink structure of the crawled document. The Best first search technique discussed above comes under the crawling techniques that make use of web page content to find the relevancy of page to be crawled.

Chakrabarti et.al [22] proposed accelerated focused crawler. In this method there are two crawlers. First one is called baseline and next one is called apprentice. **Baseline** is the traditional focused crawler that navigates through the web in order to build a database of classified parent and target pages. **Apprentice** is trained on this database and eventually able to guide the crawl by determining the relevancy of new links based on the local context around the HREF.

Aggarwal et al. [18] have proposed an "intelligent crawling" framework in which only one classifier is used. The classifier trains as the crawl progresses. The crawler learns characteristics of the linkage structure of the World Wide Web while performing the crawling. Specifically, the intelligent crawler uses the in linking web page content, candidate URL structure, or other behaviors of the in linking web pages or siblings in order to estimate the probability that a candidate is useful for a given crawl. These techniques 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.

Numerous techniques for focused crawling that try to combine textual and linking information for efficient URL ordering exists in the literature. Many of these are extensions to Page Rank and HITS. An extension to HITS where nodes have additional properties and make use of web page content in addition to its graph structure is proposed in [18] as a remedy to the problem of nepotism. Page Rank was proposed by Brin and Page [19] as a possible model of user surfing behavior. The Page Rank of a page represents the probability that a random surfer (one who follows links randomly from page to page) will be on that page at any given time. A page's score depends recursively upon the scores of the pages that point to it. Source pages distribute their Page Rank across all of their outlinks.

Formally:

$$PR(p) = (1-\gamma) + \gamma \sum_{\{d \in in(p)\}} PR(d) / |out(d)|$$

Where p is the page being scored, $in(p)$ is the set of pages pointing to p, $out(d)$ is the set of links out of d, and the constant $\gamma < 1$ is a damping factor that represents the probability that the random surfer requests another random page. As originally proposed Page Rank was intended to be used in combination with content-based criteria to rank retrieved sets of documents [19].This is the way Page Rank has been used in Google search engine. More recently **Page Rank** has been used to guide crawlers and to assess page quality. Page Rank requires a recursive calculation until convergence, thus its computation can be a very resource-intensive process. In the ideal situation we have to recalculate PageRanks every time a URL needs to be selected from the frontier. For improving efficiency PageRanks can be recomputed at regular intervals. In building Focused crawler (or Baseline or Best first crawler) and Accelerated Focused crawlers (or Apprentice) the general crawling infrastructure developed is used. This multi-threaded implementation took care of document retrieval, caching in the local file system and compliance with the robot exclusion protocol. This infrastructure can be used to build a wide variety of topical crawlers.

3. CRAWLING INFRASTRUCTURE

The aim of this system is to present a crawling system that helps in building topical collections in less time. The crawler is called accelerated focused crawler (Apprentice). Its performance is improved by appropriate offline training from another focused crawler called Baseline. Accelerated focused crawler adds links to the frontier with a score equal to relevance of text around the link in the parent page. Baseline

crawler adds links to frontier with a score equal to relevance of parent page. Accelerated focused crawler has downloaded more pages relevant to the topic and it has downloaded them in less time compared to the traditional focused crawler (focused crawler). This work is distinguished from prior art in following important ways:

Two classifiers: Two classifiers are used in this project unlike Aggarwal et al [18] who use only one classifier. The first one is used to obtain 'enriched' training data for the second one. The apprentice is a simplified reinforcement learner. It improves the harvest rate.

No manual path collection: Two-classifier framework essentially eliminates the manual effort needed to create reinforcement paths or context graphs. The input needed to start off a focused crawl is just a pre-defined topic taxonomy which is easily available on the web and a few focus topics.

Offline Training: Apprentice is trained offline with the documents collected by baseline crawler where as apprentice crawler Chakarbarti et al. [22] was trained continually online with the documents collected by baseline crawler.

No manual feature tuning: Rather than tune ad-hoc notions of proximity between text and hyperlinks, features of link (u,v) are encoded using the DOM-tree of u, and automatically learn a robust definition of 'nearness' of textual feature to (u,v). In contrast, Aggarwal et al [18]use many tuned constants combining the strength of text and link-based predictors. Features are taken as words around a link where as Chakrabarti et al. [22] used word-distance pairs as features.

This work differs from SharkSearch in that it do not use anchor text and score of ancestors in calculating the relevance of a link. Page Rank, HITS when used to guide the crawl are popularity/*location* driven approaches where as these crawlers are interest driven approaches. The breadth-first or random crawl would have a negligible fraction of positive training instances and it would also clearly lose its way very soon. The best first crawler is used as baseline crawler which is used to provide positive training instances to apprentice. This work is specifically useful in conjunction with the use of context graphs. When the context graph learner predicts that a goal is several links away, it is crucial to offer additional guidance to the crawler based on local structure in pages, because the fan-out at that radius could be enormous. This work can also be extended to include HITS algorithm. After the apprentice has crawled for certain amount of time it can be stopped and HITS can be applied on the graph structure of retrieved pages which identifies **hub sites**. Then the apprentice can be resumed to start the crawl from the hub sites so that more pages relevant to the topic can be fetched.

The importance of a crawling infrastructure stems from the fact that a topical crawler has diverse components. On one hand the crawler must tackle networking issues, and on the other it must be able to use complex heuristics possibly utilizing machine intelligence techniques. The infrastructure is implemented using layered design. Figure 2 details high-level layered design for topical crawling infrastructure. Each layer is oblivious to the implementation details of other layers; this makes it easy to replace one implementation of a layer with another as long as their interface is kept the same.

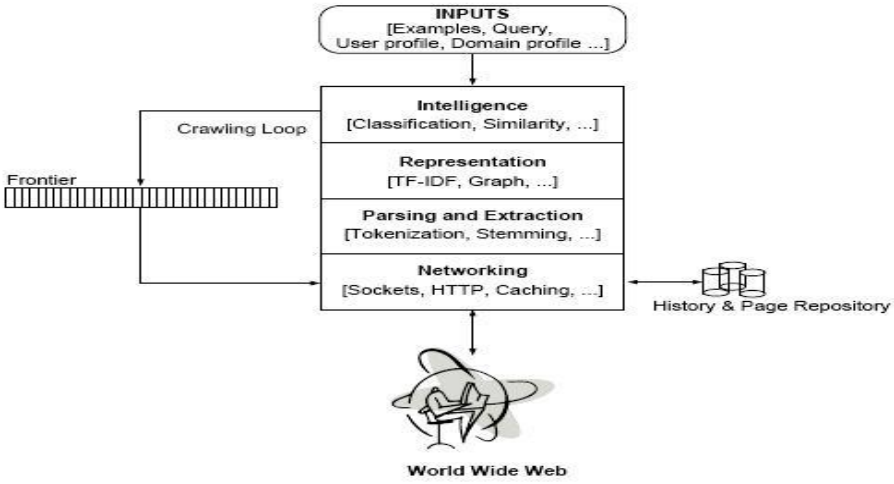

Fig. 2. Topical Crawling Infrastructure

In a crawling loop a URL is picked from the frontier (a list of unvisited URLs), the page corresponding to the URL is fetched from the Web, the fetched page is processed through all the infrastructure layers (see Figure 5.1), and the unvisited URLs from the page are added to the frontier. The networking layer is primarily responsible for fetching and storing a page and making the process transparent to the layers above it. The parsing and extraction layer parses the page and extracts the needed data such as hyperlink URLs and their contexts (words, phrases etc.). The extracted data is then represented in a formal notation (say a vector) by the representation layer before being passed onto the intelligence layer that associates priorities or scores with unvisited URLs in the page. In a general-purpose crawler the topmost two layers (representation and intelligence layers) may not be implemented. For example, the crawler may be an exhaustive crawler. However, for a topical crawler, the unvisited URLs from the fetched page are scored to reflect their potential value before being added to the frontier. Hence, a topical crawler includes an implementation of the representation layer and the intelligence layer. The score given to an unvisited URL estimates the benefit of visiting the page corresponding to the URL. The crawling process may be terminated when a certain number of pages have been crawled. If the crawler is ready to crawl another page and the frontier is empty, the situation signals a dead-end for the crawler. The crawler has no new pages to fetch and hence it stops. This high level design is used in construction of these crawlers. This design is further extended as shown in Figure 3 to include user interface layer to get the required inputs from user and a database layer to save the results of focused crawler (baseline) which are then used to train accelerated focused crawler (apprentice).

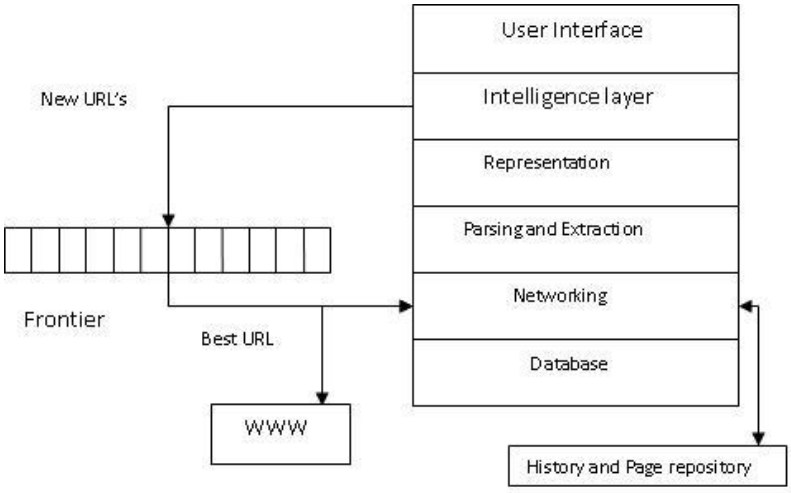

Fig. 3. Accelerated Focused crawler design.

The details of layers (and associated components) use in this design are described here. These crawlers may be modeled as graph search algorithms. As described before, the Web is seen as a large graph with pages at its nodes and hyperlinks as its edges. A crawler starts at one or more of the nodes (seeds) and then follows their edges to reach other nodes. The process of fetching a page and extracting the links within it is analogous to expanding a node in graph search. A topical crawler tries to follow edges that are expected to lead to portions of the graph that are relevant to a topic.

Inputs: The inputs to the intelligence layer initiate a crawl. In the intelligence layer the inputs may be used to suitably represent the desired information need, topic, or theme for which the crawler needs to fetch Web pages. An input can be as simple as a set of keywords or as complex as taxonomy with representative Web pages. A set of keywords can be converted into a vector of weights where each weight represents the relative importance of the word. If the input is a set of pages that represent positive and negative examples of a topic, it can be used to train a classifier. In this thesis, the input to the crawlers will be positive and negative examples. These positive and negative samples are collected from DMoz topical taxonomy. The URL's to train the classifiers as well as seed URL's are collected from **content.rdf** file.The input to Focused crawler (Baseline) is a set of seed URL's, database consisting of URL's relevant to topic, its parent and topic name and a feature file. Accelerated Focused Crawler (Apprentice) is trained on data or text documents gathered by baseline. Apprentice takes database constructed by baseline which has link, link's parent and topic information, cache directory which consists of web pages gathered by baseline, a file consisting of features collected during training phase and a set of seed URLs as inputs. The numbers of pages to be crawled, number of features to select, the type of feature selection, page repository name are given through user interface.

Frontier: The frontier is the to-do list of a crawler that contains the URLs of unvisited pages. In graph search terminology the frontier is an open list of unexpanded (unvisited) nodes. Although it may be necessary to store the frontier on disk for large scale crawlers, the frontier is implemented as an in-memory data structure for simplicity. Based on the available memory, one can decide on the maximum size of the frontier. Due to the large amount of memory available on PCs today, a frontier size can even be set to 10,000 URLs or more. The maximum frontier size is set to 10, 000 in these experiments. Given a maximum frontier size we need a mechanism to decide which URLs to ignore when this limit is reached. Note that the frontier can fill rather quickly as pages are crawled. One can expect around 60,000 URLs in the frontier with a crawl of 10,000 pages, assuming an average of about 7 links per page as measure. At times, a crawler may encounter a spider trap that leads it to a large number of different URLs that refer to the same page. One way to alleviate this problem is by limiting the number of pages that the crawler accesses from a given domain. The code associated with the frontier made sure that every set of k (say 100) URLs, picked by the crawler, are from k different fully qualified host names (e.g. www.cnn.com). As side-effects, the crawler is courteous by not accessing the same Web site too often, and the crawled pages tend to be more diverse.

URL Extraction and Canonicalization: HTML Parsers are freely available for many different languages. The parser available with Java SDK was used. These parsers provide the functionality to easily identify HTML tags and associated attribute-value pairs in a given HTML document. For example, to extract hyperlink URLs from a Web page, these parsers are used to find anchor tags and grab the values of associated "href" attributes. This code is used to get tokens or words in html page. To extract words around the href link the above logic is extended. Any relative URLs are converted to absolute URLs using the base URL of the page from which they were extracted. Different URLs that correspond to the same Web page can be mapped onto a single canonical form. This is important in order to avoid fetching the same page more than once.

4. RESULTS COMPARISON AND DISCUSSION

Offline analysis is done after all the pages have been downloaded and the experiments are over. Document frequency and mutual information feature selection methods are used in this project. By varying the feature selection method, 4 crawlers have been constructed and evaluated in this thesis. They are named as baselineDF, baselineMI, apprenticeDF and apprenticeMI.

These crawlers are evaluated for different number of pages crawled, for different number of features gathered and for features gathered from different distances from the link. The topics Food_Service, Model_Aviation, Nonprofit_ Resources, Mutual_ Funds, Roller-skating, Database Theory are divided

into 2 sets. The first three topics form one set and the last three topics from another set. All the crawlers have crawled for these two sets of topics. Recall and Precision are measured as described in table 1.The graphs are plotted. The horizontal axis in all the plots is time approximated by the number of pages crawled. The vertical axis shows the performance metric- precision or recall. Average recall and precision along with corresponding error bars are also shown in the plots.

4.1 Analysis:

Topics Crawled: **Food_Service, Model_Aviation, Nonprofit Resources**

Number of pages crawled: **10000**

Distance from the link from which features are gathered (only for apprentice): 10

Number of features used for training the crawlers: 500

The average target recalls of different crawlers by the time they have crawled **10000** pages are as shown below

TABLE I. METHOD VS AVG RECALL

Crawler	Average Target Recall
BaselineDF	5.3
BaselineMI	5.3
ApprenticeDF	11.5
ApprenticeMI	11

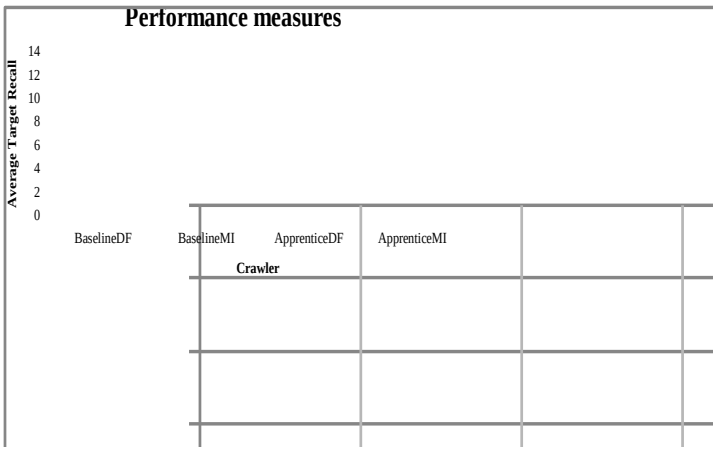

Fig. 4. performance of different methods

By the Table 1 data and fig 4 we can see that apprentice crawlers have outperformed baseline crawlers in finding the topic relevant target pages. All the four crawlers have found more number of topic relevant target pages for the topic Nonprofit Resources. This may due to the cooperative nature of this topic. Food Service and Model Aviation have competitive nature. When the crawlers are evaluated using the lexical similarity between crawled pages and topic descriptions the average recall values are as shown Table 2. By this data we can see that the apprentice crawlers have performed better than baseline crawlers.

TABLE II. METHOD VS AVG RECALL

Crawler	Average Recall
BaselineDF	390
BaselineMI	390
ApprenticeDF	650
ApprenticeMI	660

Topics Crawled: **Database Theory, Mutual Funds, Roller-skating**

Number of pages Crawled: **5000**

Distance from the link from which features are gathered (only for apprentice): 10

Number of features used for training the crawlers: 500, the average target recalls of different crawlers by the time they have crawled **5000** pages are as shown table III

TABLE III. METHOD VS AVG RECALL

Crawler	Average Target Recall
BaselineDF	5.5
BaselineMI	8
ApprenticeDF	12
ApprenticeMI	12

From the figure 5 this result may indicate that database theory is less popular topic in Web i.e. not many websites have relevant information about Database theory. Apprentice crawlers have shown better performance in this case also.

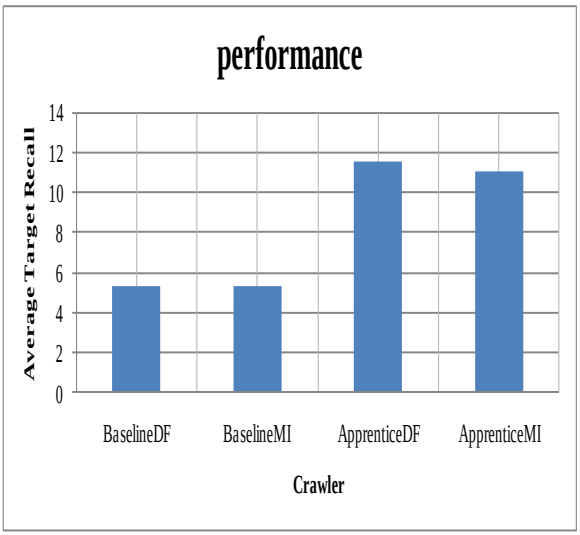

Fig 2 Performance of different methods

This shows that popularity of topic also effects crawler performance. When the crawlers are evaluated using the lexical similarity between crawled pages (whether or not they are in target set) and topic descriptions the average recall values are as shown below:

TABLE IV. METHOD VS AVG RECALL

awler	erage Recall
selineDF	0
selineMI	0
prenticeDF	0
prenticeMI	0

The recall values are in descending order of magnitude for the topics Mutual Funds, Roller Skating and Database Theory. This is because mutual funds is described briefly when compared to roller skating and roller skating is described briefly when compared to database theory by the DMoz editors. So it suggests that when crawlers are driven by keyword queries or topic descriptions it is better to describe topic or theme briefly using only and all prominent terms of the topic. This helps in collecting more number of pages relevant to the topic. By the above two results it is obvious that apprentice crawlers (accelerated focused crawlers) perform better than baseline crawlers (traditional focused crawlers).

5. CONCLUSIONS

Accelerated focused crawler through offline relevance feedback is a simple enhancement to a focused crawler. It assigns better priorities to the unvisited URLs in the crawl frontier. There is no need to manually train the system with the paths leading to relevant pages. The system is trained with documents relevant to the topic gathered from dmoz.org. Features are extracted from the DOM representation of parent (source) page which is simple compared to all other techniques. The crawlers have performed equally well with both document frequency and mutual information feature selection methods. Accelerated .crawler has outperformed focused crawler as the number of pages to crawl has exceeded pages. For all the 6 topics crawled accelerated crawlers has shown better target recall than focused crawler.

6. REFERENCES

[1] Yohanes, Banu Wirawan, H. Handoko, and Hartanto Kusuma Wardana, "Focused Crawler Optimization Using Genetic Algorithm", TELKOMNIKA (Telecommunication Computing Electronics and Control) , pp.403-410, 2013.

[2] Barbosa, Luciano, and Juliana Freire., "An adaptive crawler for locating hidden-web entry points", International conference on World Wide Web, pp. 441-450, 2007.

[3] Li, Yanni, Yuping Wang, and Jintao Du., "E-FFC: an enhanced form-focused crawler for domain-specific deep web databases", Journal of Intelligent Information Systems ,pp.159-184,2013.

[4] Bergmark, Donna, Carl Lagoze, and Alex Sbityakov, "Focused crawls, tunneling, and digital libraries", Research and Advanced Technology for Digital Libraries, pp. 91-106, 2002.

[5] Fu, Tianjun, Ahmed Abbasi, and Hsinchun Chen., "A focused crawler for Dark Web forums", Journal of the American Society for Information Science and Technology, pp.1213-1231,2010.

[6] Bedi, Punam, Anjali Thukral, and Hema Banati., "Focused crawling of tagged web resources using ontology", Computers & Electrical Engineering, pp.613-628, 2013.

[7] Liu, Jin-Hong, and Yu-Liang Lu. , "Survey on topic-focused Web crawler", Application Research of Computers ,pp.26-29,2007.

[8] Yakushev, Andrei V., Alexander V. Boukhanovsky, and Peter MA Sloot., "Topic crawler for social networks monitoring", Knowledge Engineering and the Semantic Web, pp. 214-227, 2013.

[9] Brin, S., Page, L., "The anatomy of a large-scale hypertextual Web search engine," In Computer Networks and ISDN Systems, pp.107–117, 1998.

[10] Goyal, Deepali, and Mala Kalra. "A novel prediction method of relevancy for focused crawling in topic specific search", International Conference on Signal Propagation and Computer Technology (ICSPCT), pp. 257-262, 2014.

[11] Wang, Wenxian, Xingshu Chen, Yongbin Zou, Haizhou Wang, and Zongkun Dai. , "A focused crawler based on naive bayes classifier", International Symposium on Intelligent Information Technology and Security Informatics (IITSI), pp. 517-521, 2010.

[12] Chuang, Hsiu-Min, Chia-Hui Chang, and Ting-Yao Kao. , "Effective Web Crawling for Chinese Addresses and Associated Information", Springer International Publishing on E-Commerce and Web Technologies, pp. 13-25, 2014.

[13] Yang, Sheng-Yuan. , "OntoCrawler: A focused crawler with ontology-supported website models for information agents", Expert Systems with Applications , pp.5381-5389,2010.

[14] Hati, Devashis, Biswajit Sahoo, and Amritesh Kumar, "Adaptive focused crawling based on link analysis", International Conference on Education Technology and Computer (ICETC), pp. V4-455, 2010.

[15] Achsan, Harry T. Yani, and Wahyu Catur Wibowo, "A Fast Distributed Focused-web Crawling", Procedia Engineering, pp.492-499, 2014.

[16] Dey, Manas Kanti, Hasan Md Suhag Chowdhury, Debakar Shamanta, and Khandakar Entenam Unayes Ahmed., "Focused web crawling: A framework for crawling of country based financial data", International Conference on Information and Financial Engineering (ICIFE), pp. 409-412, 2010.

[17] Najork, M., Wiener, J., "Breadth-first search crawling yields high-quality pages" , International conference on World Wide Web, pp.114-118,2001.

[18] Aggarwal, C., Al-Garawi, F.,Yu, P., "Intelligent Crawling on the World Wide Web with Arbitrary Predicates", Int. World Wide Web Conference, pp. 96-105, 2001.

[19] Bharat, K., Henzinger, M., "Improved algorithms for topic distillation in hyperlinked environments," In Proc. of the 21st annual international ACM SIGIR conference on Research and development in information retrieval, pp-104-111, Melbourne, Australia, pp.104-111, 1998.

[20] Brin, S., Page, L., "The anatomy of a large-scale hypertextual Web search engine," In Computer Networks and ISDN Systems, pp.107–117, 1998.

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

[22] Chakrabarti, S., Punera, K., Subramanyam, M., "Accelerated Focused Crawling through Online Relevance Feedback," International conference on World Wide Web Honolulu, pp.148-159, 2002.