Priority Queue Based Estimation of Importance of Web Pages for Web Crawlers

Mohammed Rashad Baker*, M. Ali Akcayol
Computer Engineering Department, Gazi University, Ankara, Turkey.

* Corresponding author. Email: mr_baker@live.com
Manuscript submitted February 2, 2017; accepted May 15, 2017.
doi: 10.17706/ijcee.2017.9.1.330-342

Abstract: There are hundreds of new web pages that are added daily to web directories. Web crawlers are developing over the same time of web pages growing up rapidly. Thus, the need for an efficient web crawler that deals with most of the web pages. Most of the web crawlers do not have the ability to visit and parse pages using URLs. In this study, a new web crawler algorithm has been developed using the priority queue. URLs, in crawled web pages, have been divided into inter domain links and intra domain links. The algorithm sets weight to these hyperlinks according to the type of links and stores links in the priority queue. Experimental results show that the developed algorithm gives a well crawled performance against unreached crawled web pages. In addition, the developed algorithm has a good capability to eliminate duplicated URLs.

Key words: Web crawler, page importance, link priority, priority queue.

1. Introduction

Getting access through huge information around the world is becoming more important than ever [1]. Therefore, the importance of achieving the desired information in the right way will gradually increase in the upcoming years. In order to collect all/most of the web pages in the Internet, the need of web robots occurred. It is becoming the most common tool used either to achieve certain web pages or to gathers specific information from a web page in any given subject.

A search engine is a tool used to search for content on the Internet with a specified user query. Search engines list almost all of the web sites in the world. These lists can be categorized, and provide a quick access to information that is asked by the users. There are hundreds of new web pages added daily to web directories [2]. Academic researches show how it is important to prioritize finding good or important pages and going through them in a fast way over discovering less important web pages.

Web crawlers have the ability to visit all web pages on the Internet to classify and index the existing and new web pages. The web crawler agents simply send HTTP requests for web pages that exist on other hosts. After HTTP connection, web crawler starts to visit a specific connected web page and extracts all hyperlinks or other contents of the web page. Then; stores its textual summarization in order to use it after that for indexing the contents. The crawler then parses those web pages to find new URLs.

Most of existed web Crawlers agents do not have the ability to visit and parse web pages. The reason is that the network bandwidth is very expensive [3]. Another reason is that there may be a limitation in storage where the crawler tries to store the data in a disk [4].

In this paper, a new web crawler algorithm has been developed based on priority queue for estimating the importance of web pages. In this work, URLs have been divided into two categories as intra links and inter links. The weight is set for URLs according to their types.

2. Related Works

There are many web crawler algorithms developed to manage web sites. Fish algorithm is developed by De Bra *et al.* [5]. This algorithm works in a similar way of catching a fish. According to this approach, the group of fishes moves towards the food. The group that is not near the food will die. Each crawler agent explores "feeder" areas which contain more "food", i.e. relevant documents, and possibly "dry" directions with no relevant findings. Although fish algorithm is easy to apply, it has some limitations such us connectivity-score distribution which is given 1 to each connected node and gave 0 or 0.5 discrete calculations.

The Shark-Search method [6] suggests using Vector Space Model (VSM) [7] as a technique to select page priority from crawl candidate pages. To determine the value of a node, it will go into web page content, anchor text, the text surrounding the URLs and the priority mount for parent URLs. The difference between shark and fish algorithms is that shark algorithm uses fuzzy logic technique for distributing scores among the URLs while fish algorithm uses binary scoring to sort URLs. Another difference is that shark search algorithm uses Vector Space Model (VSM) for searching while fish search algorithm uses regular expressions for searching inside web documents.

To determine high quality web pages to crawl, Cho *et al.* [8], presents a new connectivity-based criteria using Breadth-first algorithm. Cho et al, use four different algorithms (Breadth-first, backlink count, Page Rank, and Random algorithm) to crawl a single domain (Stanford.edu) web site. The experimental results showed the high ranked web pages are crawled before the low ranked pages. A Partial Page Rank algorithm will be the best choice to get better results. The next best algorithm is Breadth-first, then Backlink-count will be fine in uses as a web crawler algorithm, besides, it is shown that to find high page rank web pages, Breadth-first search algorithm will be the recommended one to get better results.

Najork and Wiener have applied a real crawler over 328 million of web pages [9]. To crawl web pages Najork and Wiener suggest using the connectivity-based link as criteria to get priority in web pages crawling. As a result, they showed that the connectivity-based criteria is getting better results than other known criteria like popularity of pages and content analyzing. The reason is the connectivity-based criteria is easy to use and get fast results without needing any extra information. As experimental results, Mercator [10] is used to crawl and download web pages. Also it uses Connectivity Server [11] in order to reach URLs faster inside downloaded web pages.

Other research in Breath-first algorithm comes from Yates *et al.* [12]. They used Page Rank as criteria to test Breadth-first, Back link-count, Batch-Page Rank, Partial-Page Rank, OPIC and Larger-sites-first algorithms. The experimental results show that Breadth-first algorithm is a good strategy to get the first 20%-30% of web pages at the beginning of crawling the Web. Depending to their results, the performance of Breadth-first algorithm will be lower after a couple of crawling days. This happens because of the large number of URLs of other pages that points to these web pages, and then the quality average for crawled pages is getting down gradually day after day.

Abiteboul *et al.* [13] introduced OPIC (On-line Page Importance Computation) as new crawling strategy algorithm. In OPIC, each page will be given a value to start with “cash”. Then it is starting to distribute it equally to all pages which is connected to and calculating all these mounts of “cash” to get the page score. Beside “cash” mount, there is another mount in OPIC algorithm named “history”. The mount of “history” is used as a memory of the pages, OPIC uses “history” mount to get the mount to the visits page from the start

of crawling process to reach the last crawled page. Although OPIC algorithm is similar to Page Rank algorithm in calculating the score of each web page, it is faster and done using single step. Such process occurs because OPIC crawling algorithm is downloading the pages with high cash mount. An OPIC crawling algorithm tries to download first pages in the crawling frontier with higher value of "cash". According to the OPIC crawling algorithm, the web pages may be downloaded several times depending on the page issue and this will affect and increase crawling time.

Zareh Bidoki *et al.* introduces smart Crawling Algorithm depends on Reinforcement learning algorithm (FICA) [14]. In FICA crawling algorithm, the priority of crawling web pages is depends on a concept named logarithmic distance between the links. The logarithmic distance between the links (Link-Connectivity) as a criteria, similar to Breadth-first algorithm determines which web page to be crawled next. FICA algorithm is uses less resource with less complex to crawl web pages and rank the web pages while crawling it on-line. Zareh Bidoki *et al.* tries to develop FICA algorithm and purpose FICA+ [15] algorithm. FICA+ algorithm is derivate from the FICA algorithm by using of backlink calculation and the specification of Breadth-first search algorithm. Zareh Bidoki *et al.* used University of California, Berkeley as a database source to test their algorithms. The goal was to use a FICA+ algorithm with Breadth-first, Backlink count and Partial PageRank, OPIC and FICA algorithms. The other goal was to determine which algorithm is getting important web pages with a higher PageRank than the others. The result showed that FICA+ got better results than the other crawling algorithms.

C. Wang, *et al.* [16] introduced OTIE (On-Line Topical Importance Estimation) algorithm to manage the results and improve it from the frontier in the crawling process. The OTIE algorithm arranges the URLs inside the frontier by using a combination of Link-based and Content-based criteria analyzer. OTIE algorithm is a focused crawling algorithm that is used for general purpose. They used Apache Nutch [17] for easy implementation and test. There is similarity between OTIE algorithm and OPIC algorithm. The similarity is that OTIE contains "cash" that is transferred between web pages as it exists in OPIC algorithm. Depending on previous works [18], [19], Link-based method in a focused crawler algorithm gets low performance in downloading web pages. To solve this issue they used bias, the cash mount distributes in OTIE in order to prefer on-topic web pages and to push down off-topic web pages.

3. Methodology

According to the information obtained as a result of the literature, a successful web crawler algorithm must parse all links founded within a web page. Besides the relationship between the links, it increases the strength of the crawling operation. The other element of a successful web crawler algorithm design is to measure the importance of the link. To do that, the crawler should parse all the links from a given seed link and divide them into two kinds of links. Out-Link which is the link founded in the extracted seed link and refers to other links outside the seed's domain (Inter-Domain). In-Link is the link founded in the extracted seed link and refers to the same domain which the seed URL belongs to (Intra-Domain). URL frontier is a data structure that contains all of the URLs intended to be downloaded. Most of web crawler applications are using frontier Breadth-First with FIFO technique to select the URL seed to start crawling.

In data structure concepts: FIFO technique happens when elements are removed from the rear end of the queue in order to insert elements into the front end of the queue. This issue is different and more complex in web crawler. It starts sending a large number of HTTP requests to a server and this makes it a complex structure. In this context, to be held in parallel with multiple HTTP requests, it should not go back to the head of the queue it should be done in a parallel way. To use this structure in a developed algorithm, a Priority Queue structure is proposed to work as a frontier of the developed web crawler algorithm. All parsed URLs are placed inside Priority Queue according to their gained scores. The selection of a next seed

URL will be determined according to the highest score received by the links.

To avoid parsed URLs to be waiting for a long time inside the developed Frontier, a time control mechanism has been developed. Time control mechanism will control URL waiting time inside Frontier in every new parsed URLs insertion process. Using this mechanism every URL inside Frontier will wait for specific time to be crawled. If this URL reaches this time it will be dropped from Frontier structure. Thus, to crawl a URL, it should be under the specific time to be set as maximum waiting time otherwise it will be dropped from Frontier.

It is known that the number of URLs that exist in any seed URL is not determined. Taking a process with only one URL in each crawler process will slow up program performance. Therefore the need to use multi-threading process to speed up dequeue operation from developed frontier Priority Queue structure is very important issue.

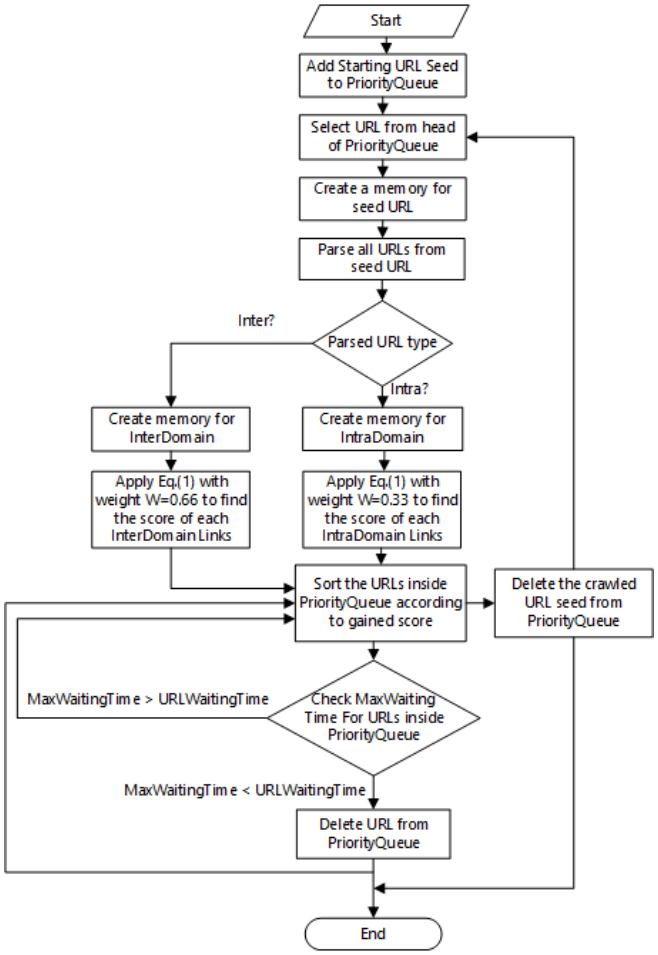

Fig. 1. The flowchart of developed web crawler algorithm.

The flowchart of developed web crawler algorithm is given in Fig. 1. After selecting a web page as seed URL, the crawler will clear unwanted tags and try to categorize all URLs inside it into Intra Domain and Inter Domain. Intra Domain, is the group of links founded in the extracted seed link. It refers to the same domain which the seed URL belongs. Inter Domain is the group of links founded in the extracted seed link and refers to other links outside the seed's domain. To measure the importance of each parsed web page, they are all scored according to the type of the page.

In this work, the focus is on the outgoing links (Inter Domain) contained in a seed URL more than in-links (Intra Domain). The reason is to avoid link-loops inside on domain. It is believed that new links from

different web pages will lead us to non-stop crawling process and the algorithm will continue finding new domains to be crawled. Because of that, amount with 2/3 is given as a weight to Inter Domains' link. In the same time a mount with 1/3 is given as weight to Intra Domain.

According to each category of links and its specification of each group of links, the general form of the equation (1) will be applied:

$$LS = \frac{\sum(\alpha)-\beta}{\sum \alpha} * \theta \quad (1)$$

where *LS* is Link Score, α is the sum of Inter Link Size and Intra Link Size. β is the minimum value of link size between Inter Domain and Intra Domain links and θ is assigned weight for each category of links. Fig. 2 shows the function of developed web crawler algorithm and how it works.

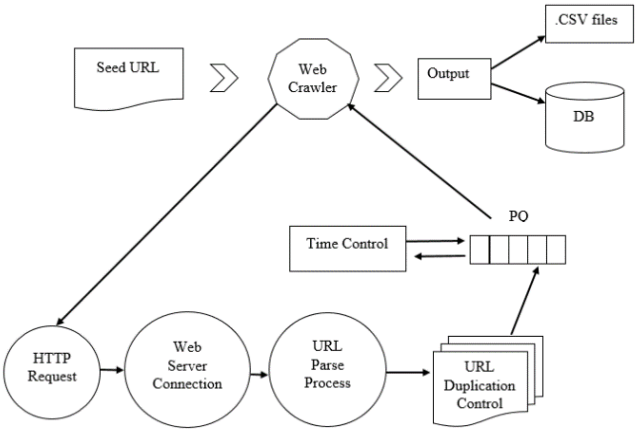

Fig. 2. The developed crawling algorithm and its functions.

The pseudo code of the proposed crawling algorithm is shown in Algorithm 1. In the algorithm, PQ is the Priority Queue containing URLs parsed from seed URL. X refers to the weight mount of Inter domain and Intra domain. Here X is selectable value for both of Inter and Intra domain weights. To determine the score of Inter domain0.66 is used as a value for X, and for Intra domain X=0.33. M, is the created memory for the seed URL to store the weights and record the scores. LS the score of the links that can be applied to determine the score of each Inter and Intra domain links.

```
Input: seed_URL,
Initialise PQ,
X: Inter domain wieght or Intra domain weight,
enqueue (seed_URL)

    Initialise M for seed_URL,
    for (1 to all seed_URL links)
        Parse seed_URL
            if (Parsed URLs belongs to different domain)
            then (add links to Inter_Domain)
            else (add links to Intra_Domain)
    if Inter_Domain>Intra_Domain
    then LS = ((all seed_URL links - Intra_Domain)/all seed_URL links) * X
    else LS = ((all seed_URL links - Inter_Domain)/all seed_URL links) * X

enqueue(Inter_Domain, Intra_Domain)
    for (1 to PQ)
    if (MaxWaitingTime < links)
    then delete it
dequeue(seed_URL),
```

Algorithm 1: The pseudo code of the proposed crawling algorithm.

According to Fig. 3, when the crawler starts to process seed URL (A), it is starts to parse its URLs. These

URLs are: (B, C, D, E, F, G and H). Then it starts to categorize the URLs into Inter and Intra domains. Here five URLs (B, C, D, E and F) are Inter domain of seed (A) that refer to different domains, two URLs (G and H) are Intra domains for seed (A) that refer to the same seed URL domains. Regarding the Equation (1), to determine the Inter Domain Links Score LS_{inter} the value of α which is mentioned before should be known before summing up Inter Link Size $\propto_{inter}$ and Intra Link Size $\propto_{intra}$. If $\propto_{inter} > \propto_{intra}$, the value of β_{intra}= 2. To find the scores of each Inter Domain, links use the value of =0.66 as following Equation (2):

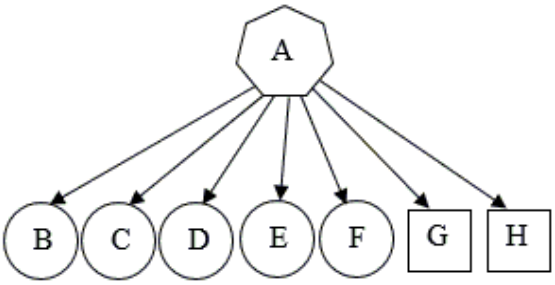

Fig. 3. Sample of web crawler tree.

$$LS_{inter} = \frac{\sum(\alpha_{inter} + \alpha_{intra}) - \beta_{intra}}{\sum(\alpha_{inter} + \alpha_{intra})} * 0.66 \quad (2)$$

Using the Equation (3) with changing the value of θ to 0.33, will help to find the score of Intra Domain Links:

$$LS_{intra} = \frac{\sum(\alpha_{inter} + \alpha_{intra}) - \beta_{intra}}{\sum(\alpha_{inter} + \alpha_{intra})} * 0.33 \quad (3)$$

A Priority Queue structure will be built at this stage. These URLs in the queue will be stored according to the gained score. Inside the priority queue structure, the URLs will be sorted descending from the highest score URL to the lowest score. The next seed URL will be selected from the queue according to the highest URL score that exists in the tail of the queue. Every selected URLs will have amount of weight (W= 1) to start with. When crawler starts the process, the seed URL will be deleted and add the extracted URLs to the Priority Queue according to its gained score. The crawling process will not stop working until there are no more URLs found in the Priority Queue.

During crawling process, Priority Queue structures will be under control by developing the time control mechanism to examine all URLs waiting inside the Frontier. All scored URLs will be stored inside the Frontier with their insertion time. By the time the Frontier structure will be getting bigger. Thus, to avoid memory heap error, every URL will be dropped if it reaches the maximum waiting time while waiting for its turn to be crawled. Therefore, crawling process will go through the Frontier and select next URL seed according to its score and waiting time properties. After parsing all its URLs, the selected seed URL will be deleted from the Frontier but will be kept by applied duplication control mechanism.

4. Experiment Results

According to literature research results, there are many URLs used to perform as different seeds during web crawler process. In the developed web crawler algorithm, *http://www.stanford.edu* is used as a dataset used in [8] too. Besides, *http://www.wikipedia.org* the world's biggest online encyclopedia is used. The 3rd seed that was *http://www.yok.gov.tr* a Turkish governmental web site that contains important information about Turkish universities and also other Turkish governmental web sites.

After running the developed web crawler algorithm, FIFO Queue techniques are used to select seeds from dataset to start the crawling process. Using Priority Queue techniques parsed URLs were sorted inside it

according to the scores they have gained. With the similarity to FIFO techniques, the next URL seed will be selected from Priority Queue structure will be the URL that gained a higher score than other URLs. The number of URLs exists in any seed URL is unknown and dequeue only one URL each crawling process will slow up program performance, so multi-threading process is used to speed up the dequeue operation from developed frontier Priority Queue structure.

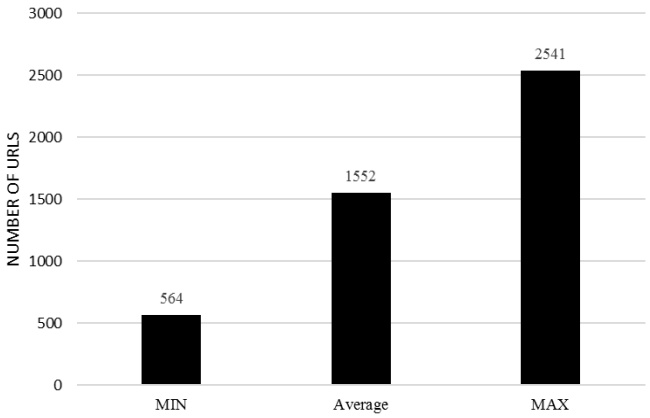

Fig. 4. MAX, Average and MIN crawled web pages during crawling process.

Overall, the number of URLs waiting at the frontier structure of the web crawler application consists of hundreds of millions of URLs. Therefore, URLs must be saved to disk. The reason for assigning scores for parsing links is to make the developed algorithm to decide which link will be crawled earlier than other links.

As experimental results, the maximum, average and minimum speed of crawled web pages within a given time segment is demonstrated. Fig. 4, shows the performance of the developed crawling algorithm by getting the minimum, average and maximum number of crawled web pages during crawling operation.

As shown in Fig. 4, there is uncrawled data during this experimental. The reason is that to crawl a given seed URL, the program will give a connection timeout to connect to the HTTP protocol, some web sites take time to respond to this HTTP connection requests and some of web sites will not respond to this request, besides, to crawl a web page the program will load the whole content of a given page into memory before starting to parse it and extract its links. All these reasons can affect crawling speed performance.

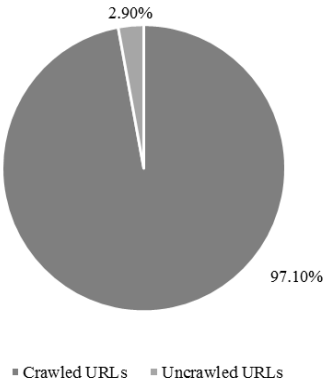

Fig. 5. Percentages of crawled web pages vs. uncrawled web pages.

Fig. 5 is showing the percentages of crawled web pages is 97.10% of whole crawled web pages while uncrawled web pages form 2.90% of the whole crawled web pages during the crawling process for many reasons. Table 1 shows uncrawled web pages errors' types and their percentages.

Table 1. General Error Types for Uncrawled URLs

Error Type	Percentages
404	30.97%
403	19.67%
503	5.83%
Other HTTP Status Codes	8.38%
Unknown Supported MIME Type	21.68%
Unknown Errors	10.75%
Time Out Errors	2.73%

To analyze uncrawled web pages errors types in more details, error types has been divided into two parts. HTTP status types and other errors types. As shown in Table 2, uncrawled web pages error that comes from HTTP status types was 64.85% from total uncrawled URLs while 35.15% of total uncrawled URLs was coming from other errors types during the crawling process.

Table 2. Error Types for Uncrawled URLs

Error Type	Percentages
HTTP Status Codes	64.85%
Other Errors	35.15%

Analyzing HTTP status errors in more details Table 3 shows that 47.75% HTTP status errors belong to HTTP state error (404) 52.25% makes other HTTP error states.

Table 3. HTTP Status Error Code for Uncrawled Web Pages

HTTP Status Error	Percentages
404	47.75%
403	30.34%
503	8.99%
Other HTTP Status Codes	12.92%

While this developed algorithm is depending on high priority to crawl URL links, Fig. 6 shows minimum, average and maximum time for waiting for a single link inside the Priority Queue.

According to developed algorithm and waiting time control mechanism, a single URL will be waiting inside Priority Queue structure for maximum 30 minutes. Regarding developed mechanism, the lowest score gained by parsed URLs will be dropped from Crawling Frontier to avoid filling the Queue without crawling.

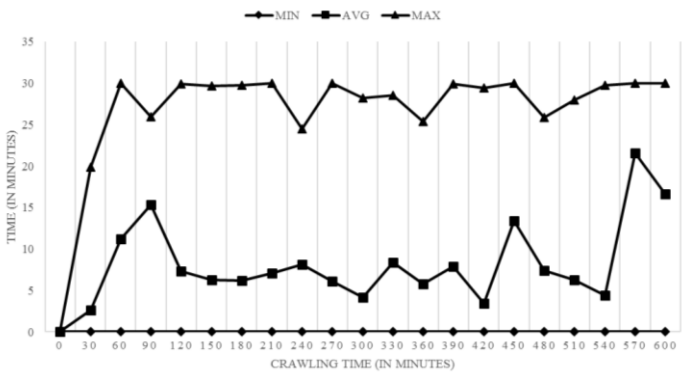

Fig. 6. MAX, Average and MIN waiting time inside Priority Queue every 30 minutes of crawling process.

Fig. 7 shows the number of URLs that had been dropped. The number of dropped URLs after 30 minutes from the start of crawling process are more than other time duration. The reason is that the developed Priority Queue control mechanism starts after 30 minutes from crawling process. Thus the waiting URLs inside Priority Queue will get its normal forum after the 60th minute from the start of crawling process. Using the developed time control mechanism, every URL will be checked to measure its waiting time inside Frontier. This new mechanism will be activated every time a new parsed URL passes inside Frontier structure.

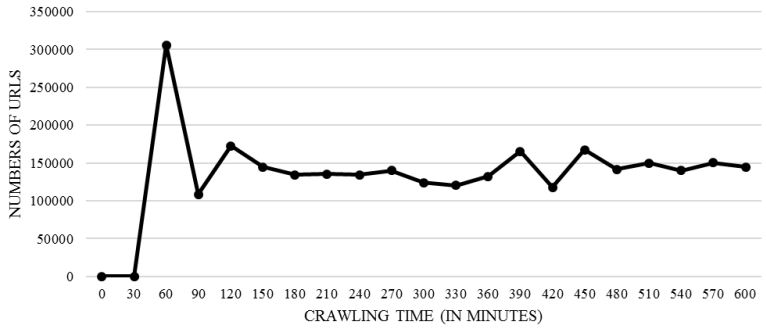

Fig. 7. The minimum, average and maximum dropped URLs from Priority Queue.

By using the developed time control mechanism, the URLs will not be kept more than a specific time inside the Priority Queue. The URLs that had waiting more than the specific time inside the Priority Queue will be dropped. Regarding this mechanism, the maximum waiting to time for any URL will be under the specific time that has been set in time control mechanism.

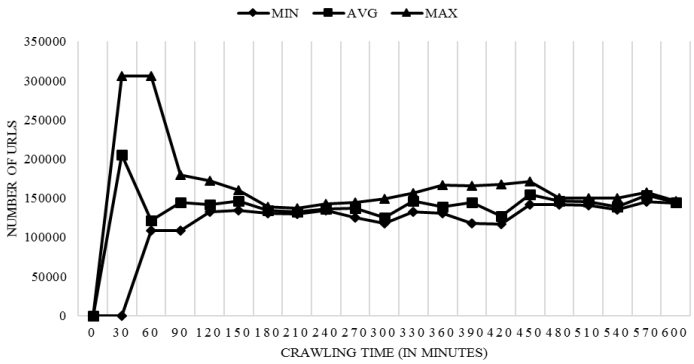

Fig. 8. Fill of Priority Queue during crawling process.

Fig. 8 shows that there are more 300000 URLs inside the Priority Queue in the first 60 minutes from the start of crawling process. But regarding the developed time control mechanism, the Priority Queue structure is getting to be more stable and optimize the URLs inside it (100000-200000 URLs) during the crawling process.

To measure the speed of crawling process, an analyze of minimum, average and maximum crawling speed has been done. Fig. 9 shows the speed of crawled web pages, for the given 30 minutes of crawling time. This speed is referring to the time needed to parse all URLs from current seed.

The time duration (240-360 minutes) of crawling process shows the maximum time between 1 to 1.5 minutes needs to parse all URLs from current parent URL. While in the 480th minute of crawling process, the maximum time is 12 minutes needed to parse all URLs from current parent URL. The reason for that difference is that the parent URL (seed URL) in time duration (240-360 minutes) contains fewer child URLs

(parsed URLs) so the crawling speed will be faster than the parent URL that contains more child URLs (parsed URLs) and leads to slowness in the crawling speed.

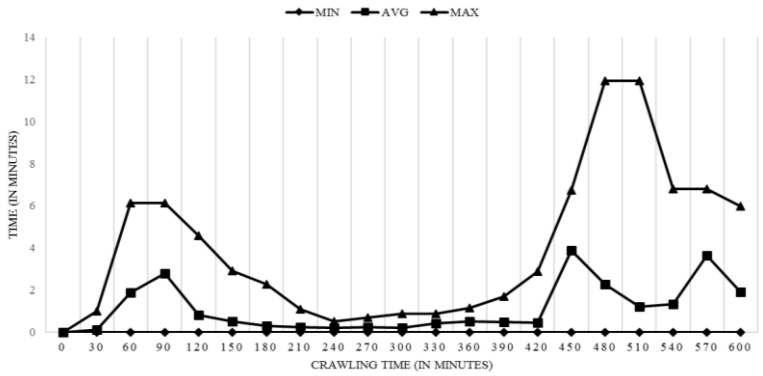

Fig. 9. MAX, average and MIN speed of crawling during crawling process.

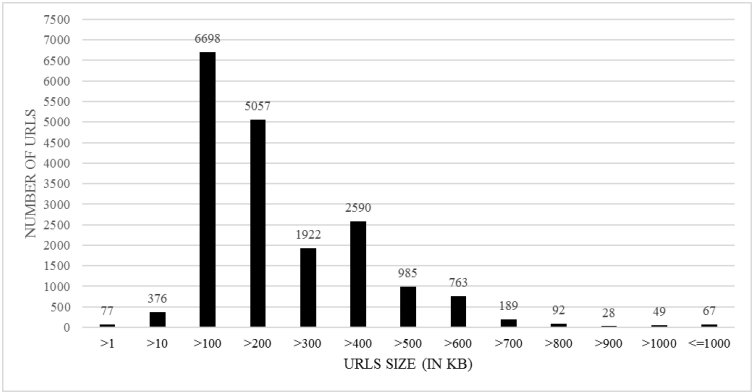

Fig. 10. Size of crawled web pages.

Another factor that could be affecting crawling process speed is web pages size. Fig. 10 shows that although the maximum size of crawled web pages was 100 KB but there were web pages with size 500 KB too. As a result, getting bigger web page size leads to slowness in crawling process.

In this developed web crawler algorithm as mentioned before, more attention was given to develop a crawler algorithm with Inter Domain URL. The reason for that is to avoid link-loops inside the domain and that new links from different web pages will lead us to non-stop crawling process and the algorithm will continue to find new domains to be crawled. Because of that, assigned 2/3 of seed URL's weight was assigned to Inter Domains' links and 1/3 of given weight was assigned to Intra Domain. Fig. 11 shows Priority Queue's contents of Inter and Intra domain links for every 30 minutes of crawling process.

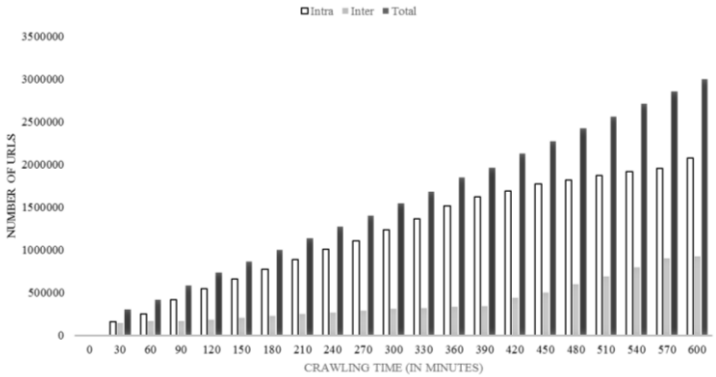

Fig. 11. Size of crawled web pages.

Table 4. Most Viewed Country's Domain Name Distributions

Domain Name	Percentages
.ca	49.25%
.uk	16.84%
.it	7.46%
.be	5.61%
.tr	4.83%
.de	3.41%
.nz	2.63%
.jp	1.49%
.gl	1.07%
.bg	0.92%
Others	6.47%

After separating these web pages to domain names, the most crawled domain name distribution was determined. Table 4 shows the most viewed country's domain name distribution.

The table shows that although the crawling process starts with .edu, .org and .gov domain names, but during crawling process, the most crawled web pages was a country domain name like .ca Canada, .UK United Kingdom and .it Italy.

From total crawled web pages as shown in Table 5 the most crawled domain name was .com extension. According to developed web crawler algorithm, giving Inter domain URLs more importance to be crawled over Intra domain URLs lead the crawling process to crawl new web pages from different domains.

Table 5. Most Crawled Domain Name

Domain Name	Percentages
.com	95.23%
.jobs	2.29%
.org	1.69%
.net	0.43%
.net	0.12%
.info	0.11%
other	0.12%

The web pages on the Internet are connected to each other. While some of the web pages are connected to new web pages, new web pages that are pointing (connecting) to the same web pages will appear, leading to duplication in web pages, which means these web pages were already visited/crawled before.

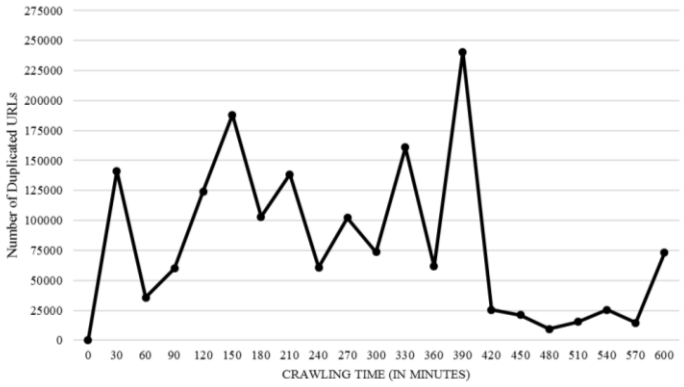

Fig. 12. Duplicated URLs counts for every 30 minutes of crawling process.

Fig. 12 shows the duplication of URLs that had been detected in crawling process for each 30 minutes. To avoid URL duplication while crawling process, a duplication control process starts inside the Priority Queue to check whether the URL was processed before or not by using the advantage of Linked Hash Set inside it. Using Linked Hash Set as a control gate will double check the parsed URLs before adding them to the Priority Queue structure to see while it's crawled before or not. If the candidate parsed URL was not crawled before, then the URL will be added to the Priority Queue, hence waiting its turn according to their scores.

5. Conclusion

In this study, a new algorithm has been introduced to crawl web pages. The developed algorithm is based on using priority queue as seed frontier and dividing crawled URLs into inter and intra links. The developed algorithm focuses on adding crawled inter domain links to the frontier more than focusing on intra domain links. The reason for that is to avoid link-loops inside a domain. As a result, this leads to discovering new links on different hosts and domains. The experimental results show that the developed crawler algorithm gives a good crawled performance against unreached crawled web pages. Besides the developed algorithm also has the capability to eliminate duplicate URLs.

References

[1] Brin, S., & Page, L. (2012). Reprint of: The anatomy of a large-scale hypertextual web search engine. *Computer Networks*, *56(18),* 3825-3833.

[2] Lewandowski, D. (2008). A three-year study on the freshness of web search engine databases. *Journal of Information Science, 34(6),* 817-831.

[3] Ali, H. (2008). *Effective Web Crawlers.* PhD. dissertation thesis, School of Computer Science and Information Technology, Science, Engineering, and Technology Portfolio, RMIT Univ., Melbourne, Victoria.

[4] Cho, J. (2001). *Crawling the Web: Discovery and Maintenance of Large-Scale Web Data.* Ph.D. Dissertation thesis, Dep. of Computer Science, Stanford University.

[5] De Bra, P., Houben, G. J., Kornatzky, Y., & Post, R. (October 1994). Information retrieval in distributed hypertexts. *Intelligent Multimedia Information Retrieval Systems and Management*, *1,* 481-491.

[6] Hersovici, M., Jacovi, M., Maarek, Y. S., Pelleg, D., Shtalhaim, M., & Ur, S. (1998). The shark-search algorithm. An application: Tailored Web site mapping. *Computer Networks and ISDN Systems*, *30(1),* 317-326.

[7] Salton, G., Wong, A., & Yang, C. S. (1975). A vector space model for automatic indexing. *Communications of the ACM*, *18(11),* 613-620.

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

[9] Najork, M., & Wiener, J. L. (April 2001). Breadth-First crawling yields high-quality pages. *Proceedings of the 10th International Conference on World Wide Web* (pp. 114-118). ACM.

[10] Heydon, A., & Najork, M. (1999). Mercator: A scalable, extensible web crawler. *World Wide Web*, *2(4),* 219-229.

[11] Bharat, K., Broder, A., Henzinger, M., Kumar, P., & Venkatasubramanian, S. (1998). The connectivity server: Fast access to linkage information on the web. *Computer Networks and ISDN Systems*, *30(1-7),* 469-477.

[12] Baeza-Yates, R., Castillo, C., Marin, M., & Rodriguez, A. (May 2005). Crawling a country: Better strategies than breadth-first for web page ordering. *Proceedings of Special Interest Tracks and Posters of the 14th*

International Conference on World Wide Web (pp. 864-872). ACM.

[13] Abiteboul, S., Preda, M., & Cobena, G. (May 2003). Adaptive on-line page importance computation. *Proceedings of the 12th International Conference on World Wide Web* (pp. 280-290). ACM.

[14] Bidoki, A. M. Z., Yazdani, N., & Ghodsnia, P. (2009). FICA: A novel intelligent crawling algorithm based on reinforcement learning. *Web Intelligence and Agent Systems: An International Journal*, *7(4),* 363-373.

[15] Golshani, M. A., Derhami, V., & ZarehBidoki, A. (December 2011). A novel crawling algorithm for web pages. *Proceedings of Asia Information Retrieval Symposium* (pp. 263-272). Springer Berlin Heidelberg.

[16] Wang, C., Guan, Z. Y., Chen, C., Bu, J. J., Wang, J. F., & Lin, H. Z. (2009). On-line topical importance estimation: An effective focused crawling algorithm combining link and content analysis. *Journal of Zhejiang University-Science A*, *10(8),* 1114-1124.

[17] Nutch, A. Retrieved February 1, 2017 from http://nutch. apache.org

[18] Menczer, F., Pant, G., Srinivasan, P., & Ruiz, M. E. (September 2001). Evaluating topic-driven web crawlers. *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 241-249). ACM.

[19] Chau, M., & Chen, H. (2003). Comparison of three vertical search spiders. *Computer, 36(5),* 56-62.

Mohammed Rashad Baker received the B.S degree in software engineering techniques from Collage of Technology–Kirkuk, Iraq in 2005. He received the M.Sc degree in computer engineering from Gazi University, Ankara, Turkey in 2009. And currently he is a Ph.D candidate at Gazi University, Faculty of Engineering, Department of Computer Engineering. His research interests include web mining, web crawler and web ranking, web and mobil wireless mesh networks, wireless network routing protocols.

M. Ali Akcayol received the B.S degree in electronics and computer systems education from Gazi University in 1993. He received the M.Sc degree and Ph.D degree in Institute of Science and Technology from Gazi University, Ankara, Turkey in 1998 and 2001, respectively. His research interests include mobile wireless networking, web technologies, web mining, bigdata, cloud computing, artificial intelligence, intelligent optimization techniques and hybrid intelligent systems.