*ICIC Express Letters* ICIC International ©2021 ISSN 1881-803X
Volume **15**, Number **12**, December **2021** pp. 1345–1354

# FOCUSED WEB CRAWLER USING GENETIC ALGORITHMS AND SYMBIOTIC ORGANISM SEARCH

Alfonsus Wilson[1], Hendrawan Armanto[2], Gunawan[2] and Jefri Tanwijaya[1]

[1]Department of Computer Science
Bina Nusantara University
Jl. K. H. Syahdan No. 9, Kemanggisan Palmerah, Jakarta 11480, Indonesia
{ alfonsus.wilson; jefri.tanwijaya }@binus.ac.id

[2]Department of Informatics Engineering
Institut Sains dan Teknologi Terpadu Surabaya
Ngagel Jaya Tengah No. 73-77, Surabaya 60284, Indonesia
{ hendrawan; gunawan }@stts.edu

Received February 2021; accepted May 2021

Abstract. *Internet saves a lot of information and has expanding rapidly. Crawler is commonly used to surf the Internet to collect news that is linked with a URL. Focused crawler can selectively collect relevant news according to the topic given. This paper proposes a focused web crawler using GA with a query modification using the SOS algorithm. Focused crawler was tested on 5 topics with different domains each and with download limit of 400 pages. The results showed a harvest rate of 80.66%. With the same time, focused crawler can collect more relevant news than BFS crawler. So, it can be said that this focused crawler can collect news that is relevant to the topic effectively.*
**Keywords:** Focused crawler, Genetic algorithm, Information retrieval, Okapi BM25, Cosine similarity

1. **Introduction.** Since the founding of the Internet in 1993, the World Wide Web has grown exponentially in terms of information and capacity. Currently, the number of websites has reached more than 1.6 million websites worldwide. Thus, the search engine is a very beneficial tool in helping users to find information and their needs on the Internet. Web crawler is the key of all search engines. The crawler visits a page and follows the hyperlinks found on that page to go to another page, which then saved for further indexing. However, due to the growing capacity of the Internet, it is increasingly difficult for search engines to provide effective information for end-users.

Several studies that studied focused crawler have identified several issues that cannot be resolved with regular focused crawling algorithms. Firstly, a website can be linked through co-citation, which makes crawler miss the domain of the web page. Second, relevant web pages in a domain can have different Web Communities linked by irrelevant web pages. Third, relevant web pages on the same domain can also be found without the URL pointing to that web page.

The use of metaheuristic algorithms in solving complex problems has been recognized and developed by many scientists. Various kinds of books and papers have been written to support the science of metaheuristic algorithm. There have been many metaheuristic algorithms programs that were continually produced.

The problem that often occurs in focused crawler designs, especially those using local searching architecture, is the limited number of pages that can be visited. In this algorithm, crawler can find web pages with high relevance only on URLs that are close to the seed URL (neighbor URL). Problems like this are often referred to as local optimal.

DOI: 10.24507/icicel.15.12.1345

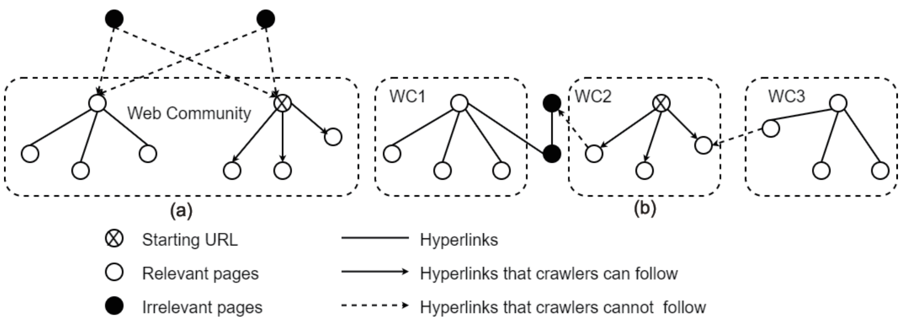


Figure 1. Problems that occur in local searching algorithm

Subsequent research discovers that local searching algorithms are not suitable for focused crawling purposes [1]. First, many pages that are potential to possess high relevance value are not directly linked to the seed URL, but through a co-citation link. The problem diagram on a local searching algorithm can be seen in Figure 1.

In Figure 1, (a) the crawler can pass co-cited linked pages, (b) the crawler rotates in that environment only. An example of the problem illustrated in image (b) is a relevant page separated by an irrelevant page, so that crawler cannot go to that page. Another problem is that sometimes, there are several links that lead to a relevant page, but there is no URL of the page pointing to the original page (b).

The rest of the paper is organized as follows. In Section 2, we discuss related work in focused crawler. In Section 3, we outlined the proposed working method of focused crawler using Genetic Algorithm (GA) along with the fitness and the problems representation. The proposed method results are analyzed and discussed in Section 4, and the result using different similarity function named cosine, weight table, and Okapi BM25. Finally, Section 5 concludes the paper.

## 2. **Related Works.**

Focused crawler was first introduced by Chakrabarti et al. in 1999 [1]. Menczer et al. created a topic-driven Web crawler [3] and compared the crawler with several other crawling strategies. Then, it was followed by Rennie and McCallum who employed reinforcement learning [4].

In addition, there are other strategies which measure the similarity of a page with a topic using metrics and sort URLs for the next iteration [5], as well as a learning scheme to identify the next URL that should be crawled [6]. Following the previous research is Liu et al., who used a clustering strategy for crawling [7].

Many researches discuss the URL ordering strategy. One of them is PageRank [8]. PageRank is a mechanism used by Google in its search engine. The basic point of PageRank is to judge a page P not only by how many pages go to P, but also how important they are. The PageRank concept then was developed by adding the similarity value to the topic keyword [9]. The URL in the frontier will be sorted based on the number of topic keywords on the previous page, and then the PageRank value will be calculated. Chau and Chen [10] compared the crawler with a neural network crawler to calculate its efficiency.

Another study proposed by Sun et al. [11] is using a hybrid strategy in focused crawling based on meta search, which was later developed by Shokouhi et al. [12] using the Gcrawler strategy with a genetic algorithm. In [15], a crawler is proposed with a subject-based strategy and link-based Web analysis simultaneously to search for news globally. Continued in the paper, a crawler using a reinforcement learning strategy and fuzzy clustering was created.

In addition, there are focused web crawlers that apply semantic methods, such as semantic focused crawler to helping mine web crimes based on these web ontologies [13].

Another example of the application of the semantic method is using the Knowledge Representation Scheme (KRS), where this approach with the help of KRS helps to get the relevant elements from a domain automatically through semantic analysis of the corpus input [14].

Genetic algorithms have been proven to be able to solve problems both linear and nonlinear by exploring all regions of state space and looking for promising new areas with mutation, crossover, and selection operations that are applied to each individual within the population [17]. A more thorough discussion can be found in books written by Davis [18], Goldberg and Holland [19], Holland [20], and Michalewicz [17].

To alleviate the problems of local searching algorithms, researchers have suggested several strategies. One of them is by using more starting URLs. However, composing a list of high-quality starting URLs is an expensive and time-consuming task.

The GA-crawler made in this paper has a slightly different approach. The creation of the seed-URL uses the help of search engines, so that crawler possesses quality seed-pages, regardless of the domain, without any input documents. The combined top results from multiple search engines had high coverage over the web, and by employing SOS (Symbiotic Organisms Search) in the mutation phase, the crawler can explore a wider area of exploration. This approach will eventually make the crawler jump over the unexplored area thoroughly and to avoid local optimal.

## 3. Proposed Focused Crawler Model.

In focused crawler problems, the selection of URLs that are right on target or relevant to a given topic needs to be resolved or optimized. However, to be able to implement genetic algorithms into a focused web crawler, some adjustments need to be made, especially in the representation, input and output sections, to the fitness function used. The GA-crawler architecture design can be seen in Figure 2.

### 3.1. Individual representation.

The query given by the user was thrown to the search engine to get at least 10 news URLs that are relevant to the query. In this phase, Google Search API is used as an intermediary medium to get search results on the Google search engine. The main purpose of using the Google search engine is to equalize the search algorithm for each domain. The URLs returned by the Google Search API will then be downloaded (fetched) and used as seed pages for the focused web crawler.

Individuals in a genetic algorithm are composed of several genes which are either an integer or a string. However, since the individual representations on this crawler are not integers or strings, the genes that compose a web page will be properties on that web page. The genes that make up an individual include

**URL:** The URL where the web page is downloaded will be stored as the genes of the individual. URL is required for data storage that is displayed as output.

**TFIDF MATRIX:** The tfidf matrix calculation for each document is needed to find the word with the highest tfidf value. This word will then be used to form the weight table.

**TEXT:** Text is the news or web's content pages that passed the preprocessing process. Thus gen is also for data storage that is displayed as output.

**TITLE:** Title text on every web page. The title page usually stores information that represents the content of the web page. This title will be stored and used as an additional fitness measure.

### 3.2. Fitness function.

Fitness function is a function used to assess an individual, how close the individual's quality is to the goal to be achieved. Because the individuals on the crawler are not composed of numbers or strings that can be divided into genes, a topic-specific weight table will be drawn up which refers to the initial population of crawler which will be followed by a fitness calculation.

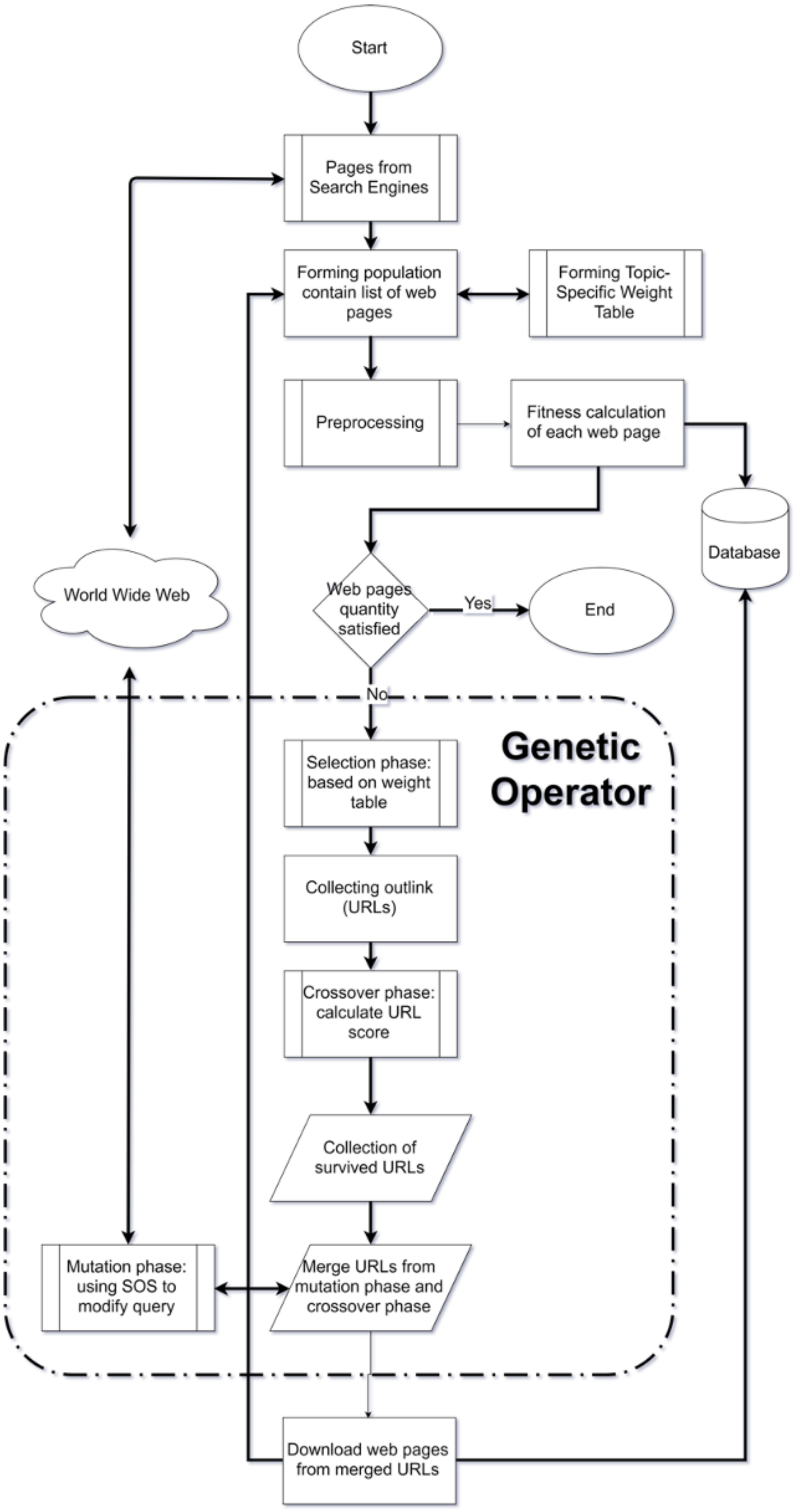

Start
Pages from Search Engines
Forming population contain list of web pages
Forming Topic-Specific Weight Table
Preprocessing
Fitness calculation of each web page
Database
World Wide Web
Web pages quantity satisfied
Yes
End
No
Genetic Operator
Selection phase: based on weight table
Collecting outlink (URLs)
Crossover phase: calculate URL score
Collection of survived URLs
Mutation phase: using SOS to modify query
Merge URLs from mutation phase and crossover phase
Download web pages from merged URLs

FIGURE 2. GA-crawler program architecture

3.2.1. *Making topic-specific weight table.* Making this weight table begins with the calculation of TF-DF in the initial population formed from the search engine. After obtaining the TF-DF value for each word in the corpus, this value is normalized using max normalization:

$$W(t) = \frac{w(t)}{\max(w)} \tag{1}$$

where $W(t)$ is the weight for $t$ and will be normalized by $\max(w)$ that is obtained from the highest TF-DF value from the $w$ word collection. This weight table will also be added with token from the preprocessed query with a weight of 1.

Table 1 shows an example of a weight table made from the query "hasil tes motogp Qatar". The compiled weight table will be expanded by adding new relevant terms as the crawler generation continues. Amongst the web pages downloaded by crawler, those with a fitness value greater than or equal to 0.7 are considered to be web pages that are relevant to the query. The keywords or terms with the highest TF-DF value from each of these web pages will be extracted to be included in the weight table with a weight value equal to the relevance value of the relevant web page.

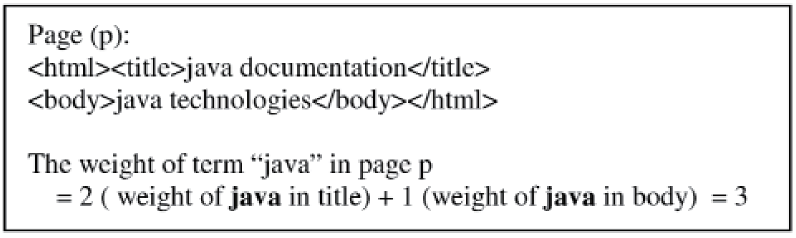
Table 1. Weight table example

| Term | Weight |
|---|---|
| hasil | 1.0 |
| tes | 1.0 |
| motogp | 1.0 |
| qatar | 1.0 |
| vinales | 0.630850 |
| spa | 0.930349 |

3.2.2. *Calculation of fitness value using cosine similarity.* The cosine similarity compared the real value of the weight table vector with the vector of each web page. The web page vector will be constructed from the terms in the weight table found on the web page being processed. The weight of the terms on the web page will be determined by the TF-DF value of each term. The same word in different positions has different information values. For example, a sentence in a news title will describe the news topic more than the content of the news itself. Therefore, the position weights will be divided into the following:

$$fkp = \begin{cases} 2, & \text{title text} \\ 1, & \text{common text} \end{cases} \tag{2}$$

$fkp$ is a weighted composition of $k$ keywords that are in different positions from the $p$ page. Thus, it can be concluded that $wkp$ or the total weight of each $k$ keyword on the $p$ page is the news title added to the $fkp$ news content. An example of calculating $wkp$ for $k =$ "java" can be seen in Figure 3.


Page (p):
<html><title>java documentation</title>
<body>java technologies</body></html>

The weight of term "java" in page p
= 2 ( weight of **java** in title) + 1 (weight of **java** in body) = 3

Figure 3. *wkp* calculation example

Assuming the TF-DF value for the word "java" is 1 in the document $p$, and the word "java" appears in the title and content of the news, the *wkp* value for the word "java" in the document "$p$" is 3.

After the weight vector on the document has been formed, then this vector will be calculated of its similarity to the topic weight table vector using the cosine similarity [21]. The cosine similarity formula will produce a value in the range 0-1 with a page value close to 1 which indicates that the page is relevant to the weight table vector, while a value close to 0 will indicate that the page is moving away from the weight table vector [15].

This fitness calculation does not rule out zero fitness or it can be said that an individual does not have the same word as the topic weight table (irrelevant news) which causes the individual's fitness value to be 0. To overcome this, and so the individual has a survival rate in the selection process, the fitness value of each individual after going through the cosine similarity will be added with a bias value of 0.05.

3.2.3. *Calculation of fitness value using Okapi BM25.* Just like the previous fitness calculations that used cosine similarity, Okapi BM25 (BM stands for Best Matching) is also a ranking function used by several search engines to sort documents based on their relevance to the given query [22]. In this paper, BM25 is used as a comparison with the weight table.

BM25 is a function that ranks documents based on the words in the query that appears in each document, regardless of word relationships between documents. BM25 is a function with 2 parameters, namely $k_1$ and $b$. With a query $Q$, which is composed of $q_1, q_2, \ldots, q_n$ words, the BM25 value for $D$ document is

$$score(D, Q) = \sum_{i=1}^{n} IDF(q_i) \cdot \frac{f(q_i, D) \cdot (k_1 + 1)}{f(q_i, D) + k_1 \cdot \left(1 - b + b \cdot \frac{|D|}{avgdl}\right)} \tag{3}$$

where $f(q_i, D)$ is the term frequency $q_i$ in the $D$ document, $|D|$ is the number of words in the $D$ document, and *avgdl* is the average number of words in the corpus. $k_1$ and $b$ are the parameters [23], which are usually $k_1 \in [1.2, 2.0]$ and $b = 0.75$. $IDF(q_i)$ is the IDF value of the word $q_i$.

## 3.3. Genetic operators on the focused web crawler.

The genetic operator used in this focused web crawler has a little difference with the genetic operator in general, especially in the crossover and mutation operations.

In the selection section, the program will try to select individuals with a high level of fitness compared to other individuals [24], in the crossover section, the collected URLs will be selected to be included in the crawl queue, and in the mutation section, the program will develop a query from users to get new URLs that are different but still relevant with the help of search engines.

3.3.1. *Web pages selection.* The selection process used is proportional roulette wheel selection. By using this selection model, each individual will be chosen with probability according to his fitness value. When the wheel is rotated, it will eventually stop with a pointer that points to one of the 6 available segments. The individual probability value $i$ by $P_i$ with $f_1, f_2, \ldots, f_n$ becoming an individual fitness value $1, 2, \ldots, n$ can be formulated as:

$$P_i = \frac{f_i}{\sum_{j=1}^{n} f_j} \tag{4}$$

The advantage of using roulette wheel selection is that all individuals will have the possibility to be selected. This is especially useful for dealing with irrelevant pages. Sometimes, relevant news can be linked to irrelevant web pages, so it is necessary for an irrelevant web page to be selected so that crawler can browse the web page widely.

3.3.2. *Link-based crossover.* In a focused web crawler, as the individual representation of the genetic algorithm is a news page, the crossover operation does not combine 2 individuals into 1, but calculates the value of the outlink URL that was successfully extracted from the selected individual. The crossover operation here will generate a value for each URL like a fitness calculation for an individual.

Each individual who succeeds in surviving the selection process will have their outlink extracted into a collection of URLs. While these URLs are a good extract from the page, they need to be filtered so that they do not fill up the crawling queues of the next generation. Once collected, each URL, $\mu$, will be calculated of its crossover value using the formula [2]:

$$Crossover(\mu) = \sum_{w} fitness(P) \tag{5}$$

where $w$ is all web pages containing URL $\mu$.

3.3.3. *SOS mutation.* The purpose of using the mutation on this focused web crawler is to give crawler the ability to explore a circle of other web pages, which may not be directly linked to individuals on GA.

Since this algorithm is part of a mutation, the program input and output are not explicitly required, but are implicitly created by the program. The main inputs to the SOS algorithm are

**Query:** Queries from users are used as a measure of the fitness value of the organism.

**Documents:** Current generation web pages are required for keyword extraction.

The output of this mutation process is a query sentence after calculating the SOS algorithm into it. The sentence components that are formed will be composed of news documents that are used as input.

Organisms Representation. Before transforming the document into a vector, first every word that appears most frequently in each document will be saved in a table. For example, by querying "debat capres jokowi", it gets 10 pages in the GA population, and it forms

- Doc1={debat}
- Doc2={capres}
- Doc3={jokowi}
- Doc4={prabowo}
- Doc5={jalan}
- Doc6={isu}
- Doc7={prabowosandiaga}
- Doc8={sirat}
- Doc9={jokowi}
- Doc10={prabowo}

All words found in the document are organized into a vector:

$$V = \{\text{debat}, \text{capres}, \text{jokowi}, \text{prabowo}, \text{jalan}, \text{isu}, \text{prabowosandiaga}, \text{sirat}\}$$

Then, each news page is created of an array with the same length as the vector length, but with a value of 1 if the word is found in the relevant news, and $-1$ otherwise. The organism form is

- Doc1=$[1, 1, 1, -1, -1, -1, 1, -1]$
- Doc2=$[-1, 1, -1.1, -1, -1, -1, 1, -1]$
- Doc3=$[1, -1, -1, -1, -1, 1, -1]$
- ...
- Query=$[1, 1, 1, -1, -1, -1, -1, -1]$

The number of elements for each organism depends on the number of documents in the current population and the number of words extracted from those documents.

The Jaccard coefficient formula [25] is used between the vector organism and the query vector. By employing the Jaccard coefficient, the fitness value produced by each organism is in the range of 0 to 1. Genes in organisms must be converted into a binary form before they can be calculated using the Jaccard coefficient. The sigmoid function is used as a function of activating each gene, and after passing the activation function, each gene in the organism will be rounded to a binary value (0 and 1).

In mutualism, $X_i$ is the $i$-th organism in the ecosystem. Other organisms, $X_j$ are randomly selected to interact with organisms $i$. These two organisms interact mutually to increase the survival of the ecosystem.

In commensalism operation, organisms $X_j$ are randomly selected from the ecosystem to interact with organisms $X_i$. In this case, the $i$ organism tries to benefit from the interaction.

In parasitism operation, the organism $X_j$ acts like a malaria mosquito and creates a parasite called "$Parasite\_Vector$". The $Parasite\_Vector$ will be created by duplicating the organism $X_j$. This parasite then attempts to replace organisms $X_i$ in the ecosystem. If $Parasite\_Vector$ created is better, then this parasite replaces the organism $X_i$, but if not, then the organism $X_i$ is considered to have immunity to this parasite and $Parasite\_Vector$ will be removed (considered unable to survive in this ecosystem).

## 4. **Results and Discussion.**

The evaluation used to measure the performance of the focused crawler is the comparison of the relevant pages against all the pages downloaded by the focused crawler, which is usually called the harvest rate. Harvest rate value is the percentage value which shows how well a crawler is to filter irrelevant pages. In general, the formula for calculating the harvest rate is

$$Harvest\ Rate = \frac{\sum_{i \in V} r_i}{|V|} \tag{6}$$

where $|V|$ is the total pages downloaded by the crawler, $r_i$ is the relevance value of a page to the given topic, and the value of $r_i$ consists of only 1 and 0. If the page is relevant, then the value for $r_i = 1$, if not, $r_i = 0$.

There are 2 types of fitness functions used on this focused crawler, namely Okapi BM25 and Cosine Similarity with Weight Table. These two fitness are tested (and compared with the BFS (Breadth First Search) crawler) with the parameters in the previous scenario. The results of the trial comparison between the fitness function and the harvest rate can be seen in Figure 4.

It can be seen from Figure 4 that the use of BM25 as a fitness function for a focused crawler does not provide satisfying results. The average value of focused crawler harvest rate with BM25 after crawling 200 pages is 0.586993819, slightly higher than BFS crawler with a value of 0.55125. This value is far below the use of the cosine weight table with an average value of 0.97495356. This is because the calculation formula for BM25 emphasized on the query, making it difficult to find the relevant page according to BM25.

In the next trial, focused crawler is tested on 5 domains with different topics with a limit of 400 news pages. The results of trials with this topic can be seen in Figure 5.

In Figure 5, the topic and domain have a huge impact on the crawler performance. In antaranews.com, the news pages written sometimes have an outlink to news pages in English. Undeniably, this disturbs the performance of focused crawler, because the words in English cannot be recognized by crawler. Likewise with cnnindonesia.com, this news portal does not have enough outlinks to be crawled simultaneously, so it is necessary to adjust parameters such as increasing population size. The average harvest rate for each topic (Politic, Sport, Technology, Education, Automobile) is: 0.914280754, 0.862709337, 0.861786922, 0.66430155, 0.730313981.

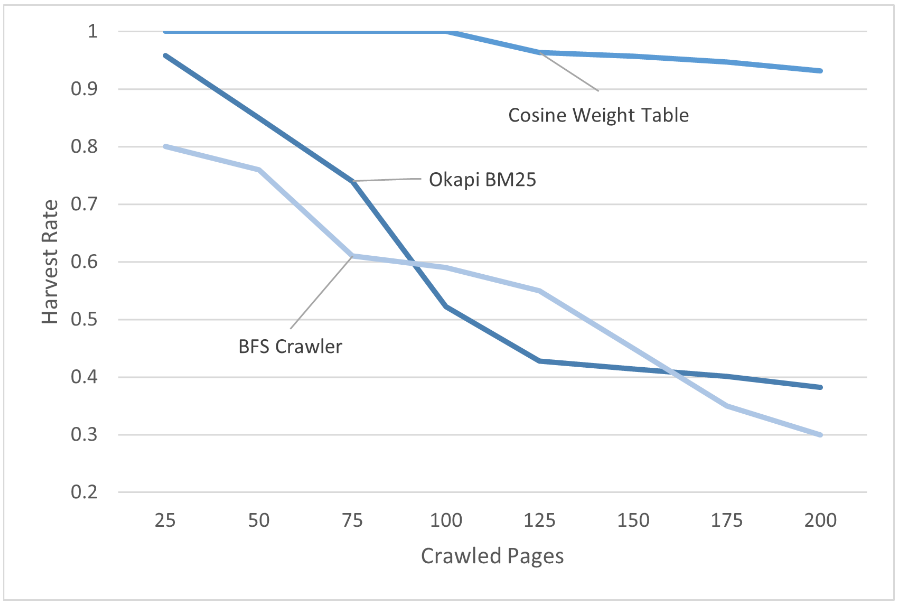


Figure 4. Effect of fitness function on harvest rate

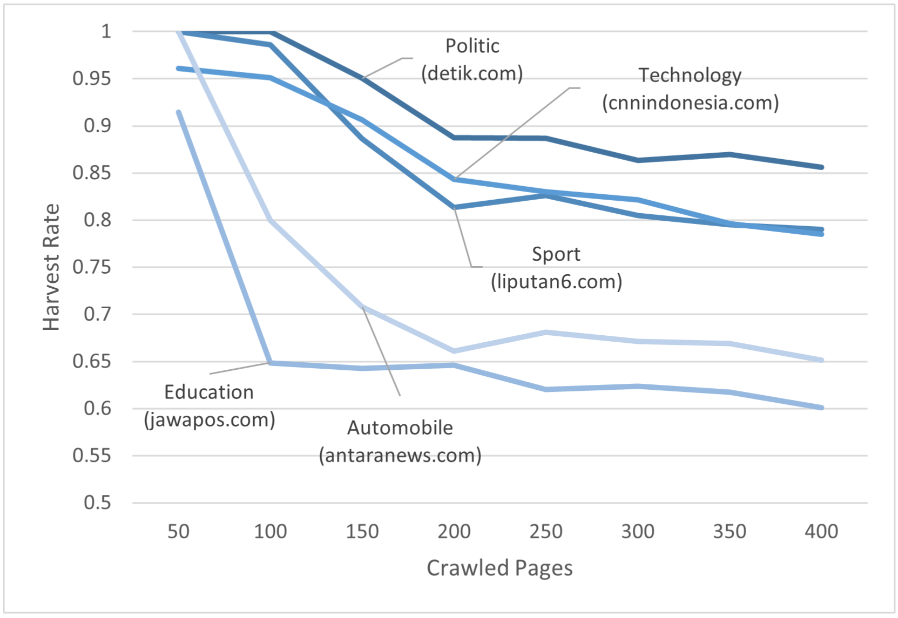


Figure 5. Harvest rate value on certain topics

## 5. Conclusions.

It is vital to build a focused crawler that can adapt to the information on the Internet that continues to grow. This paper proposes a crawling technique by employing GA as a search algorithm for the crawling process. By combining content-based and link-based analysis, the proposed technique has the potential to solve problems that may have occurred with previous focused crawler.

In the test results, it can be seen that the weight table method is effective in browsing web pages than using the Okapi BM25, or the BFS crawler. Larger scale experiments may be needed to study the performance of the GA-crawler against other search algorithms.

## REFERENCES

[1] S. Chakrabarti, M. Van den Berg and B. Dom, Focused crawling: A new approach to topic-specific Web resource discovery, *Computer Networks*, vol.31, nos.11-16, pp.1623-1640, 1999.

[2] B. W. Yohanes, H. Handoko and H. K. Wardana, Focused crawler optimization using genetic algorithm, *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol.9, no.3, pp.403-410, 2011.

[3] F. Menczer, G. Pant, P. Srinivasan and M. E. Ruiz, Evaluating topic-driven Web crawlers, *Proc. of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp.241-249, 2001.

[4] J. Rennie and A. McCallum, Efficient web spidering with reinforcement learning, *Proc. of the International Conference on Machine Learning*, 1999.

[5] J. Cho, H. Garcia-Molina and L. Page, Efficient crawling through URL ordering, *Computer Networks and ISDN Systems*, vol.30, nos.1-7, pp.161-172, 1998.

[6] N. Angkawattanawit and A. Rungsawang, Learnable crawling: An efficient approach to topic-specific web resource discovery, *Proc. of the 2nd International Symposium on Communications and Information Technology (ISCIT)*, 2002.

[7] B. Liu, C. W. Chin and H. T. Ng, Mining topic-specific concepts and definitions on the web, *Proc. of the 12th International Conference on World Wide Web*, pp.251-260, 2003.

[8] L. Page, S. Brin, R. Motwani and T. Winograd, The PageRank citation ranking: Bringing order to the web, *Stanford InfoLab*, 1999.

[9] P. Srinivasan, G. Pant and F. Menczer, Target seeking crawlers and their topical performance, *Proc. of Int. Conf. Research and Development in Information Retrieval*, 2002.

[10] M. Chau and H. Chen, Comparison of three vertical search spiders, *Computer*, vol.36, no.5, pp.56-62, 2003.

[11] Y. Sun, P. Jin and L. Yue, A framework of a hybrid focused web crawler, *2008 2nd International Conference on Future Generation Communication and Networking Symposia*, vol.2, pp.50-53, 2008.

[12] M. Shokouhi, P. Chubak and Z. Raeesy, Enhancing focused crawling with genetic algorithms, *International Conference on Information Technology: Coding and Computing (ITCC'05)*, vol.2, pp.503-508, 2005.

[13] J. Hosseinkhani, H. Taherdoost and S. Keikhaee, ANTON framework based on semantic focused crawler to support web crime mining using SVM, *Annals of Data Science*, pp.1-14, 2019.

[14] J. Hernandez, H. M. Marin-Castro and M. Morales-Sandoval, A semantic focused web crawler based on a knowledge representation schema, *Applied Sciences*, vol.10, no.11, p.3837, 2020.

[15] A. Pal, D. S. Tomar and S. C. Shrivastava, Effective focused crawling based on content and link structure analysis, *International Journal of Computer Science and Information Security*, vol.2, no.1, 2009.

[16] Q. Zhu, An algorithm OFC for the focused web crawler, *2007 International Conference on Machine Learning and Cybernetics*, vol.7, pp.4059-4063, 2007.

[17] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, Springer Science and Business Media, 2013.

[18] L. Davis, *Handbook of Genetic Algorithms*, 1991.

[19] D. E. Goldberg and J. H. Holland, Genetic algorithms and machine learning, *Machine Learning*, vol.3, pp.95-99, 1988.

[20] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*, MIT Press, 1992.

[21] C. Qu, B. Wang and P. Wei, Efficient focused crawling strategy using combination of link structure and content similarity, *2008 IEEE International Symposium on IT in Medicine and Education*, pp.1045-1048, 2008.

[22] S. Robertson and H. Zaragoza, The probabilistic relevance framework: BM25 and beyond, *Foundations and Trends in Information Retrieval*, vol.3, no.4, pp.333-389, 2009.

[23] H. Schütze, C. D. Manning and P. Raghavan, *Introduction to Information Retrieval*, Cambridge University Press, Cambridge, 2008.

[24] N. M. Razali and J. Geraghty, Genetic algorithm performance with different selection strategies in solving TSP, *Proceedings of the World Congress on Engineering*, vol.2, no.1, pp.1-6, 2011.

[25] V. Thada and V. Jaglan, Comparison of Jaccard, dice, cosine similarity coefficient to find best fitness value for web retrieved documents using genetic algorithm, *International Journal of Innovations in Engineering and Technology*, vol.2, no.4, pp.202-205, 2013.