Focused Web Crawling Algorithms

Andas Amrin*, Chunlei Xia, Shuguang Dai
University of Shanghai for Science and Technology, 516 Junggong Rd., Yangpu District, Shanghai, China.

* Corresponding autor. Tel.: +8613817338874; email: andas999@mail.ru
Manuscript submitted February 13, 2015; accepted May 21, 2015.
doi: 10.17706/jcp.10.4.245-251

Abstract: Nowadays the web is rich of any kind of information. And this information is freely available thanks to the hypermedia information systems and the Internet. This information greatly influenced our lives, our lifestyle and way of thinking. A web search engine is a complex multi-level system that helps us to search the information that available on the Internet. A web crawler is one of the most important parts of the search engine. It's a robot that systematically browses and indexes the World Wide Web. A focused web crawler is used crawling only web pages that are relevant to the user given topic or web page link. A focused crawler is a part of the search system that helps user to find most relevant information from the Internet. In our days, this area of computer science is very popular and important for the development of science and technology, because The Internet is constantly and rapidly growing, and the information in it. In this article we will review most effective focused web crawling algorithms that determines the fate of the search system.

Key words: Data mining, focused web crawling algorithms, search engine.

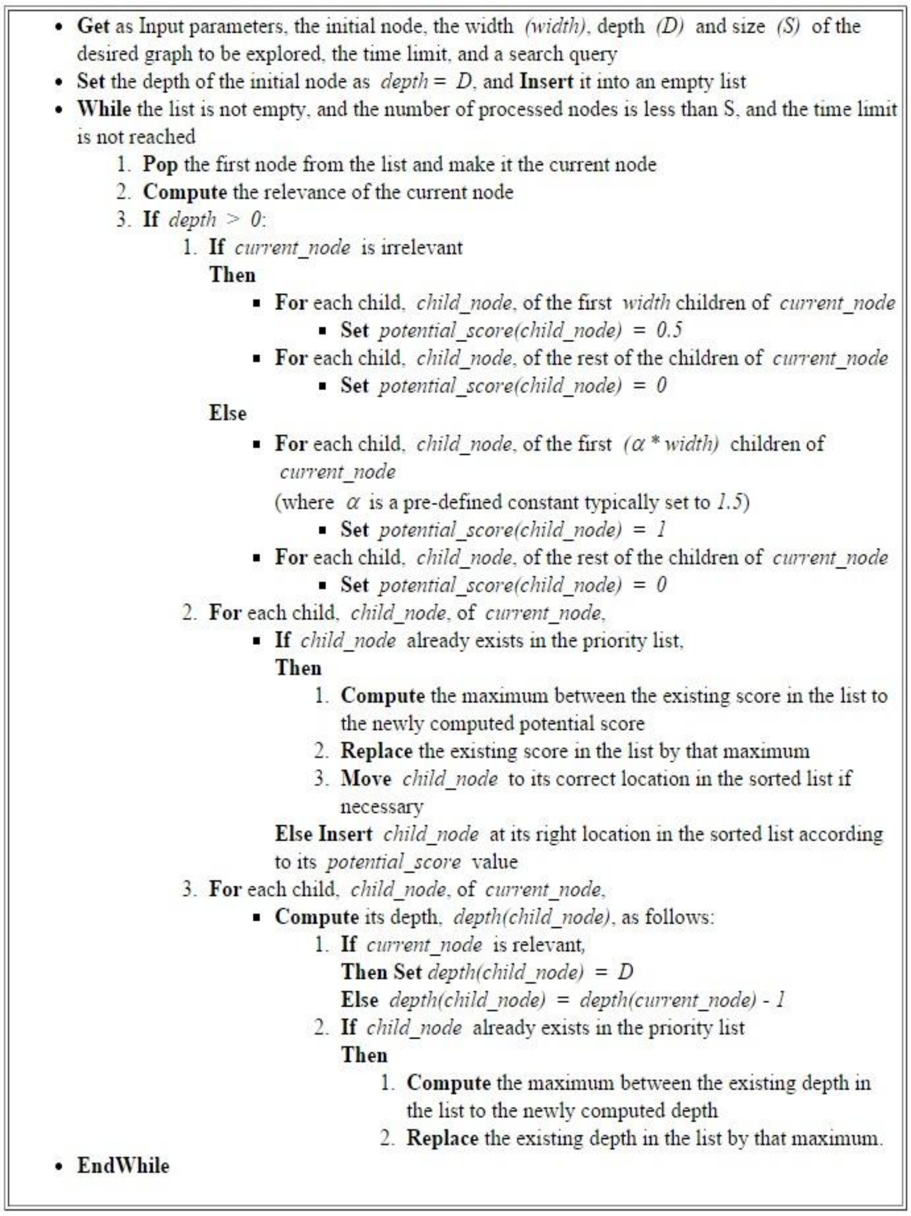
1. Introduction

The World Wide Web is growing exponentially, and the amount of information in it is also growing rapidly. In the early days of the Internet, search engines used very simple methods and web crawling algorithms, like Breadth First Search or Depth First Search, to crawl and index web pages and data in it. But because of low relevance results of this methods, researchers and engineers implementing more effective strategies, methods and algorithms of web crawling. One of these methods is a focused web crawling method that allows search engines to find web pages of high relevance more effectively. One of the pioneer researchers in this area that fairly comprehensively described the principles of focused crawling strategy is Soumen Chakrabarti. His research [1] have a huge impact on the research of many researchers in this area. This days search engines uses more complex algorithms and data analysing systems, based on the technologies and techniques of artificial intelligence.

2. Algorithms of Focused Web Crawling

2.1. The Fish Search Algorithm

Fish Search algorithm [2], [3] is an algorithm that was created for efficient focused web crawler. This algorithm is one of the earliest focused crawling algorithms. Fish Search focused crawling algorithm that was implemented to dynamically search information on the Internet. Searching system using Fish search algorithm is more efficient than ordinary search systems as it uses a navigation strategy to automatically browse the web.



```
• Get as Input parameters, the initial node, the width (width), depth (D) and size (S) of the
  desired graph to be explored, the time limit, and a search query
• Set the depth of the initial node as depth = D, and Insert it into an empty list
• While the list is not empty, and the number of processed nodes is less than S, and the time limit
  is not reached
      1. Pop the first node from the list and make it the current node
      2. Compute the relevance of the current node
      3. If depth > 0:
            1. If current_node is irrelevant
               Then
                  ▪ For each child, child_node, of the first width children of current_node
                        ▪ Set potential_score(child_node) = 0.5
                  ▪ For each child, child_node, of the rest of the children of current_node
                        ▪ Set potential_score(child_node) = 0
               Else
                  ▪ For each child, child_node, of the first (α * width) children of
                    current_node
                    (where α is a pre-defined constant typically set to 1.5)
                        ▪ Set potential_score(child_node) = 1
                  ▪ For each child, child_node, of the rest of the children of current_node
                        ▪ Set potential_score(child_node) = 0
            2. For each child, child_node, of current_node,
                  ▪ If child_node already exists in the priority list,
                    Then
                        1. Compute the maximum between the existing score in the list to
                           the newly computed potential score
                        2. Replace the existing score in the list by that maximum
                        3. Move child_node to its correct location in the sorted list if
                           necessary
                    Else Insert child_node at its right location in the sorted list according
                    to its potential_score value
            3. For each child, child_node, of current_node,
                  ▪ Compute its depth, depth(child_node), as follows:
                        1. If current_node is relevant,
                           Then Set depth(child_node) = D
                           Else depth(child_node) = depth(current_node) - 1
                        2. If child_node already exists in the priority list
                           Then
                              1. Compute the maximum between the existing depth in
                                 the list to the newly computed depth
                              2. Replace the existing depth in the list by that maximum.
• EndWhile
```

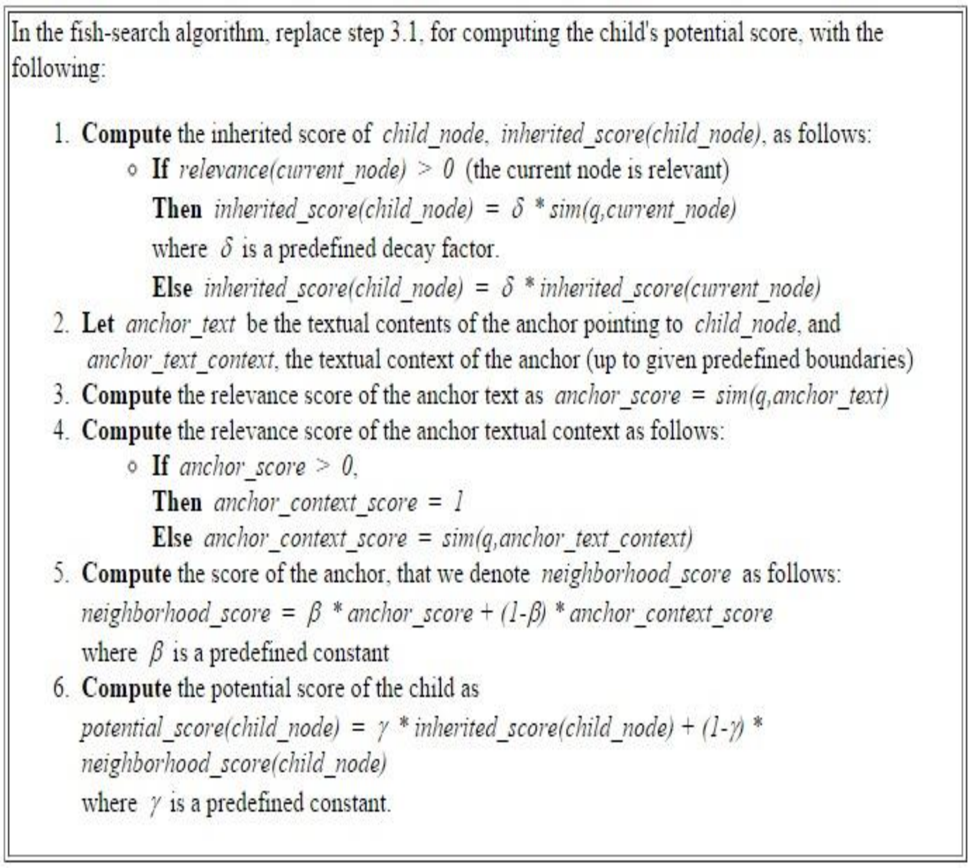
Fig. 1. The fish-search algorithm pseudo-code.

In research [3], the Fish Search crawling system was implemented as a client based searching system tool that automatically navigates which webpage to crawl, thereby working more like a browsing user, but acting much faster and follows an optimized strategy. Client-based crawling have some significant disadvantages, like slow operation and resource consumption of the network.

The algorithm requires three types of actions:

1) The most important and difficult is the first step, it requires finding starting URLs, which will be the starting point of searching process.
2) Web documents are extracted and scanned for the information which is relevant at the receiving end.
3) The extracted web documents are reviewed to find links to other web documents or URL's of webpages.

The main principle of the Fish Search algorithm can be expressed as the following: at first it takes input data which is seed URL and query, and creates a priority list in dynamic way (according to the seed URLs) of the next web links (we call it nodes) to be explored. At every step of creation of the list, the first node is retrieved from the URL list and processed. As each text of the documents be available, it is processed and analyzed by a scoring tool ranking whether this document is relevant (1) or irrelevant (0) to the search query and, considering the score, the algorithm decides whether to continue the exploration in that route or not: Whenever a web data source is fetched, it is scanned for URLs. The URL nodes acuminated to by these links (marked "children") are each assigned a value of depth. If the parent URL is marked as relevant, the depth of the children URL is set to some predetermined value. Else, the depth of the children URL is set to be one less than the depth of the parent link. When the depth reaches 0, the direction is dropped and none of its children URL is included into the URL list. Fig. 1 is a description of the Fish Search algorithm.



```
In the fish-search algorithm, replace step 3.1, for computing the child's potential score, with the
following:

  1. Compute the inherited score of child_node, inherited_score(child_node), as follows:
       ○ If relevance(current_node) > 0 (the current node is relevant)
         Then inherited_score(child_node) = $\delta$ * sim(q,current_node)
         where $\delta$ is a predefined decay factor.
         Else inherited_score(child_node) = $\delta$ * inherited_score(current_node)
  2. Let anchor_text be the textual contents of the anchor pointing to child_node, and
     anchor_text_context, the textual context of the anchor (up to given predefined boundaries)
  3. Compute the relevance score of the anchor text as anchor_score = sim(q,anchor_text)
  4. Compute the relevance score of the anchor textual context as follows:
       ○ If anchor_score > 0,
         Then anchor_context_score = 1
         Else anchor_context_score = sim(q,anchor_text_context)
  5. Compute the score of the anchor, that we denote neighborhood_score as follows:
     neighborhood_score = $\beta$ * anchor_score + (1-$\beta$) * anchor_context_score
     where $\beta$ is a predefined constant
  6. Compute the potential score of the child as
     potential_score(child_node) = $\gamma$ * inherited_score(child_node) + (1-$\gamma$) *
     neighborhood_score(child_node)
     where $\gamma$ is a predefined constant.
```

Fig. 2. The shark-search algorithm pseudo-code.

The Fish Search system use a principle of the fish school metaphor, where URL presented like a fish, searching for food and producing children (called width). They move in the food's direction. The distance that fish and children can go without food is called depth. When they find fish they live longer and reproduce a lot. If they do not find any food they die and children also. And also polluted water have negative impact on their destiny. The Fish Search search criteria is based on the keyword search, regular expression search and external filters. In spite of the advantages of the system, the fish search system have some limitations in time consuming, usage of network resources and it can't be available other WWW browser users.

The Fish Search has a few limitations of the search:

1) Crawling and downloading web documents through the WWW may be significantly time consuming, which is unacceptable for users.
2) The usage of resources of network is occasionally considered terrifically high. Compared to ordinary users visiting webpages and reading documents the fish search crawlers significantly loads not only network, also web servers.
3) The algorithm can only retrieve documents for which URL's are found in other web pages. It means that this algorithm can't search documents from the "hidden" web.

The efficiency of the algorithm depends on the ability to find more relevant documents and avoid processing of irrelevant documents. And also, the crawler must explore a substantial part of the Web as fast as possible.

2.2. The Shark Search Algorithm

As is shown in Fig. 2, shark search algorithm [2] is an improved version of the Fish Search algorithm. While this algorithm uses the same simple Fish School metaphor, it discovers and retrieves more relevant information in the same exploration time with improved search abilities. The Shark Search system uses better relevance scoring techniques for neighboring pages before accessing and analyzing them. The system have a significant impact on search efficiency because of improvement of the relevance classification system. It uses a score between 0 and 1, instead of the binary evaluation of information relevance. This approach gives much better results than binary classification. The second improvement is the method of inheritance of node's children. System gives every child an inherited score that have a huge impact on the relevance score of the children and children's children. And most significant improvement is that system calculates children's relevance using not only ancestor's heritage, but also use meta-data to analyze its relevance score. According to an experiment results the Shark Search is more effective in quality of information retrieved and operation time than its ancestor.

2.3. The Best-First Search

The Best-First [4] algorithm focuses on the retrieval of pages which are relevant to a particular given topic. It's an algorithm that uses a score to define which page has a best score. This algorithm uses a rule to select the best page. In most cases it uses artificial intelligence algorithms (Naïve Bayes, Cosine Similarity, Support Vector Machine, k-nearest neighbour algorithm, Gaussian mixture model, etc.) as a classifier to detect the best result. In many articles this algorithm has the best crawling results.

```
insert in ready queue(seeds)
while true do
if more links in ready queue then
link := dequeue best
doc := fetch(link)
score := apply rule(doc)
out links := extract links(doc)
save score(out links, score)
else
sorted links := sort(non processed queue)
insert in ready queue(sorted links)
end if
end while
```

Fig. 3. The best-first algorithm pseudo-code.

This algorithm is an algorithm of focused search which explores a graph by expanding the most hopeful node, this node selects according to a specific rule. The Best-first search algorithm principle is in evaluating the promise of node n by a heuristic technique estimation function $f(n)$ which, generally, may depend on the specification of n, the specification of the goal, the information assembled by the search up to that point, and the most important, on any additional knowledge about the problem domain. The Fig. 3 shows a pseudocode of the algorithm.

2.4. The Context Graph Focused Crawling Algorithm

Context Graph Focused Crawler [5] uses the limited capability of traditional search engines to allow users to query for pages linking to a specified document. This data can be represented as a graph which connected with each other and have a certain minimum distance necessary to move from one graph to another. This kind of data representation is used to train a set of classifiers to detect and assign documents to different categories depending on the expected link distance from the document to the target document. During the crawling process these classifiers are used to define the distance between a target document and a current processing document. And then all information is used to optimize and categorize the search. There are two main stages of the algorithm: an initialization phase when a set of context graphs and associated classifiers are constructed for each of the seed document; and a crawling phase that uses the classifier to lead the search process and updates context graph.

```
insert in ready queue(seeds)
while true do
    if more links in ready queue then
        link := dequeue best
        doc := fetch(link)
        for i = 0 to num layers do
            scoreL[i] := classifyLi(doc)
        end for
        layer belong := maximum(scoreL[i])
        out links := extract links(doc)
        save score(out links, scoreL, layer belong)
    else
        sorted links := sort(non processed queue)
        insert in ready queue(sorted links)
    end if
end while
```

Fig. 4. The context graph focused crawling algorithm pseudo-code.

```
{initialize each agent's genotype, energy and starting page}
PAGES ? maximum number of pages to visit
while number of visited pages < PAGES do
    while for each agent a do
        {pick and visit an out-link from the current agent's page}
        {update the energy estimating benefit() - cost()}
        {update the genotype as a function of the current benefit}
        if agent's energy > THRESHOLD then
            {apply the genetic operators to produce offspring}
        else
            {kill the agent}
        end if
    end while
end while
```

Fig. 5. The infospiders algorithm pseudo-code.

The Context Graph Focused Crawling Algorithm at first uses traditional search engine, like Google, Bing or Yahoo to let users to search for webpages linking to a specific data. This information can be used to make a representation of webpages that occur within a certain linking distance (min number of links needed to get from one page to another) of the target webpage. This technique used to train a set of special classifiers, which programed to find and assign documents to different categories considering link distance from one page to the target page. During the crawling process these classifiers are programmed to analyze the distance and predict approximate distance. Then this data used to control the crawling process. This focused web crawling process have two main phases: 1. An initialization phase which assumes creating context graphs and classifiers. 2. A crawling phase which supposes to use classifier to lead a crawling process and updates context graphs. The Fig. 4 presents a pseudo-code of the algorithm.

2.5. The InfoSpiders Algorithm

InfoSpiders [6] are a multiagent system in which each agent in a population of peers adapts to its local information environment by learning to estimate the value of hyperlinks, while the population as a whole attempts to cover all promising areas through selective reproduction. The agent interacts with the information environment that consists of the actual networked collection (the Web) plus information kept on local data structures. It's a type of agent-based systems that can browse online on behalf of the user, and evolve an intelligent behavior that exploits the Web's linkage and textual cues. InfoSpiders is the name of online multiagent that search only the current environment and therefore will not return stale information, and will have a better chance to improve the regency of the visited documents. These agents dynamically crawl the Web in online mode, and uses artificial intelligence techniques to adapt to the characteristics of its networked information environment. In Fig. 5 we can see a pseudocode of the algorithm.

```
insert in ready queue(seeds)
while true do
    if more links in ready queue then
        link := dequeue best
        doc := fetch(link)
        svm score := classify(doc)
        out link[] := extract links(doc)
        out anchor[] := extract links(doc)
        anchor score[] := classify(out anchor[])
        final score := svm score * factor - anchor score * (1- factor)
        save score(out links, final score)
        for i = 0 to num out links do
            if abs(anchor score[i]) > threshold then
                add to training(anchor score[i])
            end if
        end for
    else
        threshold := retrain anchor
        sorted links := sort decreasing(non processed queue)
        insert in ready queue(sorted links)
    end if
end while
```

Fig. 6. The learning anchor algorithm pseudo-code.

2.6. The Learning Anchor Algorithm

In Ref. [7] and [8] was implemented and tested a Learning Anchor Algorithm to define a relevance score of each link. This algorithm uses a classifier that assigns each link a different score depending on their anchor text information. This classification technique is more effective than the classification technique of the whole page. Also they use a method that combines the anchor and the whole parent document (using a Bayesian representation). Concluding experiment results we can conclude that anchor text alone can give much better information about the page pointed by the link than the document containing the link itself. But the main problem of this method is that anchor score does not contribute to the evaluation when the anchor text doesn't contain any information. The figure 6 presents a pseudocode of the algorithm.

In Ref. [3] thesis researchers uses not just only the anchor text, they use the whole DOM structure to classify links. This approach helps a focused crawler to assign better priorities to the unvisited links in the crawl frontier that leads to a higher rate of fetching pages and decrease false events that positively influence on PCU, network and storage resources.

3. Conclusion

In this article mainly was made a review of most effective focused crawler search algorithms. The crawling algorithm is the most important part of any search engine. Focused Crawlers uses more complex systems and techniques to define the information of high relevance and quality. Searching algorithm is the heart of the search engine system. The choice of the algorithm have a significant impact on the work and effectiveness of focused crawler and search engine. According to [9] research the best crawling algorithm is the Best First search algorithm using artificial intelligence classifiers like Support Vector Machine, Naïve Bayes Classifier, String Matching, etc. Some researchers like [1] use semi-supervised learning methods to crawl and analyze the crawled information. But according to [9] the best method to crawl most relevant information is to use unsupervised learning methods combined with focused crawling algorithms that defines the best result of currently crawling pages.

References

[1] Chakrabarti, S., van der Berg, M., & Dom, B. (1999). Focused crawling: a new approach to topic-specific web resource discovery. *Proceedings of the 8th International World-Wide Web Conference (WWW8)*.

[2] Hersovici, M., Jacovi, M., Maarek, Y., Pelleg, D., Shtalhaim, M., & Ur, S. (1998). The shark-search algorithm — an application: tailored web site mapping. *Proceedings of the Seventh International World Wide Web Conference, Brisbane*, Australia.

[3] De Bra, P. M. E., & Post, R. D. J. (1994). Information retrieval in the world wide web: Making client-based searching feasible. *Computer Networks and ISDN Systems, 27(2),* 183-192.

[4] Jamali, M., Sayyadi, H., *et al.* (2006). A method for focused crawling using combination of link structure and content similarity. *Proceedings of IEEE/WIC/ACM International Conference on Web Intelligence*.

[5] Diligenti, M., Coetzee, F., Lawrence, S., Giles, C. L., & Gori, M. (2000). Focused crawling using context graphs. *Proceedings of 26th International Conference on Very Large Databases* (pp. 527-534).

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

[7] Chakrabarti, S., Punera, K., & Subramanyam, M. (2002). Accelerated focused crawling through online relevance feedback.

[8] Passerini, A., Frasconi, P., & Soda, G. (2001). Evaluation methods for focused crawling. *Lecture Notes in Computer Science, 2175,* 33-45.

[9] Dorado, I. G. (2008). Focused Crawling: Algorithm survey and new approaches with a manual analysis.

Andas Amrin was born in Almaty, Kazakhstan on June 30, 1990. He received his B.Tech degree in computer science from Kazakh National Technical University named after K. I. Satpayev in Almaty, Kazakhstan in 2011 and the M.S. degree in computer science from the University of Shanghai for Science and Technology in Shanghai, China, in 2015. His research interests include hypertext information retrieval, web analysis, artificial intelligence and data mining.