# A SURVEY OF FOCUSED WEB CRAWLING ALGORITHMS

*Blaž Novak*
Department of Knowledge Technologies
Jozef Stefan Institute
Jamova 39, 1000 Ljubljana, Slovenia
e-mail: blaz.novak@ijs.si

**ABSTRACT**

**Web search engines collect data from the Web by "crawling" it – performing a simulated browsing of the web by extracting links from pages, downloading all of them and repeating the process ad infinitum. This process requires enormous amounts of hardware and network resources, ending up with a large fraction of the visible web[1] on the crawler's storage array. But when only information about a predefined topic set is desired, a specialization of the aforementioned process called "focused crawling" is used. What follows here is a short review of existing techniques for focused crawling.**

## 1. Introduction

The Web in many ways simulates a social network: links do not point to pages at random but reflect the page authors' idea of what other relevant or interesting pages exists. This information can be exploited to collect more on-topic data by intelligently choosing what links to follow and what pages to discard. This process is called "focused crawling".

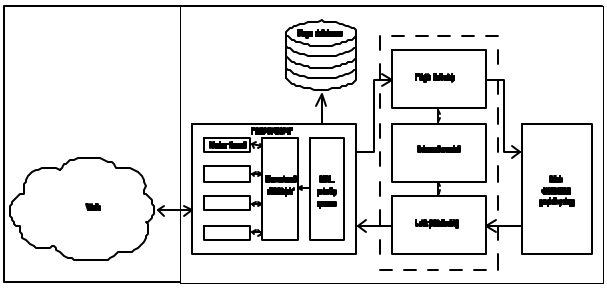

Figure 1

[1] "Visible web" is the part of the Web that can be accessed by only following the links. The vast majority of the structured information is however only accessible through constructing and submitting appropriate queries through web forms.

Figure 1 shows a structure of a simple focused crawler. The crawler is usually started with a set of seed pages that indicate the type of content the user is interested in and provide the initial links. These pages are put in a priority queue and are subsequently downloaded. Download manager must enforce several constraints including download speed and rate of retrieved pages that are located on a single host and domain while still trying to comply with URL priorities set by the rest of the system. That way slow remote servers and links are not overloaded by requests. Retrieved pages are then evaluated for topic relevance. This process may range from a simple keyword matching to complex machine learning classification schemes. Hyperlinks found on pages are extracted and ran through a filter. One possible reason for link to be omitted from the crawl is a presence of 'do not follow' META tag on the source page. It is also possible for the webmaster to specify parts of the site not to be indexed. Compliance with this so called 'Robots Exclusion Protocol' is not mandatory and can be administratively overridden on the crawler. The crawler administrator can also specify a list of pages and sites to be excluded from the crawl – for example to avoid infinitely large automatically generated crawler traps.

The next step is to predict the usefulness of following each link based on information seen so far and enqueueing it. Gathered pages can then be postprocessed and possibly the prediction model updated with new information. A non-focused crawler lacks the components marked with a dashed rectangle.

Focused crawlers are usually evaluated by "harvest rate" which is the ratio between number of relevant and all of the pages retrieved. "Loss rate" is then equal to 1 minus harvest rate.

A page from which a link was extracted is called a 'parent page' and the one to which the link points is a 'child page' or a 'target page'.

## 2. Crawling without external help

Some early work on the subject of focused collection of data from the Web was done by [DeBra94] in the context of client-based search engines. Web crawling was simulated by a "group of fish" migrating on the web. In the so called "fish search" each URL corresponds to a fish whose survivability is dependant on visited page relevance and remote server speed. Page relevance is estimated using a binary classification (the page can only be relevant or irrelevant) by a means of a simple keyword or regular expression match. Only when fish traverse a specified amount of irrelevant pages they die off - that way information that is not directly available in one 'hop' can still be found. On every document the fish produce offspring – its number being dependant on page relevance and the number of extracted links. The school of fish consequently 'migrates' in the general direction of relevant pages which are then presented as results. Starting point is specified by the user by providing 'seed' pages that are used to gather initial URLs. URLs are added to the beginning of the crawl list which makes this a sort of a depth-first search.

[Hersovici98] extends this algorithm into "shark-search". URLs of pages to be downloaded are prioritized by taking into account a linear combination of source page relevance, anchor text and neighborhood (of a predefined size) of the link on the source page and inherited relevance score. Inherited relevance score is parent page's relevance score multiplied by a specified decay factor. Unlike in [DeBra94] page relevance is calculated as a similarity between document and query in vector space model and can be any real number between 0 and 1. Anchor text and anchor context scores are also calculated as similarity to the query.

[Cho98] propose calculating the PageRank [Page98] score on the graph induced by pages downloaded so far and then using this score as a priority of URLs extracted from a page. They show some improvement over the standard breadth-first algorithm. The improvement however is not large. This may be due to the fact that the PageRank score is calculated on a very small, non-random subset of the web and also that the PageRank algorithm is too general for use in topic-driven tasks [Menczer01, Menczer02].

### 3. Crawling with the help of background knowledge

[Chakrabarti99] use an existing document taxonomy (e.g. pages in Yahoo tree) and seed documents to build a model for classification of retrieved pages into categories (corresponding to nodes in the taxonomy). The use of a taxonomy also helps at better modeling of the negative class: irrelevant pages are usually not drawn from a homogenous class but could be classified in a large number of categories with each having different properties and features. In this paper the same applies for the positive class because the user is allowed to have interest in several non-related topics at the same time. The system is built from 3 separate components: crawler, classifier and distiller. The classifier is used to determine page relevance (according to the taxonomy) which also determines future link expansion. Two different rules for link expansion are presented. Hard focus rule allows expansion of links only if the class to which the source page belongs with the highest probability is in the 'interesting' subset. Soft focus rule uses the sum of probabilities that the page belongs to one of the relevant classes to decide visit priority for children; no page is eliminated a priori. Periodically the distiller subsystem identifies hub pages (using a modified hubs&authorities algorithm [Kleinberg98]). Top hubs are then marked for revisiting.

Experiments show almost constant average relevance of 0.3 – 0.5 (averaged over 1000 URLs). Quality of results retrieved using unfocused crawler almost immediately drops to practically 0.

In [Chakrabarti02] page relevance and URL visit priorities are decided by separate models. The model for evaluating page relevance can be anything that outputs a binary classification, but the model for URL ranking (also called "apprentice") is on-line trained by samples consisting of source page features and the relevance of the target page (that kind of information is of course available only after both the source and the target page have been downloaded and the target page evaluated for relevance). For each retrieved page, the apprentice is trained on information from baseline (in this case the aforementioned taxonomy model) classifier (i.e. with what probability does the parent page belong to some class) and features around the link extracted from the parent page - to predict the relevance of the page pointed to by the link. Those predictions are then used to order URLs in the crawl priority queue. Number of false positives is shown to decrease significantly – between 30% and 90%.

[Ehrig03] consider an ontology-based algorithm for page relevance computation. After preprocessing, entities (words occurring in the ontology) are extracted from the page and counted. Relevance of the page with regard to user selected entities of interest is then computed by using several measures on ontology graph (e.g. direct match, taxonomic and more complex relationships). The harvest rate is improved compared to the baseline focused crawler (that decides on page relevance by a simple binary keyword match) but is not compared to other types focused crawlers.

[Bergmark02] describe modified 'tunneling' enhancement to best-first focused crawler approach. Since relevant information can sometimes be located only by visiting some irrelevant pages first and since the goal is not always to minimize the number of downloaded pages but to

collect a high-quality collection in a reasonable amount of time they propose to continue crawling even if irrelevant pages are found. With statistical analysis they find out that a longer path history does have an impact on relevance of pages to be retrieved in future (compared to just using the current parent pages relevance score) and construct a document distance measure that takes into account parent page's distance (which is in turn based on its parent page's distance etc).

## 4. Other approaches

[Angkawattanawit02] deal with improving recrawling performance by utilizing several databases (seed URLs, topic keywords and URL relevance predictors) that are built from previous crawl logs and used to improve harvest rate (percent of relevant pages retrieved). Seed URLs that will be used for future recrawls are computed using BHITS ([Bharat98]) algorithm on previously found pages - by selecting pages with high hub and authority scores. Keywords indicative for the target topic are extracted from title and anchor tags of previously found relevant pages. Link crawl priority is then computed as a weighted combination of similarity of link anchor text to topic keywords, source page score and predicted link score. Link score prediction is based on previously seen relevance for that specific URL.

[Aggarwal01] introduce a concept of "intelligent crawling" where the user can specify an arbitrary predicate (e.g. keywords, document similarity, … - anything that can be implemented as a function which determines documents relevance to the crawl based on URL and page content) and the system adapts itself in order to maximize the harvest rate. It is suggested that for some types of predicates the topical locality assumption of focused crawling (i.e. relevant pages are located close together) might not hold. In those cases the URL string, actual contents of pages pointing to the relevant one (not to be confused with the relevance of those pages!) or something else might do a better job at predicting relevance. A probabilistic model for URL priority prediction is trained using information about content of in-linking pages, URL tokens, short-range locality information (e.g. "parent does not satisfy predicate X but the children does") and sibling information (i.e. number of sibling pages matching the predicate so far).

### 5. Use of search engines

It is not necessary to use only the locally gathered data while crawling the web. Several attempts have been made to improve the harvest rate by utilizing search engines as a source of seed URLs and back-references, most notably [Diligenti00]. They try to solve the problem of "credit assignment" by using context graphs. It is pointed out that relevant pages can be found by knowing what kinds of off-topic pages link to them.

For each seed document a several layers deep graph is constructed that consists of pages pointing to that seed page. Because that information is not directly available from the web, a search engine is used to provide backward links. Graphs for all seed pages are then merged together and a classifier is trained to recognize a specific layer. Those predictions are then used to assign priority to the page.

Other possibilities of using remote sources include querying an index search engine for a set of seed documents, for dynamically re-seeding the crawler with random relevant pages or for retrieving all of the URLs altogether by constructing appropriate queries as done in [Ghani01].

### 5. Conclusion

The presented methods for focused crawling are not mutually exclusive and almost all of them can be incorporated into a unified framework for creation of focused corpora. Depending on the application needs however some of them are more appropriate than other. For a client-side data collection, extensive crawling can present a serious usability problem as it requires considerable amount of network resources and time. On the other hand collecting large corpuses of data imposes too much of a load on search engines and therefore requires more of a 'traditional' focused crawling technique.

### References


[Aggarwal01] "Intelligent Crawling on the World Wide Web with Arbitrary Predicates", C. Aggarwal, F. Al-Garawi and P. Yu. In *Proceedings of the 10th International World Wide Web Conference*, Hong Kong, May 2001.

[Angkawattanawit02] "Learnable Crawling: An Efficient Approach to Topic-specific Web Resource Discovery", N. Angkawattanawit, A. Rungsawang.

[Bergmark02] "Focused Crawls, Tunneling, and Digital Libraries", D. Bergmark and C. Lagoze and A. Sbityakov.

[Bharat98] "Improved algorithms for topic distillation in a hyperlinked environment", K. Bharat and M. R. Henzinger. In Proceedings of SIGIR-98, 21st {ACM} International Conference on Research and Development in Information Retrieval

[Chakrabarti99] "Focused Crawling: A New Approach to Topic-Specific Web Resource Discovery", S. Chakrabarti,

M. van den Berg and B. Dom. In *Proceedings of the 8th International WWW Conference*, Toronto, Canada, May 1999.

[Chakrabarti02] “Accelerated focused crawling through online relevance feedback”, S. Chakrabarti, K. Punera, and M. Subramanyam. In WWW, Hawaii, May 2002. ACM.

[Cho98] "Efficient Crawling Through URL Ordering", J. Cho, H. Garcia-Molina, L. Page. In *Proceedings of the 7th International WWW Conference*, Brisbane, Australia, April 1998.

[DeBra94] "Information Retrieval in Distributed Hypertexts", P. De Bra, G. Houben, Y. Kornatzky and R. Post. In *Proceedings of the 4th RIAO Conference*, 481 - 491, New York, 1994.

[Diligenti00] "Focused Crawling Using Context Graphs", M. Diligenti, F. Coetzee, S. Lawrence, C. Giles and M. Gori. In Proceedings of the 26th International Conference on Very Large Databases (VLDB 2000), Cairo, Egypt, September 2000.

[Ehrig03] “Ontology-focused Crawling of Web Documents”, M. Ehrig, A. Maedche. In Proceedings of the 2003 ACM symposium on Applied computing.

[Hersovici98] "The Shark-Search Algorithm - An Application: Tailored Web Site Mapping", M. Hersovici, M. Jacovi, Y. Maarek, D. Pelleg, M. Shtalhaim and S. Ur. In *Proceedings of the Seventh International World Wide Web Conference*, Brisbane, Australia, April 1998.

[Ghani01] "Building Minority Language Corpora by Learning to Generate Web Search Queries", R. Ghani, R. Jones and D. Mladenic. Technical Report CMU-CALD-01-100, 2001.

[Kleinberg98] “Authoritative Sources in a Hyperlinked Environment”, J. Kleinberg. *Proceedings of the ACM-SIAM Symposium of Discrete Algorithms* , 1998.

[Menczer01] "Evaluating Topic-Driven Web Crawlers", F. Menczer, G. Pant, P. Srinivasan and M. Ruiz. In Proceedings of the 24th Annual International ACM/SIGIR Conference, New Orleans, USA, 2001.

[Menczer02] "Topic-driven crawlers: Machine learning issues", F. Menczer and G. Pant and P. Srinivasan. ACM TOIT, Submitted, 2002.

[Page98] "The PageRank Citation Ranking: Bringing Order to the Web", L. Page, S. Brin, R. Motwani, T. Winograd. Stanford Digital Library Technologies Project.