Exploration versus Exploitation in Topic Driven Crawlers

Gautam Pant[@] Padmini Srinivasan[@!]

Filippo Menczer[@]

[@]Department of Management Sciences
[!]School of Library and Information Science
The University of Iowa
Iowa City, IA 52242
{gautam-pant,padmini-srinivasan,filippo-menczer}@uiowa.edu

Abstract

The dynamic nature of the Web highlights the scalability limitations of universal search engines. Topic driven crawlers can address the problem by distributing the crawling process across users, queries, or even client computers. The context available to a topic driven crawler allows for informed decisions about how to prioritize the links to be visited. Here we focus on the balance between a crawler's need to exploit this information to focus on the most promising links, and the need to explore links that appear suboptimal but might lead to more relevant pages. We investigate the issue for two different tasks: (i) seeking new relevant pages starting from a known relevant subset, and (ii) seeking relevant pages starting a few links away from the relevant subset. Using a framework and a number of quality metrics developed to evaluate topic driven crawling algorithms in a fair way, we find that a mix of exploitation and exploration is essential for both tasks, in spite of a penalty in the early stage of the crawl.

1 Introduction

A recent projection estimates the size of the visible Web today (March 2002) to be around 7 billion "static" pages [10]. The largest search engine, Google, claims to be "searching" about 2 billion pages. The fraction of the Web covered by search engines has not improved much over the past few years [16]. Even with increasing hardware and bandwidth resources at their disposal, search engines cannot keep up with the growth of the Web and with its rate of change [5].

These scalability limitations stem from search engines' attempt to crawl the whole Web, and to answer any query from any user. Decentralizing the crawling process is a more scalable approach, and bears the additional benefit that crawlers can be driven by a rich context (topics, queries, user profiles) within which to interpret pages and select the links to be visited. It comes as no surprise, therefore, that the development of topic driven crawler algorithms has received significant attention in recent years [9, 14, 8, 18, 1, 19].

Topic driven crawlers (also known as focused crawlers) respond to the particular information needs expressed by topical queries or interest profiles. These could be the needs of an individual user (query time or online crawlers) or those of a community with shared interests (topical search engines and portals).

Evaluation of topic driven crawlers is difficult due to the lack of known relevant sets for Web searches, to the presence of many conflicting page quality measures, and to the need for fair gauging of crawlers' time and space algorithmic complexity. In recent research we presented an evaluation framework designed to support the comparison of topic driven crawler algorithms

under specified resource constraints [19]. In this paper we further this line of research by investigating the relative merits of exploration versus exploitation as a defining characteristic of the crawling mechanism.

The issue of exploitation versus exploration is a universal one in machine learning and artificial intelligence, since it presents itself in any task where search is guided by quality estimations. Under some regularity assumption, one can assume that a measure of quality at one point in the search space provides some information on the quality of nearby points. A greedy algorithm can then exploit this information by concentrating the search in the vicinity of the most promising points. However, this strategy can lead to missing other equally good or even better points, for two reasons: first, the estimates may be noisy; and second, the search space may have local optima that trap the algorithm and keep it from locating global optima. In other words, it may be necessary to visit some "bad" points in order to arrive at the best ones. At the other extreme, algorithms that completely disregard quality estimates and continue to explore in a uniform or random fashion do not risk getting stuck at local optima, but they do not use the available information to bias the search and thus may spend most of their time exploring suboptimal areas. A balance between exploitation and exploration of clues is obviously called for in heuristic search algorithms, but the optimal compromise point is unknown unless the topology of the search space is well understood — which is typically not the case.

Topic driven crawlers fit into this picture very well if one views the Web as the search space, with pages as points and neighborhoods as defined by hyperlinks. A crawler must decide which pages to visit based on the cues provided by links from nearby pages. If one assumes that a relevant page has a higher probability to be near other relevant pages than to any random page, then quality estimate of pages provide cues that can be exploited to bias the search process. However, given the short range of relevance clues on the Web [17], a very relevant page might be only a few links behind an apparently irrelevant one.

Balancing the exploitation of quality estimate information with exploration of suboptimal pages is thus crucial for the performance of topic driven crawlers. It is a question that we study empirically with respect to two different tasks. In the first, we seek relevant pages starting from a set of relevant links. Applications of such a task are query-time search agents that use results of a search engine as starting points to provide a user with recent and personalized results. Since we start from relevant links, we may expect an exploratory crawler to perform reasonably well. The second task involves seeking relevant pages while starting the crawl from links that are a few links away from a relevant subset. Such a task may be a part of Web mining or competitive intelligence applications (e.g., a search starting from competitors' home pages). If we do not start from a known relevant subset, the appropriate balance of exploration vs. exploitation becomes an empirical question.

2 Evaluation Framework

2.1 Topics, Examples and Neighbors

In order to evaluate crawler algorithms, we need *topics,* some corresponding relevant *examples,* and *neighbors*. The neighbors are URLs extracted from neighborhood of the examples. We obtain our topics from the Open Directory (DMOZ). We ran randomized Breadth-First crawls starting from each of the main categories on the DMOZ site.[1] The crawlers identify DMOZ "leaves," i.e., pages that have no children category nodes. Leaves with five or more external links are then used to derive topics. We thus collected 100 topics.

A topic is represented by three types of information derived from the corresponding leaf page. First, the words in the DMOZ hierarchy form the topic's *keywords*. Second, up to 10 external links form the topic's examples. Third, we concatenate the text descriptions and anchor text of the target URLs (written by DMOZ human editors) to form a topic *description*. The difference be-

[1] http://dmoz.org

Table 1: A sample topic. The description is truncated for space limitations.

Topic Keywords	Topic Description	Examples

`Recreation`
`Hot Air Ballooning`
`Organizations`
	Aerostat Society of Australia Varied collection of photos and facts about ballooning in Australia, Airships, Parachutes, Balloon Building and more. Includes an article on the Theory of Flight. Albuquerque Aerostat Ascension Association A comprehensive site covering a range of ballooning topics including the Albuqeurque Balloon Fiesta, local education and safety programs, flying events, club activities and committees, and club history. Arizona Hot Air Balloon Club [...]	`http://www.ozemail.com.au/~p0gwil`
		`http://www.hotairballooning.org/`
		`http://www.balllooningaz.com`
		`http://www.aristotle.net/~mikev/`
		`http://www89.pair.com/techinfo/ABAC/abac.htm`
		`http://www.prairienet.org/bagi`
		`http://www.atu.com.au/~balloon/club1.html`
		`http://communities.msn.com/BalloonCrews`
		`http://www.bfa.net`
		`http://www.ask.ne.jp/~kanako/ebst.html`

tween topic keywords and topic descriptions is that we give the former to the crawlers, as models of (short) query-like topics, while we use the latter, which are much more detailed representations of the topics, to gauge the relevance of the crawled pages in our post-hoc analysis. Table 1 shows a sample topic.

The neighbors are obtained for each topic through the following process. For each of the examples, we obtain the top 20 inlinks as returned by Google.[2] Next, we get the top 20 inlinks for each of the inlinks obtained earlier. Hence, if we had 10 examples to start with, we may now have a maximum of 4000 unique URLs. A subset of 10 URLs is then picked at random from this set. The links in such a subset are called the neighbors.

2.2 Architecture

We use the a previously proposed evaluation framework to compare different crawlers [19]. The framework allows one to easily plug in modules implementing arbitrary crawling algorithms, which share data structures and utilities to optimize efficiency without affecting the fairness of the evaluation.

As mentioned before, we use the crawlers for two different tasks. For the first task, the crawlers start from the examples while for the second the starting points are the neighbors. In either case, as the pages are fetched their component URLs are added to a list that we call the *frontier*. A crawler may use topic's keywords to guide the selection of frontier URLs that are to be fetched at each iteration. For a given topic, a crawler is allowed to crawl up to `MAX_PAGES` = 2000 pages. However, a crawl may end sooner if the crawler's frontier should become empty. We use a timeout of 10 seconds for Web downloads. Large pages are chopped so that we retrieve only the first 100 KB. The only protocol allowed is HTTP (with redirection allowed), and we also filter out all but "static pages" with `text/html` content. Stale links yielding HTTP error codes are removed as they are found (only good links are used in the analysis).

We constrain the space resources a crawler algorithm can use by restricting the frontier size to `MAX_BUFFER` = 256 URLs. If the buffer becomes full then the crawler must decide which links are to be replaced as new links are added.

3 Crawling Algorithms

In this paper we study the notion of exploration versus exploitation. We begin with a single family of crawler algorithms with a single greediness parameter to control the exploration/exploitation behavior. In our previous experiments [19] we found that a naive Best-First crawler displayed the best performance among three crawlers considered. Hence, in this study we explore variants of the Best-First crawler. More generally, we examine the *Best-N-First* family of crawlers where the parameter N controls the characteristic of interest.

Best-First crawlers have been studied before [9, 14]. The basic idea is that given a frontier of

[2] http://google.yahoo.com

```
Best_N_First(topic, starting_urls, N) {
   foreach link (starting_urls) {
      enqueue(frontier, link);
   }
   while (#frontier > 0 and visited < MAX_PAGES) {
      links_to_crawl := dequeue_top_links(frontier, N);
      foreach link (randomize(links_to_crawl)) {
         doc := fetch_new_document(link);
         score :=  sim(topic, doc);
         foreach outlink (extract_links(doc)) {
            if (#frontier >= MAX_BUFFER) {
                dequeue_link_with_min_score(frontier);
            }
            enqueue(frontier, outlink, score);
         }
      }
   }
}
```

Figure 1: Pseudocode of Best-N-First crawlers.

links, the best link according to some estimation criterion is selected for crawling.

Best-N-First is a generalization in that at each iteration a batch of top N links to crawl are selected. After completing the crawl of N pages the crawler decides on the next batch of N and so on. As mentioned above, the topic's keywords are used to guide the crawl. More specifically this is done in the link selection process by computing the lexical similarity between a topic's keywords and the source page for the link. Thus the similarity between a page p and the topic is used to estimate the relevance of the pages linked from p. The N URLs with the best estimates are then selected for crawling. Cosine similarity is used by the crawlers and the links with minimum similarity score are removed from the frontier if necessary in order to not exceed the `MAX_BUFFER` size. Figure 1 offers a simplified pseudocode of a Best-N-First crawler.

Best-N-First offers an ideal context for our study. The parameter N controls the greedy behavior of the crawler. Increasing N results in crawlers with greater emphasis on exploration and consequently a reduced emphasis on exploitation. Decreasing N reverses this; selecting a smaller set of links is more exploitative of the evidence available regarding the potential merits of the links. In our experiments we test five mutants of the crawler by setting N to 1, 16, 64, 128 and 256. We refer to them as BFSN where N is one of the above values.

4 Evaluation Methods

Table 2 depicts our overall methodology for crawler evaluation. The two rows of Table 2 indicate two different methods for gauging page quality. The first is a purely lexical approach wherein similarity to the topic description is used to assess relevance. The second method is primarily linkage based and is an approximation of the retrieval/ranking method used by Google [6]; it uses PageRank to discriminate between pages containing the same number of topic keywords.

The columns of the table show that our measures are used both from a static and a dynamic perspective. The static approach examines crawl quality assessed from the full set of (up to 2000) pages crawled for each query. In contrast the dynamic measures provide a temporal characterization of the crawl strategy, by considering the pages fetched while the crawl is in progress. More specifically, the static approach measures coverage, i.e., the ability to retrieve "good" pages where the quality of a page is assessed in two different ways (corresponding to the rows of the table). Our static plots show the ability of each crawler to retrieve more or fewer highly relevant pages. This is analogous to plotting recall as a function of generality.

The dynamic approach examines the quality of retrieval as the crawl progresses. Dynamic plots offer a trajectory over time that displays the dynamic behavior of the crawl. The measures are built on average (quality-based) ranks and are generally inversely related to precision. As the average rank decreases, an increasing proportion of the crawled set can be expected to be relevant.

It should be noted that scores and ranks used in each dynamic measure are computed omnisciently, i.e., all calculations for each point in time for a crawler are done using data generated from the full crawl. For instance, all PageRank scores are calculated using the full set of retrieved pages. This strategy is quite reasonable given that we want to use the best possible evidence when judging page quality.

Table 2: Evaluation Schemes and Measures. The static scheme is based on coverage of top pages (ranked by quality metric among all crawled pages, S). $S_{crawler}$ is the set of pages visited by a crawler. The dynamic scheme is based on the ranks (by quality metric among all crawled pages, S) averaged over the crawl sets at time t, $S_{crawler}(t)$.

	Static Scheme	Dynamic Scheme
Lexical	$\|S_{crawler} \cap top_{rank_{SMART}}(S)\|$	$\sum_{p \in S_{crawler}(t)} rank_{SMART}(p)/\|S_{crawler}(t)\|$
Linkage	$\|S_{crawler} \cap top_{rank_{KW,PR}}(S)\|$	$\sum_{p \in S_{crawler}(t)} rank_{KW,PR}(p)/\|S_{crawler}(t)\|$

4.1 Lexical Based Page Quality

We use the SMART system [23] to rank the retrieved pages by their lexical similarity to the topic. The SMART system allows us to pool all the pages crawled by all the crawlers for a topic and then rank these against the topic description. The system utilizes term weighting strategies involving term frequency and inverse document frequency computed from the pooled pages for a topic. SMART computes the similarity between the query and the topic as a dot product of the topic and page vectors. It outputs a ranked set of pages based on their topic similarity scores. That is, for each page we get a rank which we refer to as $rank_{SMART}$ (cf. Table 2). Thus given a topic, the percentage of top n pages ranked by SMART (where n varies) that are retrieved by each crawler may be calculated, yielding the static evaluation metric.

For the dynamic view we use the $rank_{SMART}$ values for pages to calculate mean $rank_{SMART}$ at different points of the crawl. If we let $S_{crawler}(t)$ denote the set of pages retrieved up to time t, then we calculate mean $rank_{SMART}$ over $S_{crawler}(t)$. The set $S_{crawler}(t)$ of pages increases in size as we proceed in time. We approximate t by the number of pages crawled. The trajectory of mean $rank_{SMART}$ values over time displays the dynamic behavior of a crawler.

4.2 Linkage Based Page Quality

It has been observed that content alone does not give a fair measure of the quality of the page [15]. Algorithms such as HITS [15] and PageRank [6] use the linkage structure of the Web to rank pages. PageRank in particular estimates the *global* popularity of a page. The computation of PageRanks can be done through an iterative process. PageRanks are calculated once after all the crawls are completed. That is, we pool the pages crawled for all the topics by all the crawlers and then calculate the PageRanks according to the algorithm described in [13]. We sort the pages crawled for a given topic, by all crawlers, first based on the number of topic keywords they contain and then sort the pages with same number of keywords by their PageRank. The process gives us a $rank_{KW,PR}$ for each page crawled for a topic.

Once again, our static evaluation metric measures the percentage of top n pages (ranked by $rank_{KW,PR}$) crawled by a crawler on a topic. In the dynamic metric, mean $rank_{KW,PR}$ is plotted over each $S_{crawler}(t)$ where t is the number of pages crawled.

5 Results

For each of the evaluation schemes and metrics outlined in Table 2, we analyzed the performance of each crawler on the two tasks.

5.1 Task 1 : Starting from Examples

For the first task the crawlers start from a relevant subset of links, the examples, and use the hyperlinks to navigate and discover more relevant pages. The results for the task are summarized by the plots in Figure 2. For readability, we are only plotting the performance of a selected subset of the Best-N-First crawlers ($N = 1, 256$). The behavior of the remaining

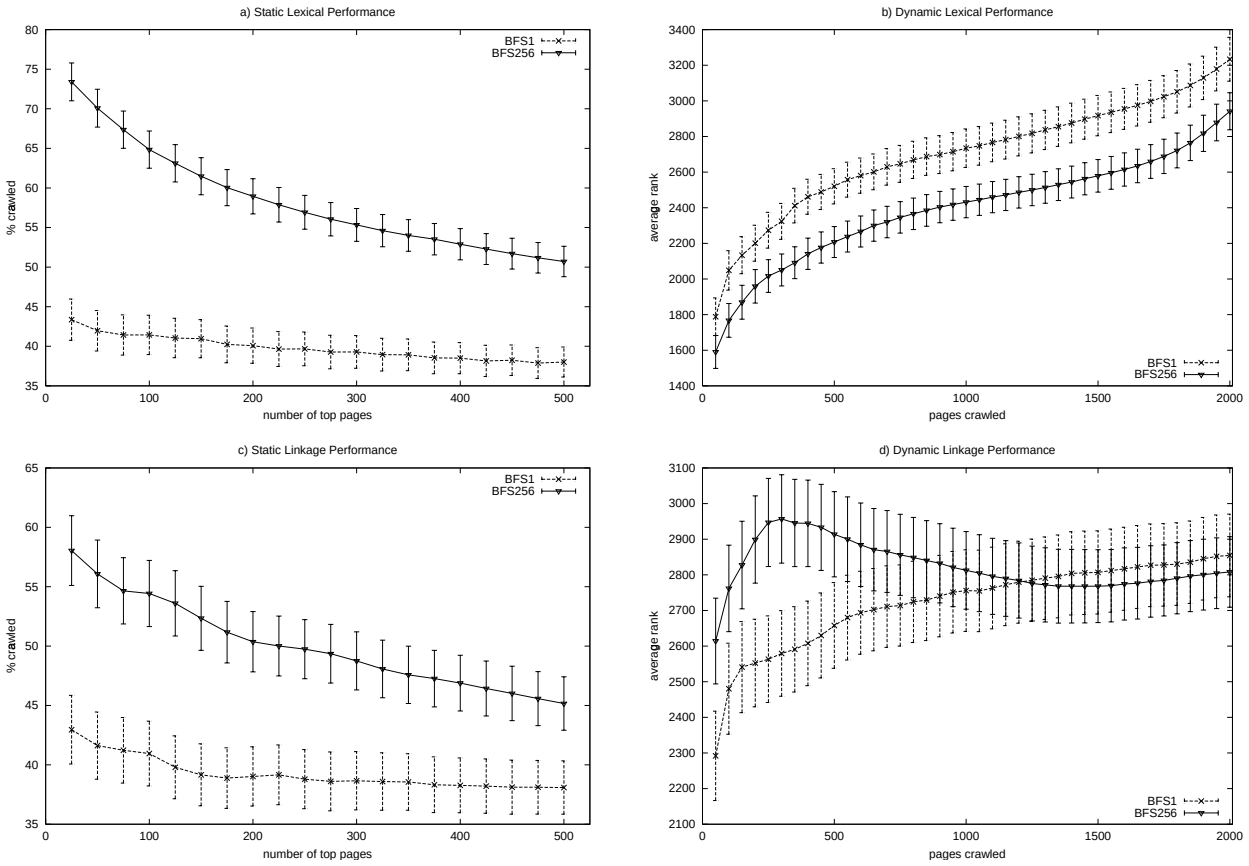

Figure 2: Static evaluation (left) and dynamic evaluation (right) of representative crawlers on Task 1. The plots correspond to lexical (top) and linkage (bottom) quality metric. Error bars correspond to ± 1 standard error across the 100 topics in this and the following plots.

crawlers (BFS16, BFS64 and BFS128) can be extrapolated between the curves corresponding to BFS1 and BFS256.

The most general observation we can draw from the plots is that BFS256 achieves a significantly better performance under the static evaluation schemes, i.e., a superior coverage of the most highly relevant pages based on both quality metrics and across different numbers of top pages (cf. Figure 2a,c). The difference between the coverage by crawlers for different N increases as one considers fewer highly relevant pages. These results indicate that exploration is important to locate the highly relevant pages when starting from relevant links, whereas too much exploitation is harmful.

The dynamic plots give us a richer picture. (Recall that here lowest average rank is best.) BFS256 still does significantly better than other crawlers on the lexical metric (cf. Figure 2b). However, the linkage metric shows that BFS256 pays a large penalty in the early stage of the crawl (cf. Figure 2d). However, the crawler regains quality over the longer run. The better coverage of highly relevant pages by this crawler (cf. Figure 2c) may help us interpret the improvement observed in the second phase of the crawl. We conjecture that by exploring suboptimal links early on, BFS256 is capable of eventually discovering paths to highly relevant pages that escape more greedy strategies.

5.2 Task 2: Starting from Neighbors

The success of a more exploratory algorithm on the first task may not come as a surprise since we start from known relevant pages. However, in

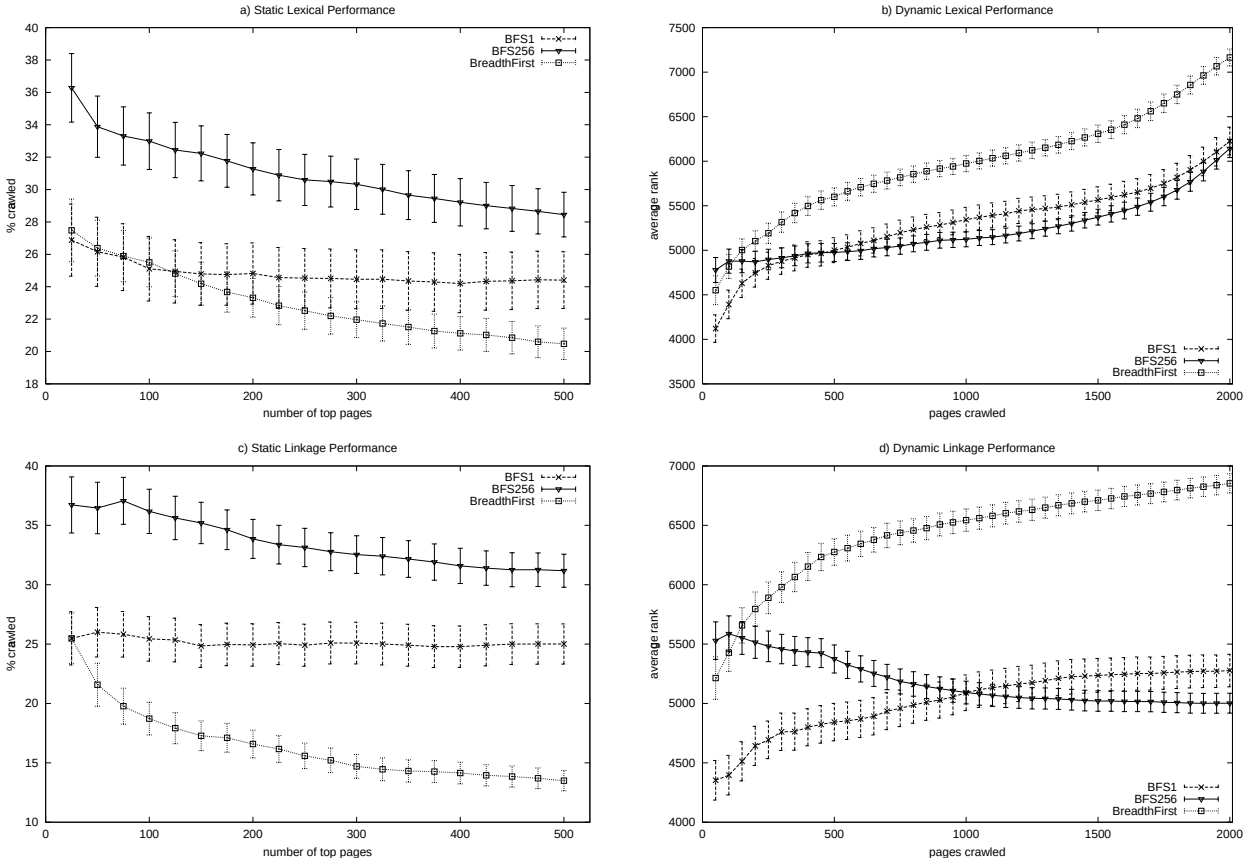

Figure 3: Static evaluation (left) and dynamic evaluation (right) of representative crawlers on Task 2. The plots correspond to lexical (top) and linkage (bottom) quality metric.

the second task we use the links obtained from the neighborhood of relevant subset as the starting points with the goal of finding more relevant pages. We take the worst (BFS1) and the best (BFS256) crawlers on Task 1, and use them for Task 2. In addition, we add a simple Breadth-First crawler that uses the limited size frontier as a FIFO queue. The Breadth-First crawler is added to observe the performance of a blind exploratory algorithm. A summary of the results is shown through plots in Figure 3. As for Task 1, we find that the more exploratory algorithm, BFS256, performs significantly better than BFS1 under static evaluations for both lexical and linkage quality metrics (cf. Figure 3a,c). In the dynamic plots (cf. Figure 3b,d) BFS256 seems to bear an initial penalty for exploration but recovers in the long run. The Breadth-First crawler performs poorly on all evaluations. Hence, as a general result we find that exploration helps an exploitative algorithm, but exploration without guidance goes astray.

Due to the availability of relevant subsets (examples) for each of the topics in the current task, we plot the average recall of the relevant examples against number of pages crawled (Figure 4). The plot illustrates the target-seeking behavior of the three crawlers if the examples are viewed as the targets. We again find BFS256 outperforming BFS1 while Breadth-First trails behind.

6 Related Research

Research on the design of effective focused crawlers is very vibrant. Many different types of crawling algorithms have been developed. For example, Chakrabarti et al. [8] use classifiers built from training sets of positive and negative

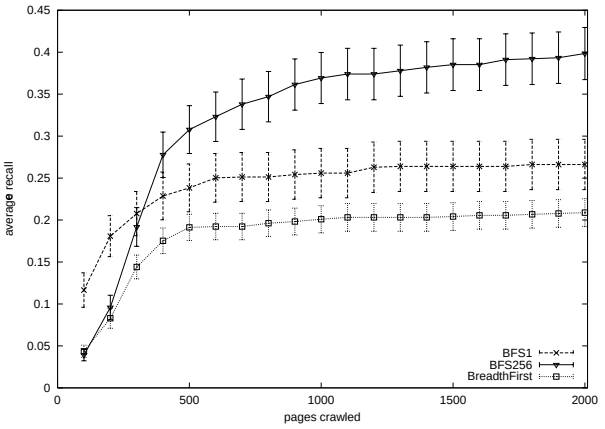

Figure 4: Average recall of examples when the crawls start from the neighbors.

example pages to guide their focused crawlers. Fetuccino [3] and InfoSpiders [18] begin their focused crawling with starting points generated from CLEVER [7] or other search engines. Most crawlers follow fixed strategies, while some can adapt in the course of the crawl by learning to estimate the quality of links [18, 1, 22].

The question of exploration versus exploitation in crawler strategies has been addressed in a number of papers, more or less directly. Fish-Search [11] limited exploration by bounding the depth along any path that appeared suboptimal. Cho et al. [9] found that exploratory crawling behaviors such as implemented in the Breadth-First algorithm lead to efficient discovery of pages with good PageRank. They also discuss the issue of limiting the memory resources (buffer size) of a crawler, which has an impact on the exploitative behavior of the crawling strategy because it forces the crawler to make frequent filtering decisions. Breadth-First crawlers also seem to find popular pages early in the crawl [20]. The exploration versus exploitation issue continues to be studied via variations on the two major classes of Breadth-First and Best-First crawlers. For example, in recent research on Breadth-First focused crawling, Diligenti et al. [12] address the "shortsightedness" of some crawlers when assessing the potential value of links to crawl. In particular, they look at how to avoid short-term gains at the expense of less-obvious but larger long-term gains. Their solution is to build classifiers that can assign pages to different classes based on the expected link distance between the current page and relevant documents.

The area of crawler quality evaluation has also received much attention in recent research [2, 9, 8, 19, 4]. For instance, many alternatives for assessing page importance have been explored, showing a range of sophistication. Cho et al. [9] use the simple presence of a word such as "computer" to indicate relevance. Amento et al. [2] compute the similarity between a page and the centroid of the seeds. In fact content-based similarity assessments form the basis of relevance decisions in several examples of research [8, 19]. Others exploit link information to estimate page relevance with methods based on in-degree, out-degree, PageRank, hubs and authorities [2, 3, 4, 8, 9, 20]. For example, Cho et al. [9] consider pages with PageRank score above a threshold as relevant. Najork and Wiener [20] use a crawler that can fetch millions of pages per day; they then calculate the average PageRank of the pages crawled daily, under the assumption that PageRank estimates relevance. Combinations of link and content-based relevance estimators are evident in several approaches [4, 7, 18].

7 Conclusions

In this paper we used an evaluation framework for topic driven crawlers to study the role of exploitation of link estimates versus exploration of suboptimal pages. We experimented with a family of simple crawler algorithms of varying greediness, under limited memory resources for two different tasks. A number of schemes and quality metrics derived from lexical features and link analysis were introduced and applied to gauge crawler performance.

We found consistently that exploration leads to better coverage of highly relevant pages, in spite of a possible penalty during the early stage of the crawl. An obvious explanation is that exploration allows to trade off short term gains for longer term and potentially larger gains. However, we also found that a blind exploration

when starting from neighbors of relevant pages, leads to poor results. Therefore, a mix of exploration and exploitation is necessary for good overall performance. When starting from relevant examples (Task 1), the better performance of crawlers with higher exploration could be attributed to their better coverage of documents close to the relevant subset. The good performance of BFS256 starting away from relevant pages, shows that its exploratory nature complements its greedy side in finding highly relevant pages. Extreme exploitation (BFS1) and blind exploration (Breadth-First), impede performance. Nevertheless, any exploitation seems to be better than none. Our results are based on short crawls of 2000 pages. The same may not hold for longer crawls; this is an issue to be addressed in future research. The dynamic evaluations do suggest that for very short crawls it is best to be greedy; this is a lesson that should be incorporated into algorithms for query time (online) crawlers such as MySpiders[3] [21].

The observation that higher exploration yields better results can motivate parallel and/or distributed implementations of topic driven crawlers, since complete orderings of the links in the frontier, as required by greedy crawler algorithms, do not seem to be necessary for good performance. Therefore crawlers based on local decisions seem to hold promise both for the performance of exploratory strategies and for the efficiency and scalability of distributed implementations. In particular, we intend to experiment with variations of crawling algorithms such as InfoSpiders [18], that allow for adaptive and distributed exploratory strategies.

Other crawler algorithms that we intend to study in future research include Best-First strategies driven by estimates other than lexical ones. For example we plan to implement a Best-N-First family using link estimates based on local versions of the $rank_{KW,PR}$ metric used in this paper for evaluations purposes. We also plan to test more sophisticated lexical crawlers such as InfoSpiders and Shark Search [14], which can prioritize over links from a single page.

[3] http://myspiders.biz.uiowa.edu

A goal of present research is to identify optimal trade-offs between exploration and exploitation, where either more exploration or more greediness would degrade performance. A large enough buffer size will have to be used so as not to constrain the range of exploration/exploitation strategies as much as happened in the experiments described here due to the small `MAX_BUFFER`. Identifying an optimal exploration/exploitation trade-off would be the first step toward the development of an adaptive crawler that would attempt to adjust the level of greediness during the crawl.

Finally, two things that we have not done in this paper are to analyze the time complexity of the crawlers and the topic-specific performance of each strategy. Regarding the former, clearly more greedy strategies require more frequent decisions and this may have an impact on the efficiency of the crawlers. Regarding the latter, we have only considered quality measures in the aggregate (across topics). It would be useful to study how appropriate trade-offs between exploration and exploitation depend on different characteristics such as topic heterogeneity. Both of these issues are the object of ongoing research.

References

[1] C. Aggarwal, F. Al-Garawi, and P. Yu. Intelligent crawling on the World Wide Web with arbitrary predicates. In *Proc. 10th Intl. World Wide Web Conference*, pages 96–105, 2001.

[2] B. Amento, L. Terveen, and W. Hill. Does "authority" mean quality? Predicting expert quality ratings of web documents. In *Proc. 23rd ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 296–303, 2000.

[3] I. Ben-Shaul, M. Herscovici, M. Jacovi, Y. Maarek, D. Pelleg, M. Shtalhaim, V. Soroka, and S. Ur. Adding support for dynamic and focused search with Fetuccino. *Computer Networks*, 31(11–16):1653–1665, 1999.

[4] K. Bharat and M. Henzinger. Improved algorithms for topic distillation in hyperlinked environments. In *Proc. 21st ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 104–111, 1998.

[5] B. E. Brewington and G. Cybenko. How dynamic is the Web? In *Proc. 9th International World-Wide Web Conference*, 2000.

[6] S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks*, 30(1–7):107–117, 1998.

[7] S. Chakrabarti, B. Dom, P. Raghavan, S. Rajagopalan, D. Gibson, and J. Kleinberg. Automatic resource compilation by analyzing hyperlink structure and associated text. *Computer Networks*, 30(1–7):65–74, 1998.

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

[9] J. Cho, H. Garcia-Molina, and L. Page. Efficient crawling through url ordering. In *Proc. 7th Intl. World Wide Web Conference, Brisbane, Australia*, 1998.

[10] Cyveillance. Sizing the internet. White paper, July 2000. http://www.cyveillance.com/web/us/corporate/white_papers.htm.

[11] P. De Bra and R. Post. Information retrieval in the World Wide Web: Making client-based searching feasible. In *Proc. 1st Intl. World Wide Web Conference*, 1994.

[12] M. Diligenti, F. Coetzee, S. Lawrence, C. L. Giles, and M. Gori. Focused crawling using context graphs. In *Proc. 26th International Conference on Very Large Databases (VLDB 2000)*, pages 527–534, Cairo, Egypt, 2000.

[13] T. Haveliwala. Efficient computation of pagerank. Technical report, Stanford Database Group, 1999.

[14] M. Hersovici, M. Jacovi, Y. S. Maarek, D. Pelleg, M. Shtalhaim, and S. Ur. The shark-search algorithm — An application: Tailored Web site mapping. In *Proc. 7th Intl. World-Wide Web Conference*, 1998.

[15] J. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.

[16] S. Lawrence and C. Giles. Accessibility of information on the Web. *Nature*, 400:107–109, 1999.

[17] F. Menczer. Links tell us about lexical and semantic web content. arXiv:cs.IR/0108004.

[18] F. Menczer and R. Belew. Adaptive retrieval agents: Internalizing local context and scaling up to the Web. *Machine Learning*, 39(2–3):203–242, 2000.

[19] F. Menczer, G. Pant, M. Ruiz, and P. Srinivasan. Evaluating topic-driven Web crawlers. In *Proc. 24th Annual Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, 2001.

[20] M. Najork and J. L. Wiener. Breadth-first search crawling yields high-quality pages. In *Proc. 10th International World Wide Web Conference*, 2001.

[21] G. Pant and F. Menczer. Myspiders: Evolve your own intelligent web crawlers. *Autonomous Agents and Multi-Agent Systems*, 5(2):221–229, 2002.

[22] J. Rennie and A. K. McCallum. Using reinforcement learning to spider the Web efficiently. In *Proc. 16th International Conf. on Machine Learning*, pages 335–343. Morgan Kaufmann, San Francisco, CA, 1999.

[23] G. Salton. *The SMART Retrieval System – Experiments in Automatic Document Processing*. Prentice-Hall, Englewood Cliffs, NJ, 1971.