On a cohesive focused and path-ascending crawling scheme for improved search results Ogban et al

Article · January 2013

CITATIONS
0

READS
27

3 authors, including:

Felix Ukpai Ogban
University of Calabar
24 PUBLICATIONS **1** CITATION
SEE PROFILE

Olumide Owolabi
University of Abuja
18 PUBLICATIONS **96** CITATIONS
SEE PROFILE

INTERNATIONAL JOURNAL OF NATURAL AND APPLIED SCIENCES (IJNAS), VOL. 8, NOS.1& 2 (2013); P. 20 – 26, 6 FIGS, 3 TABLES

On a cohesive focused and path-ascending crawling scheme for improved search results

F. U. Ogban[1*], P. O. Asagba[2], Olumide Owolabi[3]

ABSTRACT

The quality of the results collections, displayed to users of web search engines today still remains a mirage with regards the factors used in their ranking process. While the results are often amusing and expand users' horizons, they are often frustrating and consume precious time. The most important measure of a search engine is the quality of its search results. In this work, we combined the Focused crawling method developed by (Chakrabati; 1999) and Path-ascending crawling method of (Cothey; 2004), to create a hybridized method. Our major objective is to unify into one search system, the path-ascending and focused crawling methods to get into crawling the deep web at an efficient rate for a better recall and precision. The path-ascending and focused crawling systems respectively were aimed at producing increase recall and the function of similarity of a query to a matched document. The unification now is to help produce additionally the most authoritative pages in a search result and therefore increasing the precision thereof, thereby given enhance/increase authority and the strength of search/match of given queries to their corresponding documents in the web. The methodologies adopted are the Document and Query Likelihood Models. Documents or corpora of known measures in query types, recalls and precision from the Text Retrieval Conference (TREC), the Initiative for the Evaluation of XML Retrieval (INEX) and REUTERs Collection, were used as work bench for evaluation of the system. We obtained favorable results, which showed significant improvement from results if implemented using the Focused crawling method and path-ascending method as a single entity.

INTRODUCTION

The computer revolution has spawned so much information, that it is now to the point where the amount of information available on most subjects are typically so large; as to create the new and associated problems of going through that wealth of information and selecting from it the specific pieces of information most relevant to the question at hand. Retrieval of text information is a difficult task. The problem can be either that the information is misinterpreted because of natural language ambiguities or the information need can be imprecisely or vaguely defined by the user. This calls for improved automatic methods for searching and organizing text documents so information of interest can be accessed fast and accurately. Classification of web page content is essential to many tasks in web information retrieval such *as* maintaining web directories and focused crawling. The uncontrolled nature of web content presents additional challenges to web page classification as compared to traditional text classification, but the interconnected nature of hypertext also provides features that can assist the process.

Classification plays a vital role in many information management and retrieval tasks. On the Web, classification of page content is essential to focused crawling, to the assisted development of web directories, to topic-specific web link analysis, and to analysis of the topical structure of the Web.

Web page classification can also help improve the quality of web search. With the exponential growth of information on the World Wide Web, there is a great demand for developing efficient and effective methods to organize and retrieve the information available. A **crawler** is the program that retrieves Web pages for a search engine, which is widely used today. In this paper, we investigated the Focused crawling method developed by (Chakrabati; 1999) and Path-ascending crawling method of (Cothey; 2004). Our major objective is to unify into one search system, the path-ascending and focused crawling methods to get into crawling the deep web. The path-ascending and focused crawling systems respectively were aimed at producing increase recall and the function of similarity of a query to a matched document. The unification now is to help produce additionally the most authoritative pages in a search result and therefore increasing the precision thereof, thereby given enhance/ increase authority and the strength of search/ match of given queries to their corresponding documents in the web.

General review and problem definition

Page classification also known as web page classification is the process of assigning a page to one or more predefined category label. The field is often posed as a supervised learning problem (Mitchell, 1997) in which a set of labeled data is used to train a classifier which can be applied to label future examples.

*Corresponding author. Email: felix.ogban@gmail.com
[1]Department of Mathematics/Statistics & Computer Science, University of Calabar, Calabar, Nigeria
[2]Department of Computer Science, University of Port Harcourt. Nigeria
[3]Computer Center, University of Abuja, Nigeria

Retrieval of text information is a difficult task. The problem can be either that the information is misinterpreted because of natural language ambiguities or the information need can be imprecisely or vaguely defined by the user. This calls for improved automatic methods for searching and organizing text documents so information of interest can be accessed fast and accurately.

Path-ascending crawling

Some crawlers intend to download as many resources as possible from a particular Web site. (Cothey, 2004) introduced a *path-ascending crawler* that would ascend to every path in each URL that it intends to crawl. For example, when given a seed URL of http://llama.org/hamster/monkey/page.html, it will attempt to crawl /hamster/monkey/, /hamster/, and / etc. Cothey found that a path-ascending crawler was very effective in finding isolated resources, or resources for which no inbound link would have been found in regular crawling. Many Path-ascending crawlers are also known as Harvester software, because they are used to "harvest" or collect all the content - perhaps the collection of photos in a gallery - from a specific page or host.

Focused crawling

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. Web crawlers that attempt to download pages that are similar to each other are called **focused crawler** or **topical crawlers**. The concepts of topical and focused crawling were first introduced by (Menczer, 1997) and by (Chakrabarti *et al., 2003).* The main problem in focused crawling is that in the context of a Web crawler, we would like to be able to predict the similarity of the text of a given page to the query before actually downloading the page. A possible predictor is the anchor text of links; this was the approach taken by (Pinkerton, 1994) in a crawler developed in the early days of the Web. (Diligenti *et al, 2000)* propose to use the complete content of the pages already visited to infer the similarity between the driving query and the pages that have not been visited yet. 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.

Crawling the Deep Web

A vast amount of Web pages lie in the deep or invisible Web. These pages are typically only accessible by submitting queries to a database, and regular crawlers are unable to find these pages if there are no links that point to them. Google's Sitemap Protocol and mod oai (Nelson *et al.,* 2005) are intended to allow discovery of these deep-Web resources. Deep Web crawling also multiplies the number of Web links to be crawled. Some crawlers only take some of the <a href="URL" shaped URLS. In some cases, such as the Googlebot, Web crawling is done on all text contained inside the hypertext content, tags, or text.

The Crawlers

Focused crawling Algorithm

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. Web crawlers that attempt to download pages that are similar to each other are called **focused crawler** or **topical crawlers**. The concepts of topical and focused crawling were first introduced by (Menczer; 1997) and by (Chakrabarti *et al.2000).* The main problem in focused crawling is that in the context of a Web crawler, we would like to be able to predict the similarity of the text of a given page to the query before actually downloading the page. A possible predictor is the anchor text of links; this was the approach taken by Pinkerton in the first web crawler of the early days of the Web. (Diligenti *et al; 2000)* proposed using the complete content of the pages already visited to infer the similarity between the driving query and the pages that have not been visited yet. 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.

The focused crawling process

Because of limited computing resources and limited time, focused crawler has been developed. Focused crawler carefully decides which URLs to scan and in what order to pursue based on previously downloaded pages information. An early search engine which deployed the focused crawling strategy based on the intuition that relevant pages often contain relevant links. It searches deeper when relevant pages are found, and stops searching at pages not as relevant to the topic. Unfortunately, this traditional method of focused crawling show an important drawback when the pages about a topic are not directly connected.

A focused crawler traverses the web selecting out relevant pages to a predefined topic and neglecting those out of concern. While surfing the internet it is difficult to deal with irrelevant pages and to predict which links lead to quality pages. In this paper, a technique of effective focused crawling is implemented to improve the quality of web navigation. 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 also calculated based on meta data and resultant pages generated from focused crawler this is as shown in figure 1. The proposed work also uses a method for traversing the irrelevant pages that is met during crawling to improve the coverage of a specific topic.

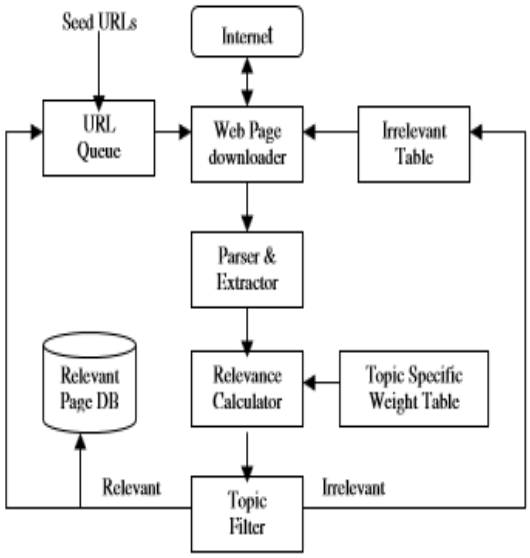

Fig. 1 . Architecture of focused crawling

The internet is the center from which pages are downloaded, a seed URL is given and queued into the Url queue. From the queue, the first URL is received by the Web page downloader and the corresponding page downloaded. The downloaded page is then subjected to parsing where the key parts of the page are extracted. Then the relevance calculator is invoked and the topics / the links filtered from the page are sent to the URL queue. Note that the relevance calculator uses the Topic Specific Weight Table to sifter the topics. Whereas, the pages found relevant are stored in the Relevance Database, those found irrelevant in this particular context are sent to the Irrelevant Table; and the circle continuous.

By this derivative, a pseudo code for the implementation of the algorithm is given below.

Pseudo code of Algorithm

```
1. if (page is Irrelevant)
2. {
3.  Initialize level = 0;
4.  url_list = extract_urls(page); // extract all the urls from the page
5. for each u in url_list {
6. compute LinkScore(u) using the link ranking equation;
7. IrrelevantTable.insert(u, LinkScore(u), level);
    // insert u into irrelevant table with linkscore and level value }
8. reorder IrrelevantTable accord to LinkScore (u);
9. while ( IrrelevantTable.size > 0)
10. {
11. get the url with highest score and call it Umax ;
12. if ( Umax.level < = maxLevel)
13. {
14.  page = downloadPage(Umax); // download the URL Um ax
15. calculate relevance of the page using equation for the measures of relevance;
16.if ( page is relevant) {
17.  RelevantPageDB = page; // put page into relevant page database
18.if ( Umax.level < maxLevel){
19.  level ++ ;
20. url_list = extract_urls(page);
21. for each u in url_list {
22. compute LinkScore(u)using the link ranking equation;
23.IrrelevantTable.insert(u, LinkScore(u), level); }
24reorder IrrelevantTable accord to LinkScore(u); }
25.} else {
26. for each u in IrrelevantTable {
27if (LinkScore(u) <= LinkScore(Umax) && u.level = = Umax.level)
28. u.level ++;}
29. }
30.} }}
```

Path-ascending crawling Algorithm

The path-ascending crawling, works like the regular shortest path algorithm along with weights. So we consider a site map of a collection of pages as an edge-weighted digraph.

An edge-weighted digraph is a digraph where we associate weights or costs with each edge. A *shortest path* from vertex ***s*** to vertex ***t*** is a directed path from ***s*** to ***t*** with the property that no other such path has a lower weight.

Properties:

We summarize several important properties and assumptions for Path-ascending crawling algorithm.

- *Paths are directed.* A shortest path must respect the direction of its edges.
- *The weights are not necessarily distances.* Geometric intuition can be helpful, but the edge weights weights might represent time or cost.
- *Not all vertices need be reachable.* If t is not reachable from s, there is no path at all, and therefore there is no shortest path from s to t.
- *Negative weights introduce complications.* For the moment, we assume that edge weights are positive (or zero).
- *Shortest paths are normally simple.* Our algorithms ignore zero-weight edges that form cycles, so that the shortest paths they find have no cycles.
- *Shortest paths are not necessarily unique.* There may be multiple paths of the lowest weight from one vertex to another; we are content to find any one of them.
- *Parallel edges and self-loops may be present.* In the text, we assume that parallel edges are not present and use the

notation v->w to refer to the edge from v to w, but our code handles them without difficulty.

METHODOLOGY

A document is a good match to a query if the document model is likely to generate the query, which will in turn happen if the document contains the query words often. This approach thus provides a different realization of some of the basic ideas for document ranking which could be applied through some acceptable rules:

i. A document or zone (Topic, Abstract, and Introduction etc) that mentions a query term more often has more to do with that query and should therefore receive a higher score in ranking.
ii. The exact ordering of the terms in a document is ignored but the number of occurrences of each term is material and we assign to each term in a document a weight for that term, which depends on the number of occurrence of the term in the document – term frequency.
iii. All terms are considered equally important when it comes to assessing relevancy on a query.

The Query Likelihood Model

The original and basic method for using language models in information retrieval is the query likelihood model. In it, we construct from each document **d** in the collection, a language model $\mathbf{M_d}$. Our goal is to rank documents by $P(d \mid q),$ where the probability of a document is interpreted as the likelihood that it is relevant to the query. Using Bayes rule in this context, we have:

$$P(d \mid q) = P(q \mid d)P(d)/P(q) \quad 1$$

However, since $P(q)$ and $P(d)$ is the same for all documents, they can be ignored. Thus above equation would be:

$$P(d \mid q) = P(q \mid d) \quad 2$$

But we could implement a genuine prior, which could include criteria like authority, length genre, newness, and number of previous people who had read the document. Given these simplifications, we return results ranked by simply $P(q \mid d),$ the probability of the query would be observed as a random sample from the respective documents model. The most common way to do this is using the multinomial unigram Language Model, which is equivalent to a multinomial Bayes model where the documents are the classes, each treated in the estimation as a separate "language":

$$P(q \mid M_d) = \prod_{w \in V} P(w \mid M_d)^{c(w,q)} \quad 3$$

For retrieval based on a language model, we treat the generation of queries as a random process. The approach is to:

i. Infer a Language Model for each document
ii. Estimate $P(q \mid m_i)$ the probability of generating the query according to each of these documents models.
iii. Rank the document according to these probabilities.

The intuition of the basic model is that the user has a prototype document in mind, and generates a query on words that appear in this document. Often, users have a reasonable idea of terms that are likely to occur in documents of interest and they will choose query terms that distinguish these documents from others in the collection. Collections statistics are an integral part of the language model, rather than being used heuristically as in many other approaches.

The hybrid method

Extra four major components, build from the paths- ascending algorithm/ assumptions were added to the Architecture of the focus crawling algorithm as is sketched in figure 2 below. The Site map component, the parallel edges and self-loop checker, the vertices and path normalizer and the weight indexer cum paths Tracer components.

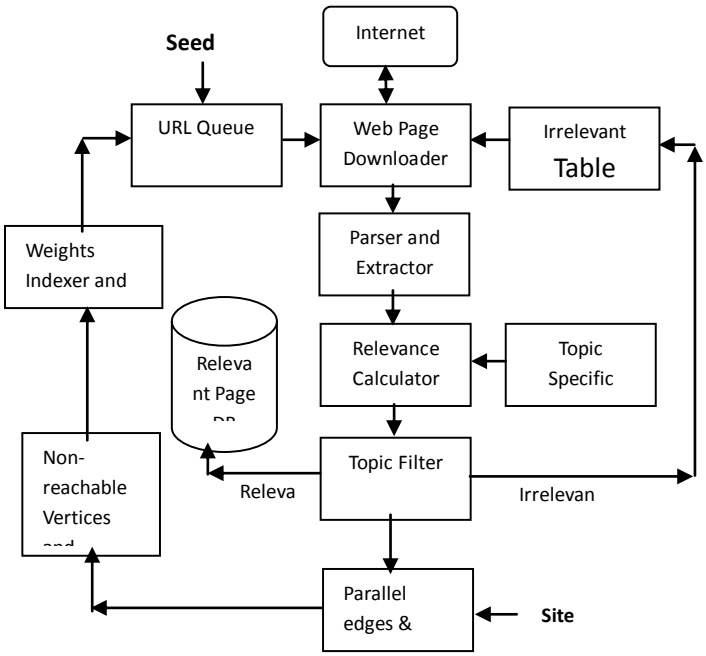

Fig. 2 . Architecture of the Hybrid method

The Parallel edges and self-loop checker is meant to check and resolve cycles – otherwise known as round loops in the sitemap.

RESULTS

Several results were produced from the series of data inputs. One of such is that shown in Table 1. A collection of ten(10) Boolean infested queries were formulated for input into the web search engines considered in the study. The TREC, INEX and REUTRES collections were the background document corpus for our search field. The result

in terms of the total number of matches found (TMF), the total pages that are relevant to the query type and topic (TPR) and the non-relevant pages (NRP) are as tabulated.

	Focused crawling		**Path-ascending crawling**		**The hybrid methods**	
Query type	TMF	TPR	TMF	TPR	TMF	TPR
1	**434**	**22**	**678**	**10**	**311**	**35**
2	**299**	**14**	**344**	**11**	**233**	**26**
3	**308**	**5**	**563**	**13**	**214**	**57**
4	**400**	**17**	**623**	**44**	**321**	**48**
5	**311**	**4**	**345**	**7**	**291**	**44**
6	**300**	**5**	**476**	**16**	**266**	**28**
7	**298**	**12**	**449**	**21**	**201**	**35**
8	**379**	**6**	**512**	**15**	**227**	**36**
9	**403**	**10**	**570**	**30**	**301**	**49**
10	**279**	**3**	**455**	**14**	**193**	**22**
TOTAL	**3411**	**98**	**5015**	**181**	**2558**	**380**

Table 1. Showing a comparative average results (in thousands) of ten categories of queries (10 queries in each category) administered to the four search engines.
TMF – Total match found
TPR – Total pages relevant (first 20 links)

The total number of matches per a query type in the table 1 above was subjected to a scattered plot. Figure 3 below is a clear indicator to the matches in terms of the number of documents that these crawling methods could extract for a given query type and topic. The number of matches on the y-axis and the query types (1 – 10) on the x-axis. While the path-ascending and the focused crawling methods tops in the number of matched document (increase in recall), the hybrid method system is however, less in recall. But in the actual collection of total pages relevant to the given query, the hybrid method tops as shown in figure 3; followed by the path-ascending before the focused crawling.

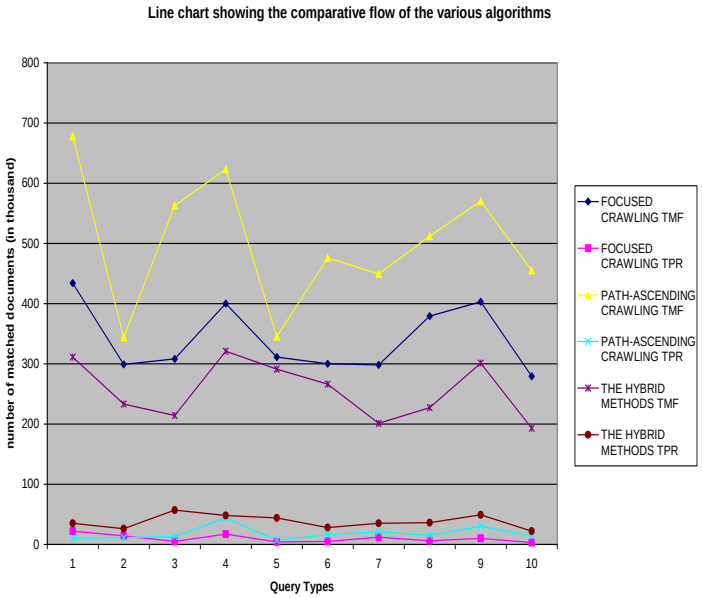

Fig. 3. A comparative matches for 10-typed queries.

However, comparatively the number of pages found relevant by weighting, ranking, and other measures, proves to be in advantage to our system. Here the total matches so found are less than other search engines and yet, the number of relevant documents so filtered is triple the other search engines as shown in table 2 below:

Table 2. Showing a comparative average results (in thousands) of ten categories of queries (10 queries in each category) administered to the three algorithms.

	TMF	**TPR**
FOCUSED CRAWLING	3411	98
PATH-ASCENDING CRAWLING	5015	181
THE HYBRID METHODS	2558	380

Translating table 2 into a stack bar as shown in figure 4 below. The Hybrid method's TPR triples those of focused and path-ascending crawling even though their relative recalls are very high.

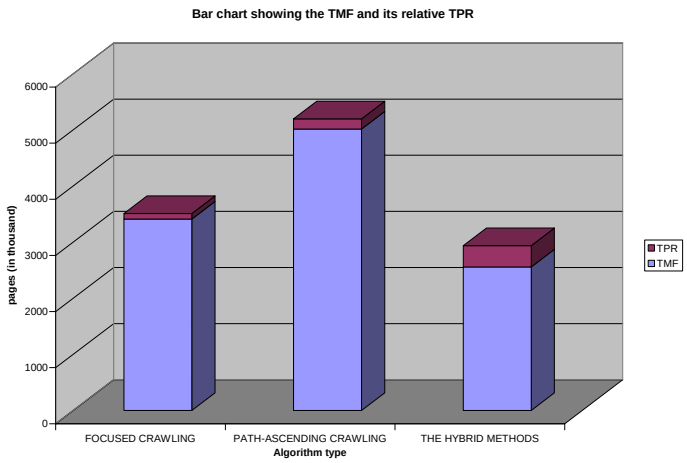

Fig.4.A comparative stack bar chart for Total Matches Found and Total Pages Relevant

A second evaluation is to find the Precision, recall and F-measure of the results given by:

$$\text{precision} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{retrieved documents}\}|}$$

$$\text{recall} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{relevant documents}\}|}$$

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Table 3 below shows the calculated recall, precision and F-measure of the gathered result. The calculation still keeps our system (the Hybrid method) at the top of the measures in the case of the listed factors. We made a clear interpretation of these by presenting table 3 in a pie as shown in figure 5, and a bar chart of the entire table content in figure 6.

Table 3. Showing the calculated recall, Precision and F-measure of table 1 above.

	FOCUSED CRAWLING		***PATH-ASCENDING CRAWLING***		***THE HYBRID METHODS***	
Precission	**0.03**	0.028731	**0.04**	0.036092	**0.15**	0.148554
Recall	**35.81**	34.80612	**28.01**	27.70718	**7.00**	6.731579
F-measures	**0.06**	0.057414	**0.07**	0.07209	**0.29**	0.290692

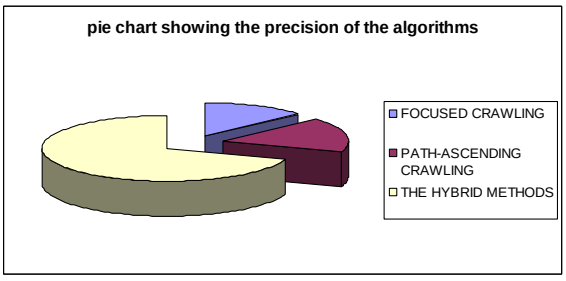

Fig. 5. A pie chart showing the precision of the Focused, Path-Ascending and Hybrid method crawling.

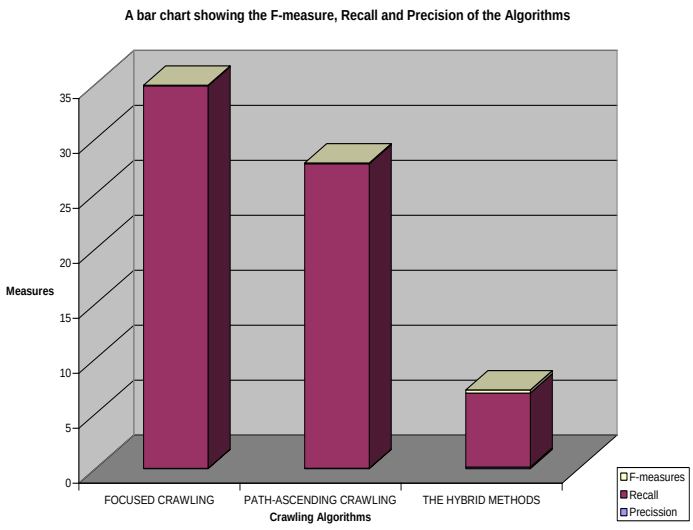

Fig. 6. A bar chart showing the F-measure, Recall and Precision of the Focused, Path-Ascending and Hybrid crawling methods.

CONCLUSION

Our main goal is to improve the quality of web search engines. Before now, most people believed that a complete search index would make it possible to find anything easily. According to Best of the Web 2000 -- Navigators, "The best navigation service should make it easy to find almost anything on the Web (once all the data is entered)." However, the Web of today is quite different. Anyone who has used a search engine recently can readily testify that the completeness of the index is not the only factor in the quality of search results. The merger of these two crawling techniques gives a clear perfect indication to a better collection. While the focused is interested in the number of relevant matches, the path-ascending method filters the hidden documents of the web that are not well linked to the site map for obvious reasons. But the Hybrid method here is interested in the precision.

REFERENCES

Cothey, V. (2004). Web-crawling reliability . *Journal of the American Society for Information Science and Technology* **55** (14). doi:10.1002/asi.20078 Web-crawling reliability].

Chakrabarti, S., van den Berg, M., and Dom, B. (1999). Focused crawling: a new approach to topic-specific web resource discovery. Computer Networks, 31(11–16):1623–1640.

Chakrabarti, S. (2000). Data mining for hypertext: a tutorial survey. SIGKDD Explorations Newsletter 1 (2): 1–11.

Chakrabarti, S. (2003). Mining the Web. Morgan Kaufmann Publishers. ISBN 1-55860-754-4

Diligenti, M., Coetzee, F., Lawrence, S., Giles, C. L., and Gori, M. (2000). Focused crawling using context graphs. In Proceedings of 26th International Conference on Very Large Databases (VLDB), Cairo, Egypt. : 527-534,

Menczer, F. and Belew, R.K. (1998). Adaptive Information Agents in Distributed Textual Environments. In K. Sycara and M. Wooldridge (eds.) Proc. 2nd Intl. Conf. on Autonomous Agents (Agents '98). ACM Press

Mitchell, T. M. (1997). Machine Learning. New York: McGraw-Hill.

Nelson, M. L. , Van de Sompel, H. , Liu, X., Harrison, T. L. and McFarland, N. (2005). "mod_oai: An Apache module for metadata harvesting". *In Proceedings of the 9th European Conference on Research and Advanced Technology for Digital Libraries (ECDL 2005)*: 509.

Pinkerton, B. (1994). Finding what people want: Experiences with the WebCrawler. In Proceedings of the First World Wide Web Conference, Geneva, Switzerland.