Gray Tunneling Based on Block Relevance for Focused Crawling

Na Luo[1,2] Wanli Zuo[1] Fuyu Yuan[3]

[1]College of Computer Science and Technology, JiLin University Key Laboratory of Symbol Computation and Knowledge Engineering of the Ministry of Education, Changchun 130012, P. R. China
[2]Computer Science Department, Northeast Normal University, Changchun 130117, P. R. China
[3]ChangChun Institute of Applied Chemistry Chinese Academy of Science, Changchun 130022, P. R. China

Abstract

In this paper the Gray Tunneling is defined and content block algorithm is used to enhance focused crawler's ability of traversing Gray Tunneling. Gray Tunneling resolves the problem that the topic-multiplicity of a web page makes the relevance of the highly relevant page to be weakened. So during the process of crawling, in order to avoid the effect caused by the web page that is irrelevant to the specific topic as a whole but relevant partially, we divide a multi-topical page into several blocks and process the blocks individually, and then we can traverse the page that is irrelevant as a whole to expand the scope crawler reached and get more relevant pages. A comprehensive experiment has been conducted, the result shows obviously that this approach outperforms Best-First and Breadth-First algorithm both in harvest rate and efficiency.

Keywords: Focused crawler, Gray tunneling, Content block, Local relevance.

1. Introduction

Unlike general-purpose web crawler which automatically traverses the web and collects all web pages, focused crawling is designed to gather collection of pages on specific topic. A focused crawler tries to "predict" whether or not a target URL is pointing to a relevant and high-quality Web page before actually fetching the page.

Focused Crawling, while quite efficient and effective does have some drawbacks. One is that it is not necessary optimal to simply follow a "best-first" search, because it is sometimes necessary to go through several irrelevant-topic pages to get to the next relevant one. With some probability, the crawl should be allowed to follow a series of bad pages in order to get to a good one. So we need to traverse the irrelevant web page to get more relevant pages. This procedure we called Tunneling.

We partitioned Tunneling into Gray Tunneling and Black Tunneling. Web pages as a whole is irrelevant, but local web pages particularly which contains a number of links is relevant; we call this Gray Tunneling (Fig 1). Gray Tunnel reflects a translucence link between contexts.

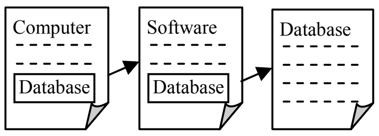

Fig. 1: Instance Diagram of Gray Tunneling

Definition 1(Tunneling Length): Let $n_1, n_2, \cdots n_k$ are web pages which have links but the whole irrelevant to the topic. p as a on-topic web page, and there have links from n_k to p . We call $n_1, n_2, \cdots n_k \to p$ is a Tunneling here is the length of Tunneling.

Definition 2(Content Block): Let n_i, n_j are web pages and there have a link l between n_i and n_j , the link text hereabout l called content block. Content Block's size can be established on the basis of need. The smallest content block is anchor text, the larger is a sub-tree of Dom tree which contains link l , the largest one is the whole document.

Definition 3(Gray Tunneling) Consider Tunneling as $n_1, n_2, \cdots n_k \to p$, for n_i and n_{i+1} , if both of them are relevant to content block , we consider this tunneling as Gray Tunneling.

Tunneling is useful for getting to desirable parts of the web. Clearly tunneling can improve the effectiveness of focused crawling by expanding its reach, and its efficiency by pruning paths which look hopeless. So we present the background and related work about tunneling in Sect. 2. Notice that the most important part of the paper that gives the algorithms for content block partition and Gray Tunneling in Sect. 3. There are experiments for evaluating our algorithm in Sect.4.

2. Related work

An essential weakness of the baseline focused crawler is its inability to model tunneling [1]; that is, it can't traverse through a chain of irrelevant pages to reach a relevant page. Two remarkable projects, the context-graph-based crawler and Cora's focused crawler tackled tunneling problem.

The context-graph-based crawler uses a best-search heuristic, but the classifiers learn the layers representing a set of pages that are at some distance to the pages in the target class (layer 0) [2]. The crawler simply uses these classifier results and inserts URLs extracted from a layer- page to the layer- queue; that is, it keeps a dedicated queue for each layer. While deciding the next page to visit, the crawler prefers the pages nearest to the target class—that is, the URLs popped from the queue that correspond to the first nonempty layer with the smallest layer label. This approach clearly solves the problem of tunneling, but unfortunately requires constructing the context graph, which in turn requires finding pages with links to a particular page(back links). Our rule-based crawler, on the other hand, uses forward links while generating the rules and transitively combines these rules to effectively imitate tunneling behavior.

Tunneling has been regarded as a difficult problem in focused crawling. Less related papers have discussed. In [3], Donna Bergmark proposed the concept of nugget and dud. A nugget is a Web document whose cosine correlation with at least one of the collection centroids is higher than some given threshold. Thus the "nugget-ness" of a document is represented by its correlation score. A dud, on the other hand, is a document that does not match any of the centroids very high. A path is the sequence of pages and links going from one nugget to the next. The path length is 2 minus the number of duds in the path. A crawl is the tree consisting of all the paths, linked together in the obvious way. In [4] Pant has improved Best-First algorithm, It begin with a single family of crawler algorithms with a single greediness parameter to control the exploration behavior. He examined the Best-N-First family of crawlers where the parameter N controls the characteristic of interest.

3. Gray tunneling based on block relevance

3.1. Content block and content block algorithm

A content block is a self-contained logical region within a page that has a well defined topic or functionality. A page can be decomposed into one or more contend blocks, corresponding to the different topics and functionalities that appear in the page. The contents of block size uncertain, the largest can cover the entire web pages, the smallest is the eighth page space even sixteenth. We propose that content blocks, as opposed to pages, are the more appropriate unit for information retrieval. The main reason is that they are more structurally cohesive, and better aligned.

In [5], There is a definition of content block, "A content block is a region of a web page that (1) has a single well-defined topic or functionality: and (2) is not nested within another region that has exactly the same topic or functionality. That is, we have two contradicting requirements from a content block: (1) that it will be "small" enough as to have just one topic or functionality; and (2) that it will be "big" enough such that no other region may have a more general topic."

In order to partition a page into blocks, we partition a page into blocks. we need a syntactic definition of blocks, which will materialize the intuitive requirements of the semantic definition into an actual algorithm. This problem was considered before by Chakrabarti et al. [6]; they suggested a sophisticated algorithm to partition "hubs" in the context of the HITS/Clever algorithm into content blocks. Since our primary goal is to design efficient hypertext cleaning algorithms that run in data gathering time, we adopt a simple heuristic to syntactically define content blocks. This definition has the advantages of being context-free, admitting an efficient implementation, and approximating the semantic definition quite faithfully.

We build a DOM tree. Each node of DOM tree can be labeled as a block. Our aim is to find a suitable marker block. In order to facilitate analysis we have neglected those nodes which show the attribute of label and uses the cues provided by HTML mark-up tags such as tables, paragraphs, headings, lists, etc. We define page partitioning algorithm in the following way:

```
Algorithm 1: Content_Block_partition Alg.
Input:Original Web page P;
      Level Parameter k
Output:Results of Content Block
Content_Block_Partition(P)
{
    Tp := HTML parse tree of P.
    Initialize(Tp)
    Tidy(Tp)
    Queue := root of Tp
    while(Queue is not empty)
    {
        v := top element in Queue
        s := Block_Height(root,0) * α
        if(v has a child with at least k links and Block_Height(v,0)⩾s)
            push all the children of v to the Queue
        else
            declare v as a block
    }
}
Block_Height(Node iNode, int height)
{
    count = height;
    if(iNode has child nodes)
        children = the child nodes of iNode
        len = the length of children
        htemp = 0
        fortemp = count
        for each child in children
            if(the child node is content block tags)
                htemp = Block_Height(childnode, count+1)
            else
                htemp = Block_Height(childnode,count)
            if(htemp>fortmp)
                fortmp=htemp
        if(fortmp>count)
            count=fortmp
    Return count
}
```

Here Block_Height(v,0) refers to the height of subtree whose root is v in Tp. α is a threshold, whose value is given by experience. We set the threshold to 0.5 and k=1.

We use Yahoo! web page for testing, After the process of parsing extracting html page and eliminating noise, we partition yahoo web page into 8 blocks. Fig 2 shows 8 blocks we have partitioned and Fig 3 shows the DOM tree of block 4.

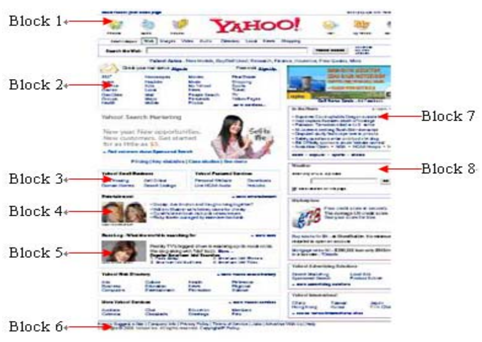

Fig. 2: Yahoo Web page after Content Block algorithm.

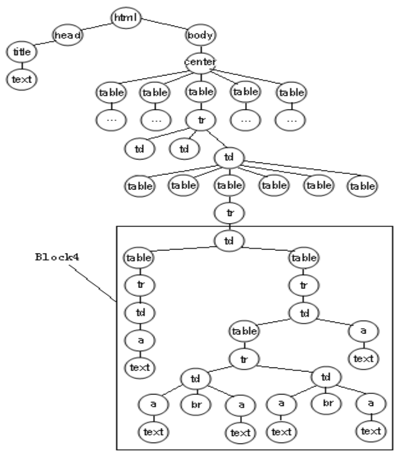

Fig. 3: DOM tree of Block 4.

3.2. Gray tunneling

Bergmark [5] proposed to use Tunneling technique to address the problems of local search. Tunneling is a heuristic-based method that solves simple global optimization problem. In the focused crawling scenario, a focused crawler using Tunneling will not give up probing a direction immediately when an irrelevant page is encountered. Instead, it continues searching in that direction for a pre-set number of steps. This allows the focused crawler to travel from one relevant Web community to another when the gap (number of irrelevant pages) between them is within a limit. Experiment results showed that focused crawlers with Gray Tunneling capability find more relevant pages than those without Tunneling.

Algorithm 2: Focused Crawler with Gray Tunneling
Input: seed URLs: *starting_urls*
Output: the pages relevant to the topic

```
Gray Tunneling(starting_urls)
{
    enqueue(url_queue, starting_url);
    while (not empty(url_queue) and not termination)
        url = dequeue(url_queue);
        page = crawl_page(url);
        enqueue(crawled pages, url);
        block_list= Content_Block_Partition(page);
        for each block in block_list
            for each link in block
                evaluate(link,block)
                add link to url_list
        for each link in url_list
            if(link ∉ url_queue and link ∉ crawled pages)
                enqueue(url_queue, link);
                reorder queue(url_queue);
}
```

4. Experiments and analysis

We use several strategies for focused crawler testing in order to verify the capability of tunneling. The same as the previous test, we use multiple threads in java for 37 topics in Open Directory Project [7] for evaluation. During evaluation, we adopt Harvest rate, Target recall and Target length to measuring performance. For Harvest and Target recall, each topic has crawled 5000 pages, and for Target length, each topic has crawled 5000 relevant pages. For each topic we select 100 seed urls and use 50 threads crawling at the same time.

4.1. Performance evaluation

4.1.1. Harvest Rate

It used to estimate the pages obtained by focused crawler whether is relevant to the topic. Harvest rate formula is followed:

$$harvest\_rate = \frac{relevant\_pages}{pages\_downloaded}$$

Here, pages_downloaded represented the total download pages and relevant_pages represented total download relevant-pages.

4.1.2. Target Recall

It used to calculate the relevant pages in proportion to the whole network. Speaking for the whole www, it is difficult to determine the total number of relevant pages, hence the need for a simulation method to estimate the actual recall, we call this Target Recall. In [70], P.Srinivasan supposed *R* is a set of all relevant pages in web, *T* is a random subset of *R*. So

$$R(t) = \frac{|C(t) \cap T|}{|T|}$$

Here C(t) is a pages set which obtained from web.

4.1.3. Target Length

Definition 4(Target Length): Target length l is the distance from seed url to the target pages. Fig. 4 shows the method. Gray panes represent relevant pages and white panes are opposite. There have some content block is relevant to the topic in those irrelevant pages. Beelines represent the path of focused crawler which real lines are crawled path and broken lines are opposite.

We come to the conclusion, after have crawled a certain number of pages, the shorter of total crawling path length the stronger of the capability of tunneling.

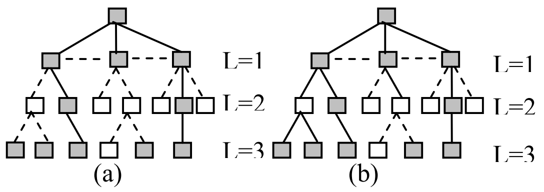

Fig. 4: Crawled Path when encounters irrelevant pages.(a) Without tunneling (b) Adopt tunneling.

4.2. Experiment results

Breadth-First without judging on the context of the unvisited URL-s, performed not well. It depends heavily on the localization of the relevant pages and web sites. Anchor text, like user's query of a search engine, is typically very short, consisting of very few terms on average, and contains rich, human-oriented information of the linked document within the context of the source document being visited. Best-First predicts the relevance of page potential URL-s by referring to the whole context of the visited web page. All out-links in one page have the same priorities. It only grouped the unvisited URL-s based on the page picked up from, and there is no difference within each group. So it has low accuracy when there is a lot of noise in the page or the page contains multiple topics.

Gray Tunneling Algorithm first partition the page into different content blocks based on different topics. Unlike Best-First, it predicts the relevance of page potential URL-s by referring to content block of the visited web page. Out-links in one page have different priorities. We first visit the highest priority page along out links. Best-First's weakness is just content block crawler's strength.

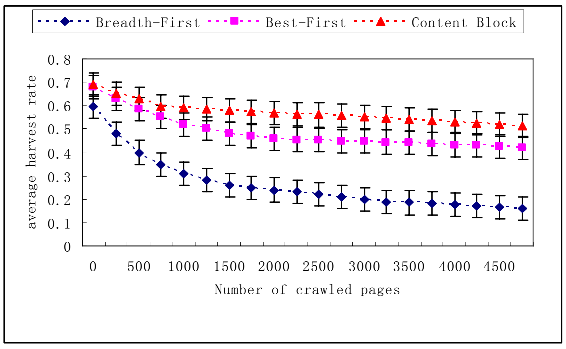

Fig. 5: Average harvest rate of each topic web pages.

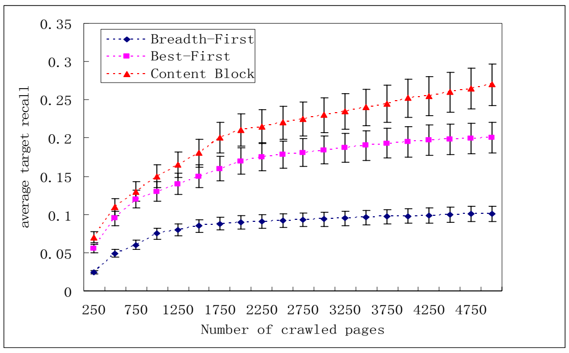

Fig. 6: Average target recall of each topic web pages.

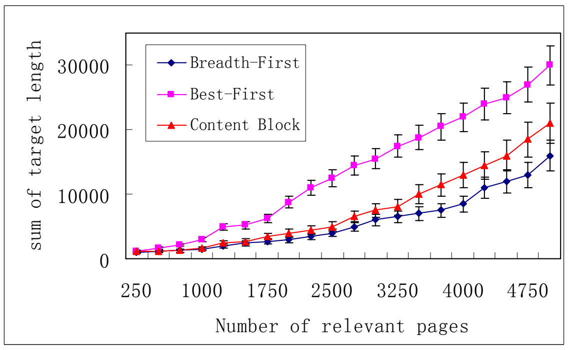

Fig. 7: Sum of target length of each topic web pages.

Fig. 5 shows that Breath-first method has not made any judgment for relativity of URLs. So many irrelevant pages have downloaded. The capability of crawler download relevant pages depends entirely on aggregation of crawled pages. Best-First methods based on relevance of the whole page to determine the priority values of URLs. Because of the influence of noisy and multi-topic, it is not high for obtain web pages of topic relevance. Content Block Method have overcome these shortcomings. Without affecting the value of the harvest rate, we try to make the highest rate of target recall.

Acknowledgement

This work is partially supported by National Nature Science Foundation of China (Grant No. 60373099). And 2006's Nature Science Foundation for Young Scholars of Northeast Normal University (Grant No. 20061005).

References

[1] P. De Bra, G. Houben, Y. Kornatzky and R. Post, Information Retrieval in Distributed Hypertexts, *Prof. of the 4th Int'l Conf. Intelligent Multimedia Information Retrieval Systems and Management (RIAO 94), Center of High Int'l Studies of Documentary Information Retrieval(CID)*, pp. 481-491,1994.

[2] M. Hersovici, A Heydon and M. Mitzenmacher, The SharkSearch Algorithm—An Application: Tailored Web Site Mapping Computer, *Networks and ISDN Systems*, 30:317-326, 1998.

[3] D. Bergmark, Carl Lagoze and Alex Sbityakov, Focused Crawls, Tunneling, and Digital Libraries, *Prof. of the 6th European Conference on Research and Advanced Technology for Digital Libraries*, pp.91-106, 2002.

[4] G. Pant, P. Srinivansan and F. Menczer, Exploration versus Exploitation in Topic Driven Crawlers, *Prof. of WWW-02 Workshop on Web Dynamics*, 2002.

[5] Z. Bar-Yossef and S. Rajagopalan, Template Detcetion via Data Mining and its Applications, *Prof. of the 11th International World Wide Web Conference*, pp.580-591, 2002

[6] S. Chakrabarti, Integrating the document object model with hyperlinks for enhanced topic distillation and information extraction, *Prof. of the 10th International World Wide Web Conference*, pp.211-220, 2001

[7] http://dmoz.org/.