# Evaluation of a Graph-based Topical Crawler

**Aurel Cami**
School of EECS
University of Central Florida
Orlando, FL 32816, USA
acami@cs.ucf.edu

**Narsingh Deo**
School of EECS
University of Central Florida
Orlando, FL 32816, USA
deo@cs.ucf.edu

***Abstract –*** *Topical (or, focused) crawlers have become important tools in dealing with the massiveness and dynamic nature of the World Wide Web. Guided by a data mining component that monitors and analyzes the boundary of the set of crawled pages, a focused crawler selectively seeks out pages on a pre-defined topic. Recent research indicates that both the textual content of web pages and the structural information enclosed in the Web graph need to be exploited in order to build high quality focused crawlers. While, a variety of text-based and graph-based measures of similarity that can direct a focused crawler toward relevant pages have been developed, much remains to be done toward formally evaluating and ranking the effectiveness of various focused crawling algorithms. Inspired by a recent and comprehensive evaluation framework for focused crawlers, we analyze the performance of a graph-based algorithm and compare it with two other algorithms: a breadth-first one and a text-based, best-first one. The results suggest that our graph-based algorithm is faster and only slightly less effective than the text-based, best-first algorithm, while significantly outperforming the breadth-first one.*

**Keywords:** Focused crawling, graph-based, clustering coefficient.

## 1 Introduction

The World Wide Web has grown amazingly large: the most recent estimate puts the size of the surface Web (i.e., the indexable Web—the part that may theoretically be indexed by search engines) at more than 11.5 billion pages [14]. This massiveness of the Web coupled with its dynamic nature have posed unprecedented challenges for Web-related applications such as crawlers and search engines [9], [15]. One important challenge is increasing the *coverage* and maintaining the *currency* (or, freshness) of search engine indices. Indeed, it has been known for several years that search engines cover only a fraction of the indexable Web. For example, as of 2000, no search engine covered more than 16% of the indexable Web and the top 11 search engines combined covered about 50% of the indexable Web [17]. In a more recent study of the search engine coverage [14] it was estimated that, as of January 2005, Google covered 76.2% of the indexable Web, followed by Yahoo! and MSN with 69.3% and 61.9%, respectively. In addition, it has been reported that the web pages indexed by top search engines are often 3 to 4 months out-of-date [25].

A potential solution to this problem is the development of focused crawlers which selectively seek out pages relevant to a pre-defined topic. Since the size of the Web portion searched by a focused crawler is much smaller than the Web itself, it is hoped that both the coverage of a desired topic and the freshness of the pages in the repository may be significantly improved. In addition to these important applications of focused crawling, there are several others such as, automatic re-population of topic taxonomies (e.g., Yahoo!, and Open Dir) with newer and more relevant pages, Web filtering (e.g., identification of hate or pornographic websites), and assisting search engines in handling Web spamming.

At the heart of a focused crawler lies a data-mining component that monitors the boundary of the set of the crawled pages and guides the crawl toward relevant pages. This component has two main parts: (i) a *classifier* that evaluates the relevance of a web page with respect to the focus topic; and (ii) a *distiller* that identifies hypertext nodes that may lead to many relevant pages within few links.

The classifiers employed in the major existing focused-crawling applications may be categorized as: (a) text-based (or, lexical); (b) graph-based (or, link-based); and (c) hybrid. In a text-based classifier, the relevance of a page is determined by the similarity between the lexical content associated with that page and the text representation of the topic. In a graph-based classifier, the relevance of a page is determined only by the structural information enclosed in the graph formed by taking the web pages as nodes and the hyperlinks between them as edges. A hybrid classifier uses both the text and the link information to assess the relevance of a page. A distiller attempts to rank the pages considered as relevant by the classifier in terms of how likely they are to lead to other relevant pages. Common measures used in distilling such highly-relevant pages are the *PageRank* [22] and the *hub score* [16] of a page. It should be mentioned that some of the existing focused crawlers employ only a classifier, or do not separate their classifier from their distiller. In the remainder of this section we review the major focused crawling systems proposed in recent years.

Cho et al. [7] investigated the optimal order in which a crawler should visit the URLs it has seen, in order to obtain more “important” pages first. They defined several importance metrics, ordering schemes, and performance evaluation measures for this problem. They also experimentally evaluated these ordering schemes on the Stanford University Web and showed that a crawler with a good ordering scheme can obtain important pages significantly faster than one without.

Dean and Henzinger [8] proposed two algorithms that find relevant pages by using only the link information. To evaluate the effectiveness of these two algorithms, these authors performed a user study comparing their algorithms’ results with Netscape's "What's Related" service. They found that the precision of these two algorithms was, respectively, 73% and 51% better than that of Netscape, despite the fact that Netscape uses content and usage pattern information in addition to the link information.

Chakrabarti et al. [6] proposed a focused crawling system consisting of a text-based classifier and a distiller that favored the nodes with a high *hub* score. They reported on extensive focused-crawling experiments using several topics at different levels of specificity. In a subsequent paper [5] Chakrabarti et al. proposed a topic distillation technique based on the spectral properties of certain matrices derived from the web graph. This method, which extends the Kleinberg’s HITS algorithm [16], estimates the quality of a page using both the textual content of the page, and the context of the page: the pages it points to, the pages that point to it, and the web neighborhood in which it appears. They showed the results produced by this distiller for broad-topic queries on the Web, and also gave some anecdotal results applying the same techniques to US Supreme Court law cases, US patents, and a set of wall street journal newspaper articles.

Diligenti et al. [10] presented a focused-crawling algorithm that builds a model for the context within which topically relevant pages occur on the Web. This context model captures typical link hierarchies within which the relevant pages occur, as well as the content of documents that frequently co-occur with relevant pages. This algorithm leverages the existing capability of large search engines to provide partial reverse crawling results. The crawling results showed significant performance improvements in crawling efficiency over a breadth-first crawler.

Glover et al. [13] proposed a classifier that assessed the relevance of a page using its text content, text location and the HTML structure. They showed that a focused crawler based on this classifier was effective in locating personal homepages and calls for papers.

Pant and Menczer [23], [19] developed a distributed, multi-agent focused crawler, called MySpiders, designed to mine the Web online at *query time*. These authors discussed the benefits and shortcomings of using dynamic-search strategies versus the traditional static methods in which search and retrieval are disjoint. The performance of MySpiders system was evaluated by comparing its effectiveness in locating recent, relevant documents with that of search engines. The evaluation results suggest that augmenting search engines with an adaptive population of intelligent search agents could lead to a significant competitive advantage.

Liu et al. [18] investigated the use of probabilistic models, such as Hidden Markov Models (HMMs) and Conditional Random Fields (CRFs), in focused crawling. These models can potentially capture both the content information of web pages and their context, or link information. These authors compared the performance of a focused crawler based on a Hidden Markov Model (HMM) with one based on a best-first strategy. Furthermore, they discussed the use of Conditional Random Fields (CRF) to overcome some difficulties with HMMs and to support the use of many, arbitrary and overlapping features.

Qin et al. [24] argued that the focused crawlers that use local search algorithms to traverse the Web space, could be easily trapped within a limited sub-graph of the Web that surrounds the starting URLs and build domain-specific collections that are not comprehensive and diverse enough to scientists and researchers. To overcome this problem with local-search algorithms, they proposed a new approach that combines focused crawling with meta-search (i.e., the simultaneous querying of the major search engines). They conducted two user studies to evaluate the performance of this crawler. The results suggest that this approach could build domain-specific collections with higher quality than previous focused-crawling techniques.

In contrast with this flurry of activity on devising focused-crawling algorithms, the research on formally evaluating the effectiveness of topical crawlers is still in its infancy. Most notable has been the work of Menczer et al. [21], [20], who have conducted several focused-crawler evaluation studies. The latest version of their evaluation framework, which we refer to as the SMP framework, is described in length in [26] and is claimed by its authors to be the most comprehensive focused-crawling evaluation method to date. The SMP framework identifies a family of crawling-tasks relevant to applications of varying nature and difficulty. Then, it proposes a set of performance measures for fair comparative evaluations of crawlers along several dimensions, including generalized notions of *precision*, *recall*, and *efficiency*. This framework relies on independent relevance judgments compiled by human editors and available from public directories, such as the Open Dir Project (ODP), and synthesizes a number of methodologies in the focused-crawling literature.

Our current work was stimulated by the development of the SMP framework. Our main objective is to conduct a comparative evaluation of a generalized version of a graph-based, greedy, focused-crawling algorithm that we had proposed earlier [3]. Starting with a set of seed pages assumed to be on the same topic, our algorithm attempts to find an optimal subgraph that contains the seed nodes and has a high value of clustering coefficient. In [3] we showed that this algorithm is a fast and effective method for

discovering *communities* in some small, synthetic and real web-like networks, such as random graphs with given community structure and Zachary's karate-club network. The SMP framework allowed us to undertake a thorough experimental study to evaluate the effectiveness of this algorithm in guiding a focused crawler. We compared our algorithm with two others—a simple breadth-first algorithm, and a best-first, text-based algorithm [26]. The breadth-first algorithm was selected as a baseline for the cost analysis, while the best-first algorithm was selected as a baseline for the performance analysis. The results suggest that our graph-based algorithm is faster and only slightly less effective than the text-based best-first algorithm, while significantly outperforming the breadth-first crawling strategy.

# 2 The SMP Evaluation Framework

Before introducing the focused-crawling algorithms investigated in this study, we briefly describe the SMP framework. This framework has several dimensions: (i) the parameters that characterize a topic; (ii) the topic selection method; and (iii) the metrics for evaluating the cost and the performance of the crawler under investigation. We skip the details of the first dimension because it did not play a major role in our experiments. Below, we provide details about the two remaining dimensions of this framework which were crucial in our study. For a detailed discussion of the SMP framework the reader is referred to [26].

## 2.1 Topic Selection

The topic selection method proposed by the SMP framework relies on topic hierarchies compiled by human editors, such as the Open Directory Project (ODP) repository (dmoz.org). Such a hierarchy organizes the URLs in its repository in a tree-like hierarchy. The SMP framework identifies topics with subtrees of the ODP hierarchy. More specifically a topic at level TOPIC-LEVEL and depth MAX-DEPTH corresponds to a subtree of the ODP tree whose root is at distance TOPIC-LEVEL from the root of ODP tree and has depth MAX-DEPTH (Figure 1).

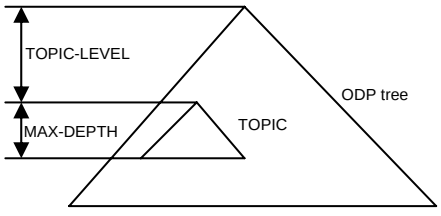



Figure 1. The topic selection in the SMP framework.

By varying the parameter TOPIC-LEVEL, one can change the *specificity* of a topic, while by varying the parameter MAX-DEPTH, one can produce *alternative descriptions* of a topic. Given a subtree of the ODP tree, the topic corresponding to this subtree is completely described by three lists: (i) a list of *keywords*; (ii) a list of target *URLs*; and (iii) a list of target *descriptions*. The list of keywords is obtained by concatenating all node labels from the root of the ODP tree down to and including the root of the topic subtree. The list of target URLs is obtained by concatenating all external links found in the nodes of the topic subtree. The list of target descriptions is obtained by concatenating the descriptions compiled by human editors for all URLs in the URL list of the topic.

The SMP framework proposes the following seed-selection method: Randomly choose a small subset $S_0$ of cardinality N-SEEDS from the set of target URLs of a topic. Then, using the back-linking service provided by most search engines (e.g., Google API, or Yahoo API), find a set $S_{DIST}$ of URLs such that, in the absence of broken links, there exists a path of length at most DIST from each node in $S_{DIST}$ to some node in $S_0$. In a practical implementation, the cardinality of $S_{DIST}$ may be smaller than the cardinality of $S_0$ due to certain restrictions imposed by search engines (e.g., the maximum number of queries allowed in a day). By increasing the value of parameter DIST from 0 to 1, 2, …, one can specify crawling tasks of increasing difficulty.

## 2.2 Evaluation Metrics

The SMP framework proposes five measures for evaluating focused crawlers. All these measures are time-dependent and allow for a temporal characterization of the behavior of a focused crawler. The first four measures characterize the effectiveness of a focused crawler in finding on-topic pages; the fifth measure is designed to conduct a performance-to-cost analysis. These measures are given below (using a notation a bit simpler than in [20]):

*1. URL-based Precision*

The URL-based precision $P_{URL}(t, D)$ of a focused crawler at time $t$ with respect to topic $D$ is defined as the fraction of URLs crawled up to time $t$ which belong to the set of target URLs of $D$, i.e.,

$$P_{URL}(t, D) = \frac{|C_t \cap T_D|}{|C_t|}. \quad (1)$$

Here $C_t$ denotes the set of URLs crawled up to time $t$ and $T_D$ denotes the set of target URLs of topic $D$.

*2. URL-based Recall*

The URL-based recall $R_{URL}(t, D)$ of a focused crawler at time $t$ with respect to topic $D$ is defined as the fraction of the target URLs crawled up to time $t$, i.e.,

$$R_{URL}(t, D) = \frac{|C_t \cap T_D|}{|T_D|}. \quad (2)$$

*3. Description-based Precision*

The description-based precision $P_{DES}(t, D)$ of a focused crawler at time $t$ with respect to topic $D$ is defined as the average cosine similarity between the Term-

Frequency-Inverse-Document-Frequency (TFIDF) vector of topic $D$ and the TFIDF vector of a crawled page $p \in C_t$, i.e.,

$$P_{DES}(t,D) = \frac{\sum_{p \in C_t} \sigma(p,D)}{|C_t|}. \quad (3)$$

Here $\sigma(p,T_D)$ denotes the cosine of the angle between the TFIDF vector of $p$ and the TFIDF vector of $D$.

*4. Description-based Recall*

The description-based recall $R_{DES}(t,D)$ of a focused crawler at time $t$ with respect to topic $D$ is defined as the sum over all pages $p \in C_t$ of cosine similarity between the TDIDF vector of $D$ and the TDIDF vector of $p$, i.e.,

$$R_{DES}(t,D) = \sum_{p \in C_t} \sigma(p,D). \quad (4)$$

*5. Recall/CPU time*

Either of the two previous recall measures ($R_{URL}$ or $R_{DES}$) may be used in the numerator of this ratio. The CPU time includes only the time spent by a crawler in deducing the relevance of pages with respect to the focus topic. The functions common to all crawlers, such as HTML parsing and page downloading, normally are not timed.

# 3 Focused Crawling Algorithms

## 3.1 Clustering-Coefficient Best First Search

The most basic version of our clustering-coefficient best first search (CC-BFS) algorithm was proposed in [3]. As the name suggests, this algorithm is based on the *clustering coefficient* graph parameter. The clustering coefficient $C(u)$ of a node $u$ is defined as the probability that two random neighbors of $u$ are neighbors themselves. Our motivation for using this graph-parameter to guide the crawl comes from the evidence that regions of Web that have a high clustering coefficient consist of pages on a common topic [11]. The CC-BFS algorithm aims to explore the extent to which it is possible to discover highly-clustered groups of nodes via a greedy search strategy.

This algorithm takes as input a set of seed pages $S$ assumed to be on the same topic and attempts to expand this set with additional pages on that topic. In each step, the algorithm takes into consideration the nodes at distance one (the set $N_1$) and the nodes at distance two (the set $N_2$) from the set of relevant nodes discovered up to that step. For each node $u \in N_1$, the algorithm computes the clustering coefficients $C_0(u), C_1(u), C_2(u)$ with respect to the graphs induced by the sets $S \cup \{u\}$, $N_1$, and $N_2 \cup \{u\}$, respectively. The decision made by this algorithm is essentially the following: If $C_0(u)$ is greater than $C_2(u)$, or if $C_1(u)$ is greater than $C_2(u)$, i.e., if the node $u$ is more clustered with the nodes of $S$ or $N_1$ than the nodes of $N_2$, then the node $u$ is placed in the set $S$ (i.e., it is considered to be "on topic"); Otherwise, it is considered irrelevant and thrown away. This process is repeated until a desired number of relevant pages has been visited.

## 3.2 Keyword-based Best First Search

The second algorithm used in our study is BFS256—an algorithm proposed by Menczer et al. [20]. This algorithm maintains a priority queue which holds the URLs waiting to be visited sorted by the similarity between the topic *keywords* and the text content of the page from which the URL was extracted (the *source* page). This similarity is computed as the cosine of the angle between the TDIDF vector representing the topic keywords and the TDIDF vector representing the source page of a URL. Each iteration of the algorithm selects for processing a batch of the top 256 URLs. The URLs of the web pages whose similarity with the topic is low are eliminated from the priority queue to make room for newly discovered better pages.

In [20] it was shown that BFS256 is a very competitive algorithm in terms of precision and recall. Therefore, we chose this algorithm as a baseline for evaluating the precision and recall of CC-BFS.

## 3.3 Breadth First Search

A basic Breadth-First algorithm visits the web pages in the order they are encountered. Because the page to be visited in each step is immediately available (the top page of a FIFO queue) this algorithm has a minimal cost. Hence, the performance-to-cost ratio of this algorithm is high, especially in the initial steps of crawl and thus we chose to use it as a baseline for evaluating the performance-to-cost ratio of CC-BFS.

# 4 Implementation

We implemented the SMP framework and the three focused crawlers discussed in Section 3 on top of UcfBot—a general purpose crawler we had developed earlier. Below we provide some details of our implementation.

## 4.1 UcfBot Crawler Architecture

UcfBot crawler was designed and implemented during a project aimed at collecting massive and suitable data sets to perform statistical analysis of the Web. UcfBot is written in C++ and runs in a Linux environment. It is: (i) a *scalable crawler*, i.e., its speed and performance remain essentially unaffected as the number of crawled pages increases; (ii) a *polite crawler*, i.e., it does not overwhelm the web servers with requests and avoids crawling certain URLs which are not supposed to be indexed; (iii) an *extensible crawler* which can be easily extended to perform other functions apart from the ones it was originally designed for. The architecture and functionality of UcfBot are discussed in detail in [2].

### 4.2 UCfBot Focused-crawling API

The **U**cf**B**ot **F**ocused-crawling (UBF) API is an interface built on top of the UcfBot crawler which allows the rapid implementation and evaluation of focused-crawling algorithms. This interface provides functions for performing the tasks that are common to all focused crawlers, such as downloading, parsing, etc. To implement a particular focused crawling strategy, the user needs to implement only three functions: (a) `setSeedUrls()` which initializes the set of starting URLs for a particular crawl; (b) `getUrlObjectPriority()` which computes the relevance of a URL with respect to the focus topic; and (c) `doOneCrawlingStep()` which implements one step of the focused crawler (i.e., the processing of the top URL or a batch of top $k$ URLs from the priority queue).

The UBF API also implements the SMP framework. The standard pre-processing of the content of the crawled Web pages ("stop-word" removal, stemming, etc.,) is hidden from the user. All evaluation measures specified by SMP framework may be computed by calling simple functions provided by UBF. The details of how to use the UBF API to implement and evaluate a focused crawler are given in [4].

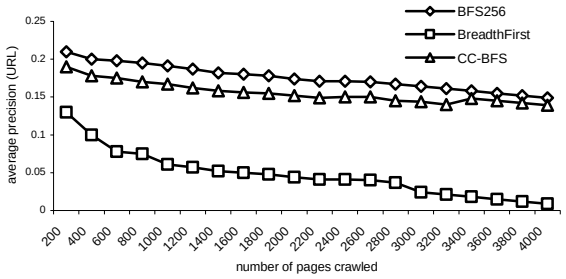
To select the sample topics for our experiments we used the most recent RDF dump of ODP (the file "structure.rdf.u8.gz" available at http://rdf.dmoz.org/rdf/) and the Perl script provided by Menczer et al. (available at http://www.informatics.indiana.edu/fil/IS/Framework/).

With TOPIC-LEVEL = MAX-DEPTH = 2 and DIST = 3, we extracted 100 topics (targets plus seeds) from this RDF dump. Starting from the seeds of each of the 100 topics, each of the three crawlers was run until it collected 4000 pages. In implementing the CC-BFS algorithm we added dummy edges between the seed nodes to force the subgraph induced by the seeds to be a clique. Each of the five metrics discussed in Section 2.2 was averaged over the 100 topics. The results for the three crawlers are shown next.

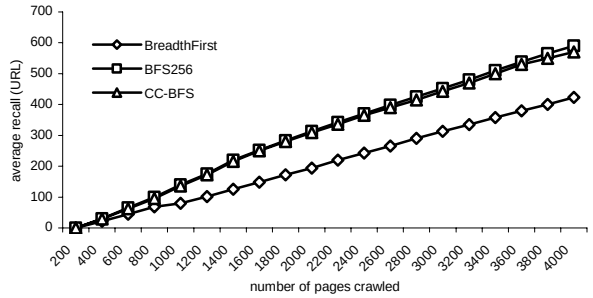
## 5 Experimental Results

Figure 1 shows two plots that give a temporal description of the relative performance of the three focused crawlers described in Section 3. Figure 1(a) shows the average URL-based precision versus the number of crawled pages. As seen, BFS256 is the best performing algorithm, followed by the CC-BFS and Breadth-First. Figure 1(b) shows the average URL-based recall versus the number of crawled pages. The BFS256 crawler performs only slightly better than the CC-BFS; both of these crawlers significantly outperform the Breadth-First crawler.

Results similar to those shown in Figure 1 were obtained for the URL-based precision and recall. Figure 2 depicts the average recall/CPU time versus the number of crawled pages. This plot shows that initially the Breadth-First crawler performs better than the other two (as expected, due to the low cost of this algorithm). However, as the number of crawled pages approaches 1000, the CC-BFS catches up and from that point on it outperforms Breadth-First and BFS256 crawlers.




(a)




(b)

Figure 1. Performance of the Breadth-First, BFS256 and CC-BFS crawlers vs. the number of crawled pages: (a) average URL-precision; (b) average URL-recall.

The explanation for this observation is that the cost of CC-BFS is lower than that of BFS256 and its effectiveness is better than that of Breadth-First.

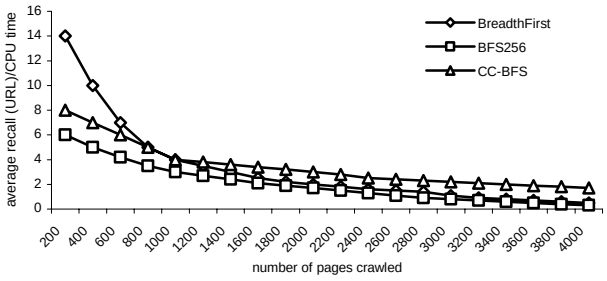



Figure 2. The average performance-to-cost ratio (recall/CPU time) of the three focused crawlers vs. the number of crawled pages.

# 6 Discussion and Future Work

Our study shows that the simple algorithm CC-BFS which attempts to find subgraphs with high values of clustering coefficient in a greedy fashion is quite effective in finding web pages on a given topic. Numerous new research directions remain the focus of our research.

First, it would be interesting to use a more general version of the CC-BFS algorithm, CC-BFS-*k*, which employs the *generalized clustering coefficient* $C^{(k)}(u)$ instead of clustering coefficient $C(u)$. The computation of $C^{(k)}(u)$ takes into consideration not only the neighbors of node $u$ but also the nodes at distance $2,3,\ldots,k$ from $u$. More specifically, $C^{(k)}(u)$ is given by:

$$C^{(k)}(u) = \sum_{i=1}^{k} w_i f_i ,$$

where $f_i$ is the fraction of pairs of distance-$i$ neighbors of $u$ which are neighbors themselves, and $w_i$ is the weight associated with distance $i$. The question that needs to be investigated is whether the additional cost imposed on the CC-BFS algorithm by using $C^{(k)}(u),\ k>1,$ would be justified by a significant increase its accuracy.

Furthermore, it would be desirable to compare the effectiveness of clustering coefficient in guiding a greedy focused crawler with that of other parameters proposed in recent literature, such as alliance coefficient (the fraction of the number of neighbors of a node known to be on-topic to the number of remaining neighbors). The goal of this research would be to rank various relevant graph parameters in terms of their effectiveness in capturing the notion of a topical community of nodes.

Another important question is to understand the similarities and differences between the text-based and graph-based focused crawlers by analyzing the amount of overlap between the sets of pages found by each of these strategies. This analysis would shed some light on the problem of how to optimally combine textual and graph information in guiding a focused crawler.

# 4 References


[1] K. Bharat and M. R. Henzinger, "Improved algorithms for topic distillation in a hyperlinked environment," In Proceedings of *21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '98)*, pp. 104-111, 1998.

[2] H. Balakrishnan, A. Cami, and N. Deo, "UcfBot-A high-performance Web Crawler," Technical Report, University of Central Florida, School of Computer Science, CS-TR-05-08, 2005.

[3] A. Cami and N. Deo, "A greedy community-mining algorithm based on clustering coefficient," *Congressus Numerantium*, vol. 172, pp. 161-176, 2005.

[4] A. Cami, H. Balakrishnan, and N. Deo. The UBF API for Developing and Evaluating Focused Crawlers. Technical Report, University of Central Florida, School of Computer Science, 2006.

[5] S. Chakrabarti, B. E. Dom, D. Gibson, R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins, "Topic distillation and spectral filtering," *Artificial Intelligence Review*, vol. 13(5-6), pp. 409-435, 1999.

[6] S. Chakrabarti, M. van den Berg, and B. Dom, "Focused crawling: a new approach to topic-specific Web resource discovery," *Computer Networks*, vol. 31(11-16), pp. 1623-1640, 1999.

[7] J. Cho, H. Garcia-Molina, and L. Page, "Efficient crawling through URL ordering," *Computer Networks and ISDN Systems*, vol. 30(1-7), pp. 161-172, 1998.

[8] J. Dean and M. R. Henzinger, "Finding related pages in the World Wide Web," In Proceedings of *8th International Conference on World Wide Web (WWW '99)*, pp. 1467-1479, 1999.

[9] N. Deo and P. Gupta, "Graph-theoretic analysis of the World Wide Web: new directions and challenges," *Mat. Contemp.*, vol. 25, pp. 49-69, 2003.

[10] M. Diligenti, F. Coetzee, S. Lawrence, C. L. Giles, and M. Gori, "Focused Crawling Using Context Graphs," In Proceedings of *26th International Conference on Very Large Data Bases (VLDB '00)*, pp. 527-534, 2000.

[11] J.-P. Eckmann, and E. Moses. “Curvature of co-links uncovers hidden thematic layers in the World Wide Web.” In Proceedings of *the National Academy of Sciences of the USA*, vol. 99, pp. 5825-5829, 2002.

[12] D. Gibson, J. Kleinberg, and P. Raghavan, "Inferring Web communities from link topology," In Proceedings of *9th ACM Conference on Hypertext and Hypermedia (HYPERTEXT '98)*, pp. 225-234, 1998.

[13] E. J. Glover, G. W. Flake, S. Lawrence, W. P. Birmingham, A. Kruger, C. L. Giles, and D. M. Pennock, "Improving category specific Web search by learning query modifications," In Proceedings of *2001 Symposium on Applications and the Internet*, pp. 23-32, 2001.

[14] A. Gulli, and A. Signorini. “The indexable Web is more than 11.5 billion pages.” In Proceedings of *WWW 2005*, May 10-14, Chiba, Japan, 2005.

[15] M. R. Henzinger, "Algorithmic challenges in web search engines," *Internet Math.*, vol. 1(1), pp. 115-123, 2003.

[16] J. M. Kleinberg, "Authoritative sources in a hyperlinked environment," *J. ACM*, vol. 46(5), pp. 604-632, 1999.

[17] S. Lawrence and C. L. Giles, "Accessibility of information on the web," *Intelligence*, vol. 11(1), pp. 32-39, 2000.

[18] H. Liu, E. Milios, and J. Janssen, "Probabilistic models for focused web crawling," In Proceedings of *6th annual ACM International Workshop on Web Information and Data Management (WIDM '04)*, pp. 16-22, 2004.

[19] F. Menczer, "Complementing search engines with online web mining agents," *Decision Support Systems*, vol. 35(2), pp. 195-212, 2003.

[20] F. Menczer, G. Pant, and P. Srinivasan, "Topical web crawlers: Evaluating adaptive algorithms," *ACM Trans. Inter. Tech.*, vol. 4(4), pp. 378-419, 2004.

[21] F. Menczer, G. Pant, P. Srinivasan, and M. E. Ruiz, "Evaluating topic-driven Web crawlers," *SIGIR Forum, Spec. Issue*, pp. 241-249, 2001.

[22] L. Page, S. Brin, R. Motwani, and T. Winograd, "The Pagerank citation ranking: Bringing order to the web," Stanford Digital Library Technologies Project, Technical Report 1998.

[23] G. Pant and F. Menczer, "MySpiders: evolve your own intelligent Web crawlers," *Autonomous Agents and Multi-Agent Systems*, vol. 5(2), pp. 221-229, 2002.

[24] J. Qin, Y. Zhou, and M. Chau, "Building domain-specific web collections for scientific digital libraries: a meta-search enhanced focused crawling method," In Proceedings of *JCDL '04: Proceedings of the 4th ACM/IEEE-CS joint conference on Digital libraries*, pp. 135-141, 2004.

[25] Search Engine Watch: http://searchenginewatch.com.

[26] P. Srinivasan, F. Menczer, and G. Pant, "A general evaluation framework for topical crawlers," *Information Retrieval*, vol. 8(3), pp. 417-447, 2005.