Journal of Innovative Technology Convergence
Vol. 5, No. 2 December 2023, pp. 45-56

Analysis of the Enhanced Boyer-Moore Search Algorithm for a Desktop PC Search Engine

Lorlyn S. Sernicula [1], Melinda L. Manalo [2], Gryam Abner M. Niebre [3], Rajean C. Anastacio [4], Jhona P. Alagos [5], Jason P. Sermeno [6]*

Abstract: In the digital era, the volume of digital contents stored on personal computers has been rapidly and continuously increasing. Finding documents among this bulk of files has become tedious and stressful. In this regard, string searching and pattern matching algorithms must be employed to alleviate the performance of search engines, making it easier to find the desired documents. This paper aims to analyze the performance of an enhanced Boyer-Moore (BM) search algorithm to assist in finding the intended information or files. A series of tests have been conducted in order to compare the enhanced BM algorithm with the traditional BM algorithm. The results have shown that the Enhanced Boyer-Moore Algorithm performed the search and managed to speed up the searching process by reducing the time duration and the accuracy of the overall performance as the input text increased.

Keywords: Enhanced Boyer-Moore search algorithm, Local search engine, String searching, String matching

1. Introduction

String searching and pattern matching algorithms are integrated into search engines to find information or files, and their performance must be efficient as the volume of digital information over the Internet and the number of files stored on desktop personal computers (PCs) rapidly increase over the past decades. String matching algorithms play a crucial role in various real-world problems, including database schemas and network systems. These algorithms are vital in processing and analyzing

[1] College of Computer Studies, University of Antique, Sibalom, Antique, Philippines
Email: lorlyn.sernicula@gmail.com
[2] College of Computer Studies, University of Antique, Sibalom, Antique, Philippines
Email: melinda.manalo@antiquespride.edu.ph
[3] College of Computer Studies, University of Antique, Sibalom, Antique, Philippines
Email: gryamniebre@gmail.com
[4] College of Computer Studies, University of Antique, Sibalom, Antique, Philippines
Email: rajean.anastacio@antiquespride.edu.ph
[5] College of Computer Studies, University of Antique, Sibalom, Antique, Philippines
Email: jhona.alagos@antiquespride.edu.ph
[6]* College of Computer Studies, University of Antique, Sibalom, Antique, Philippines
Email: jasonpsermeno@gmail.com (Corresponding Author)

Received [February 15, 2023]; Revised [April 5, 2023]; Accepted [June 20, 2023]

Published by InnoCon Publishing.
ISSN 2704-4440

textual data and in performing time-efficient tasks in multiple domains [1]. String searching and pattern matching algorithms are used to search a string within another string. They are also used in a variety of operations on strings that include simple concatenation, pattern matching, transforming strings, identifying specific patterns or substrings within strings, *etc*.

According to Simoes [2], desktop PC search engines are simply information retrieval programs on your computer that are designed to help find information. The program searches through the folders and files using specific keywords that are typed into the search bar and then returns the list of results that were found to contain the keywords. A PC search engine differs from a web search engine since the searches are performed only on local files. It also refers to an "offline search engine". According to Cole [3], desktop PC search engines that are built into current operating systems (OSs), e-mail programs, and other applications have far fewer capabilities as compared to popular Web search engines [4][5].

According to Gou [6], the most common algorithms that are used for searching include the naïve algorithm [7], Knuth-Morris-Pratt [8], Boyer-Moore [9][10], and the Rabin-Karp algorithms [9]. The idea of the naive solution is just to make a comparison character by character of the text and return all the valid shifts found [7]. On the other hand, the main characteristic of the KMP algorithm is that, each time a match between the pattern and a shift in the text fails, the algorithm will use the information given by a specific table, obtained by a preprocessing of the pattern, to avoid re-examination of the characters that have been previously checked, thus limiting the number of comparisons required [8]. The Rabin-Karp algorithm uses a totally different approach to solve the string matching problem as it is based on hashing techniques. This algorithm filters the characters that do not match and then performs the comparison by using the same hash function for each substring [11].

This paper deals with the analysis of the enhancement of the Boyer-Moore string search algorithm, developed by Robert S. Boyer and J Strother Moore in 1977 [9]. In this algorithm, the match is performed from right to left, which allows the algorithm to skip more characters than the other algorithms. The main purpose of the study is to enhance the Boyer Moore Search Algorithm as a basis for a local search engine for desktop PCs. And specifically, it attempts to achieve the following objectives:

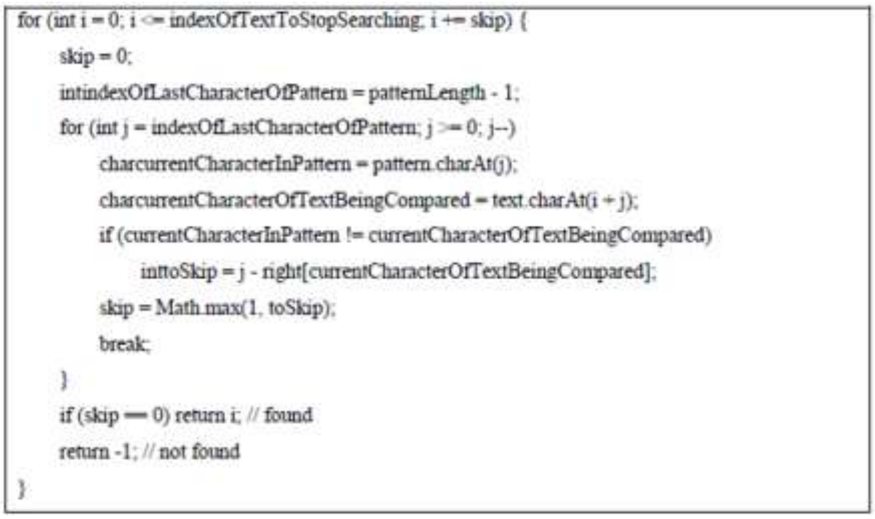
- To determine the average duration of the traditional and enhanced BM search algorithms.
- To determine the average accuracy rate of the traditional and enhanced BM search algorithms.
- To determine the average performance of the traditional and enhanced BM search algorithms.

The remainder of this paper is organized as follows: Section 2 outlines the overview of both the traditional and enhanced BM algorithms; Section 3 details the methodology for the analysis of the algorithms; the results and discussion are detailed in Section 4; and Section 5 concludes the study.

2. Overview of the Enhanced Boyer-Moore Search Algorithm

The BM search algorithm is one of the efficient algorithms used to search a string for a local search engine on desktop PCs. The BM algorithm compares the pattern with the text from right to left. If the text symbol that is compared with the rightmost pattern symbol does not occur in the pattern at all, then the pattern can be shifted.

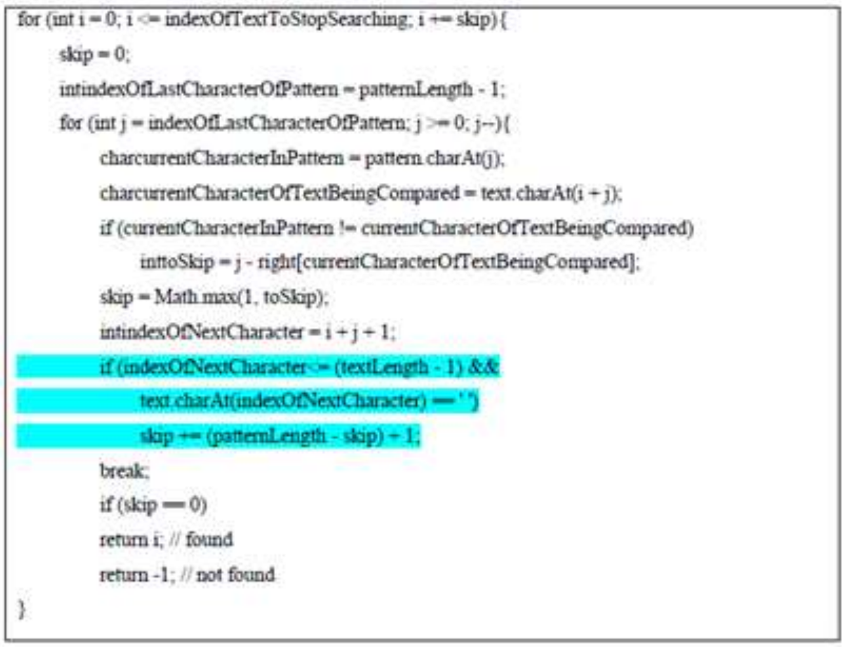
The traditional BM search algorithm, as depicted in Figure 1, is considered the most popular and efficient string-matching algorithm in common applications. It is often implemented in text editors using the "*search*" and "*substitute*" commands [9]. It works by scanning the characters of the pattern (*i.e.*, a string to be searched), beginning with the rightmost one, and performing the comparisons with the text (*i.e.*, a string being searched) from right to left. In case of a mismatch or if there is a complete match of the whole pattern, two recomputed functions (*i.e.*, bad character and good suffix functions) will be used by the algorithm to shift the window to the right [12][13].



```
for (int i = 0; i <= indexOfTextToStopSearching; i += skip) {
    skip = 0;
    intindexOfLastCharacterOfPattern = patternLength - 1;
    for (int j = indexOfLastCharacterOfPattern; j >= 0; j--)
        charcurrentCharacterInPattern = pattern.charAt(j);
        charcurrentCharacterOfTextBeingCompared = text.charAt(i + j);
        if (currentCharacterInPattern != currentCharacterOfTextBeingCompared)
            inttoSkip = j - right[currentCharacterOfTextBeingCompared];
        skip = Math.max(1, toSkip);
        break;
    }
    if (skip == 0) return i; // found
    return -1; // not found
}
```

Figure 1. The Traditional Boyer-Moore Search Algorithm

The enhanced BM search algorithm works in linear time, that is, there are a lower number of occurrences of the pattern in a string. It is observed that the performance of string searching algorithms is based on the selection of algorithms used for matching. The key search string-matching words were determined using the formula tool by comparing the length of the keywords to the text string words. The algorithm depicted in Figure 2 shows the implementation of the enhanced BM search algorithm, which can perform better than the traditional BM search algorithm.



```
for (int i = 0; i <= indexOfTextToStopSearching; i += skip){
    skip = 0;
    intindexOfLastCharacterOfPattern = patternLength - 1;
    for (int j = indexOfLastCharacterOfPattern; j >= 0; j--){
        charcurrentCharacterInPattern = pattern.charAt(j);
        charcurrentCharacterOfTextBeingCompared = text.charAt(i + j);
        if (currentCharacterInPattern != currentCharacterOfTextBeingCompared)
            inttoSkip = j - right[currentCharacterOfTextBeingCompared];
        skip = Math.max(1, toSkip);
        intindexOfNextCharacter = i + j + 1;
        if (indexOfNextCharacter <= (textLength - 1) &&
            text.charAt(indexOfNextCharacter) == ' ')
            skip += (patternLength - skip) + 1;
        break;
        if (skip == 0)
        return i; // found
        return -1; // not found
}
```

Figure 2. The Implementation of Enhanced BM Search Algorithm

3. Methodology

3.1 Statistical Tool to Measure the Search Performance

In order to determine the average performance of both the traditional and enhanced BM search algorithms, a mathematical formula, as shown in Equation 1, was used to measure the percentage of the performance of every string that was entered.

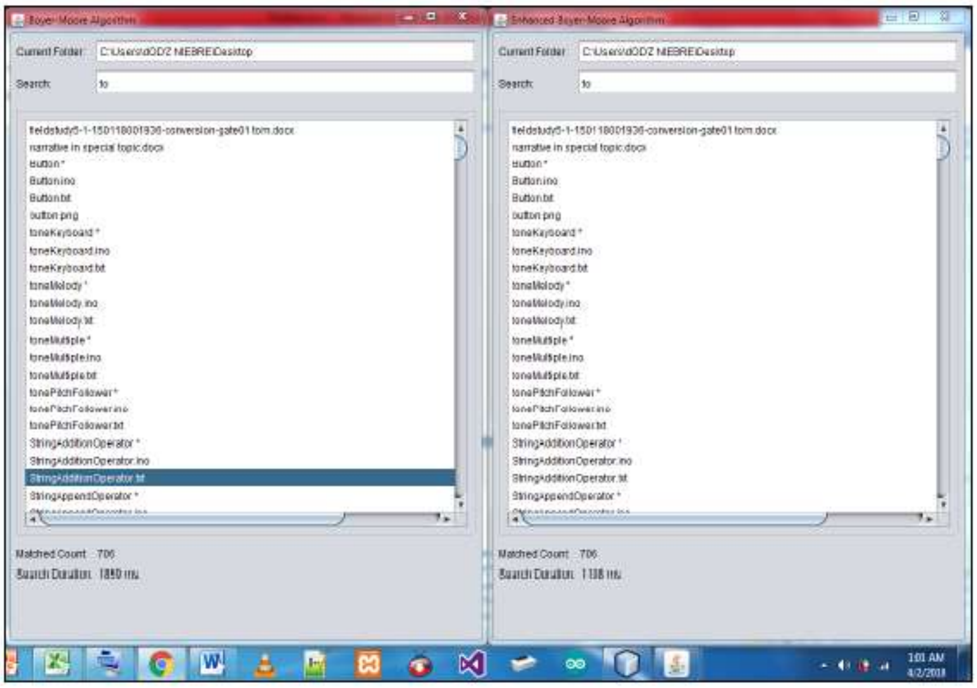
$$m = \frac{n+n^2}{2} * p \tag{1}$$

Where:

- m is the average performance;
- n is equal to the average duration of the enhanced BM search algorithm;
- n^2 is equal to the average duration of the traditional BM search algorithm;
- p which is equal to 0.100, is the accuracy coefficient of each string.

3.2 Simulation Environment to Measure the Search Performance

A desktop PC search engine prototype is developed, as shown in Figure 3, that integrates both the traditional and enhanced BM algorithms. The prototypes for both the traditional BM search algorithm and the enhanced BM search algorithm have been designed in order to test their performance in terms of their duration time in searching for strings on a desktop PC.



```
Current Folder:  C:\Users\dODZ MEBRE\Desktop
Search:          to

fieldstudy5-1-150118001936-conversion-gate01 tom.docx
narrative in special topic.docx
Button *
Button.ino
Button.txt
button.png
toneKeyboard *
toneKeyboard.ino
toneKeyboard.txt
toneMelody *
toneMelody.ino
toneMelody.txt
toneMultiple *
toneMultiple.ino
toneMultiple.txt
tonePitchFollower *
tonePitchFollower.ino
tonePitchFollower.txt
StringAdditionOperator *
StringAdditionOperator.ino
StringAdditionOperator.txt
StringAppendOperator *

Matched Count : 706
Search Duration : 1880 ms
```

```
Current Folder:  C:\Users\dODZ MEBRE\Desktop
Search:          to

fieldstudy5-1-150118001936-conversion-gate01 tom.docx
narrative in special topic.docx
Button *
Button.ino
Button.txt
button.png
toneKeyboard *
toneKeyboard.ino
toneKeyboard.txt
toneMelody *
toneMelody.ino
toneMelody.txt
toneMultiple *
toneMultiple.ino
toneMultiple.txt
tonePitchFollower *
tonePitchFollower.ino
tonePitchFollower.txt
StringAdditionOperator *
StringAdditionOperator.ino
StringAdditionOperator.txt
StringAppendOperator *

Matched Count : 706
Search Duration : 1138 ms
```

Figure 3. GUI Design of the Prototype for both Traditional and Enhanced BM Search Algorithms

For the simulation, a laptop with an Intel Core™ i5-4200 CPU, 64-bit Windows 7 Ultimate Operating System (OS), 12GB of RAM (11.9 usable), and x64-based HD graphics (1.60GHz to 2.30 GHz). In addition, NetBeans IDE 8.2 was used for coding and debugging the codes of the algorithm written in Java programming language.

The difference in speed between the traditional and enhanced BM search algorithms is determined by subtracting the start time from the end time of searching the text.

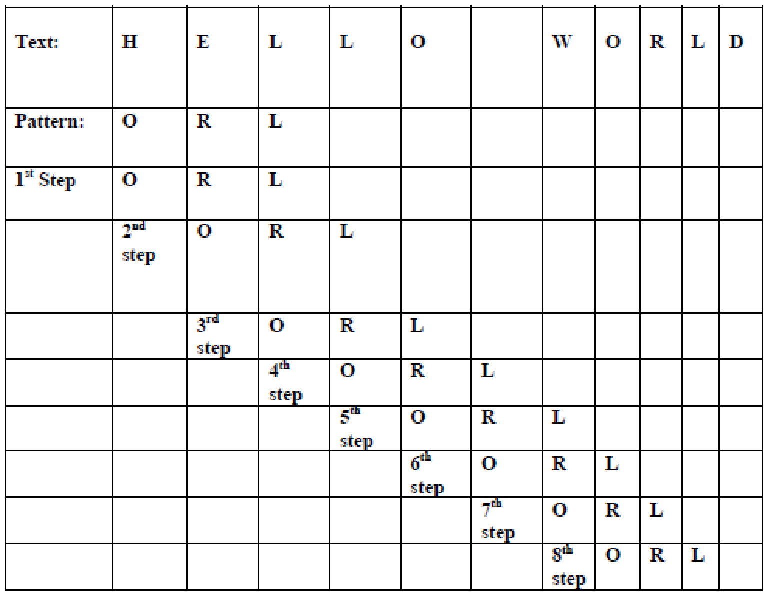
4. Results and Discussion

4.1 Presentation and Interpretation of Data

4.1.1 Traditional Boyer-Moore Search Algorithm

The step-by-step execution of the search process of the traditional BM search algorithm is shown in Table 1. The search skips from left to right but compares and checks the character from right to left.

Table 1. The Execution of the Traditional BM Search Algorithm



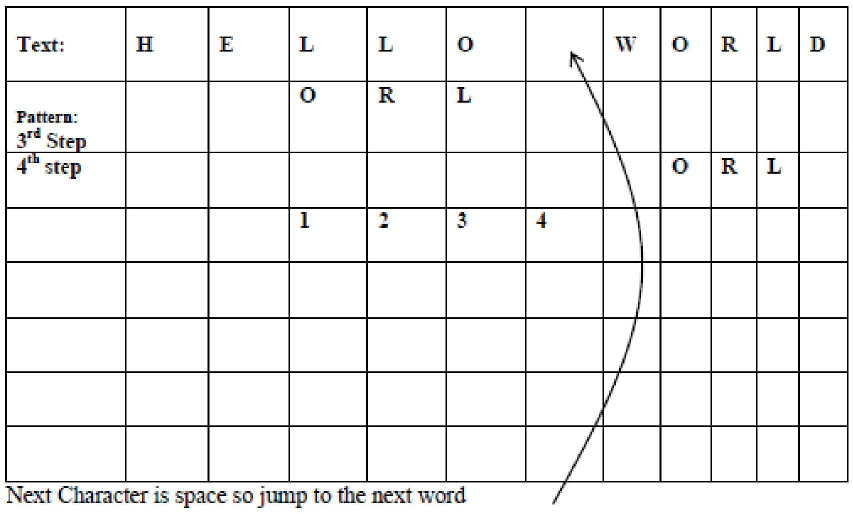
Text:	H	E	L	L	O		W	O	R	L	D
Pattern:	O	R	L								
1st Step	O	R	L								
	2nd step	O	R	L							
		3rd step	O	R	L						
			4th step	O	R	L					
				5th step	O	R	L				
					6th step	O	R	L			
						7th step	O	R	L		
							8th step	O	R	L	

In Step 1, it compares the character "L" which is found in the "ORL" pattern, with the first character "L" in the text "HELLO". The character "L" matched, then continued to compare the second pattern, which is character "R" that is found in the "ORL" pattern, then character "E" in the text "HELLO", but character "R" is not matched to character "E", so the process will proceed to Step 2, and so on, until the character being searched matches the complete text "ORL". The pattern "ORL" was found at Step 8.

4.1.2 Enhanced Boyer-Moore Search Algorithm

The search execution of the proposed enhanced BM search algorithm is shown in Table 2. It is a parallel execution and has two partitions that run at the same time. Once the next character is a "space" character, it skips the space and then moves directly to the pattern that has a string. Steps 1, 2, and 3 were skipped because of the space, and it directly moves to Step 4, then the pattern is found in Step 4.

Table 2. The Execution of the Enhanced Boyer-Moore Search Algorithm



Text:	H	E	L	L	O		W	O	R	L	D
Pattern: 3rd Step			O	R	L						
4th step								O	R	L	
			1	2	3	4					

Next Character is space so jump to the next word

4.2 Data Analysis

The results of the actual experiment are shown in this section. Figure 4 shows the difference in duration time between the traditional and the enhanced BM search algorithms in every test for String A “chapter”.

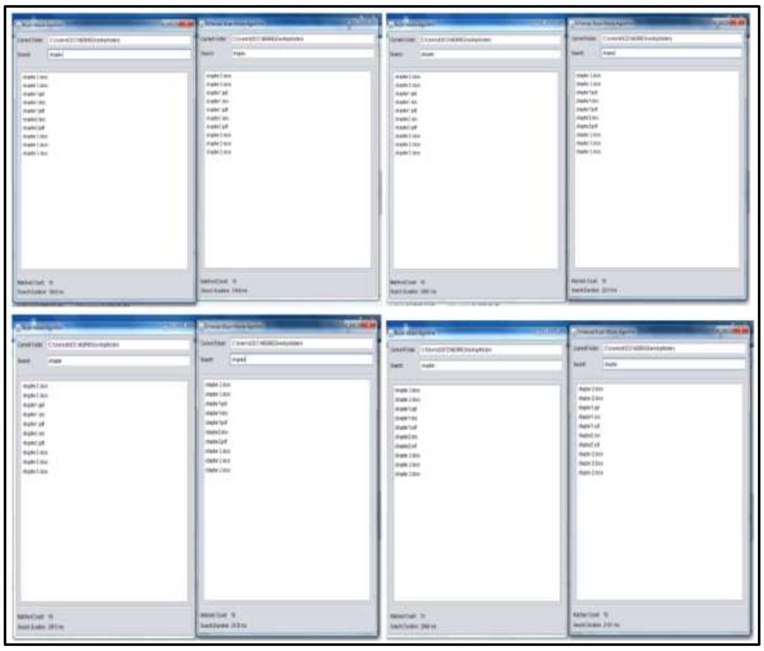


Figure 4. String A Results (chapter)

For the first search, the string “chapter” was entered. The duration time in the first test of the traditional BM search algorithm took 3.022 seconds to find the string, while the enhanced BM search algorithm took 2.140 seconds. A second test was conducted with the same string “chapter” and the traditional BM search algorithm takes 2.905 seconds, while the enhanced BM search algorithm takes 2.201 seconds to find the string. In the third test, the traditional BM search algorithm takes 2.970 seconds, while the enhanced BM search algorithm takes 2.135 seconds to find the string. Finally, in the fourth test, the traditional BM search algorithm takes 2.988 seconds, while the enhanced BM search algorithm takes 2.101 seconds. The average duration time of the traditional BM search algorithm is 2.97

seconds, while the average duration time of the enhanced BM search algorithm to find the string is 2.14 seconds. This simply shows that the enhanced BM search algorithm performs the search faster than the traditional BM search algorithm. Figure 5 shows the time durations of the four tests conducted on the string “chapter” for both the traditional and enhanced BM search algorithms.

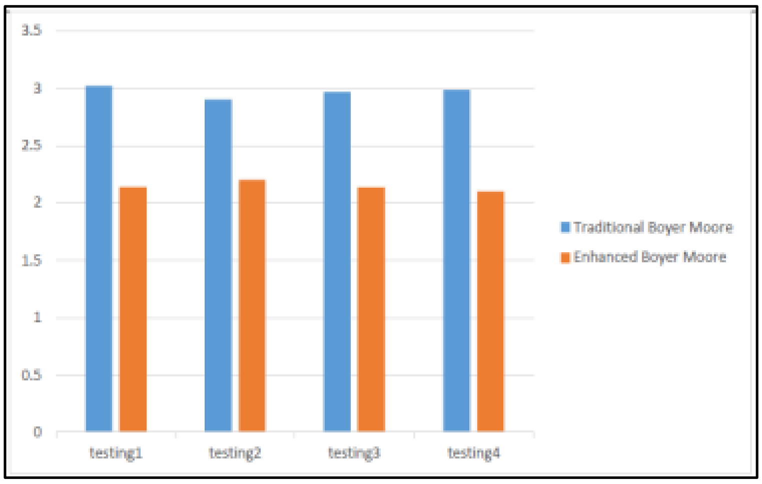


Figure 5. The Time Duration for Searching String A “chapter”

Figure 6 shows the simulation for both the traditional and enhanced BM search algorithms in every test for String B “thesis”.

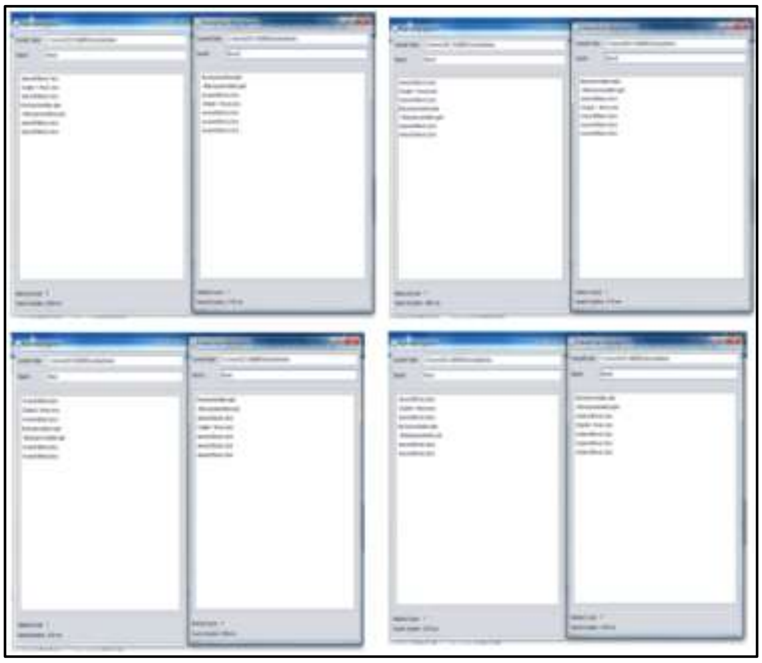


Figure 6. String B Results (thesis)

For the second run of the simulation, String B “thesis” was entered. The duration time in the first test for the traditional BM search algorithm takes 2.895 seconds, while the enhanced BM search algorithm takes 2.110 seconds to find string B. A second test was conducted with the same string “thesis”, and the traditional BM search algorithm takes 2.945 seconds, while the enhanced BM search algorithm takes 2.115 seconds. In the third test, the traditional BM search algorithm takes 2.916 seconds, while the enhanced BM search algorithm takes 2.106 seconds to perform the search. In the fourth test, the traditional BM search algorithm takes 2.926 seconds, while the enhanced BM search algorithm takes 2.166 seconds. The average duration time of the traditional BM search algorithm to find the string is 2.92 seconds, while the average duration time of the enhanced BM search algorithm is 2.12 seconds. This simply shows that the enhanced BM search algorithm performs the search faster than the traditional BM search algorithm. Figure 7 shows the time durations of the four tests conducted on the string "thesis" for both the traditional and enhanced BM search algorithms.

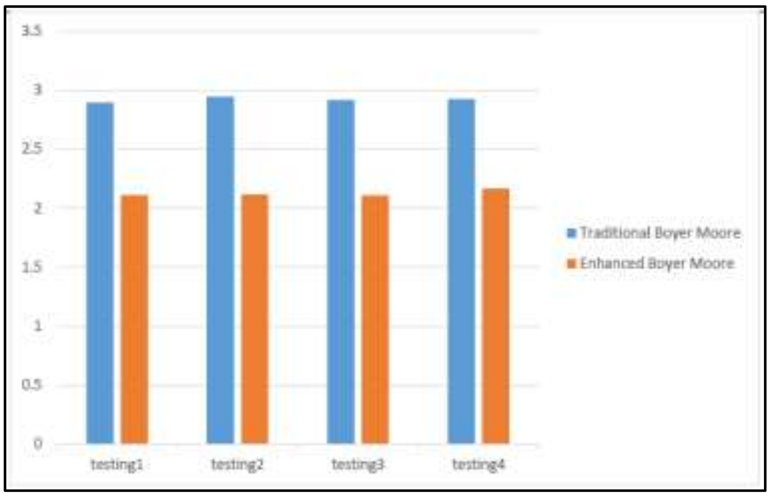


Figure 7. The Time Duration for Searching String B "thesis"

Figure 8 shows the difference duration time between the traditional and the enhanced BM search algorithms in every test for String C "doc".

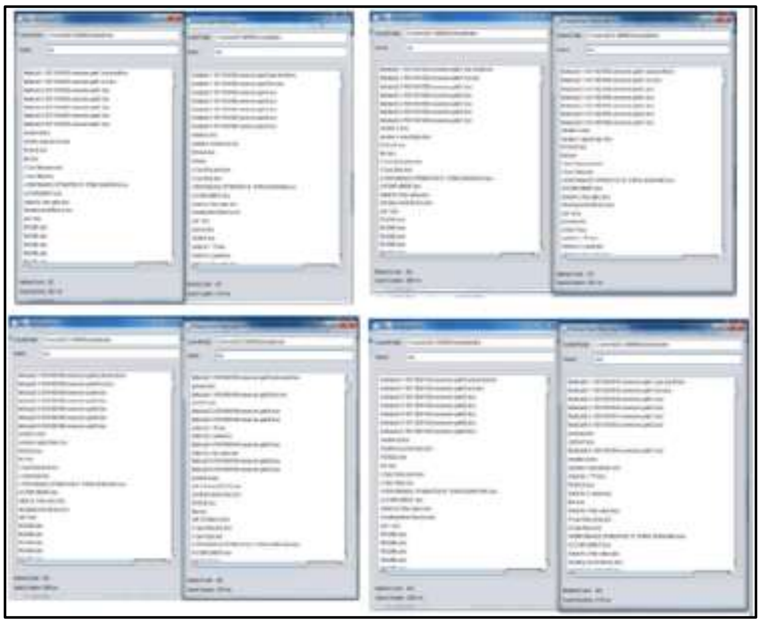


Figure 8. String C Results (doc)

In the third rum of the simulation, String C, which is "doc", was entered. In the first test, the traditional BM search algorithm takes 2.891 seconds to find the string, while the enhanced BM search algorithm takes 2.144 seconds. In the second test for the same string "doc", the traditional BM search algorithm takes 2.928 seconds, while the enhanced BM search algorithm takes 2.115 seconds. In the third test, the traditional BM search algorithm takes 2.981 seconds, while the enhanced BM search algorithm takes 2.097 seconds. During the fourth test, the traditional BM search algorithm takes 2.965 seconds, while the enhanced BM search algorithm takes 2.115 seconds. The average duration time of the traditional BM search algorithm is 2.94 seconds to perform the search, while the average duration time for the enhanced BM search algorithm is 2.12 seconds. This simply shows that the enhanced BM search algorithm performs the search faster than the traditional BM search algorithm. Figure 9 shows the time durations of the four tests conducted on the string "doc" for both the traditional and enhanced BM search algorithms.

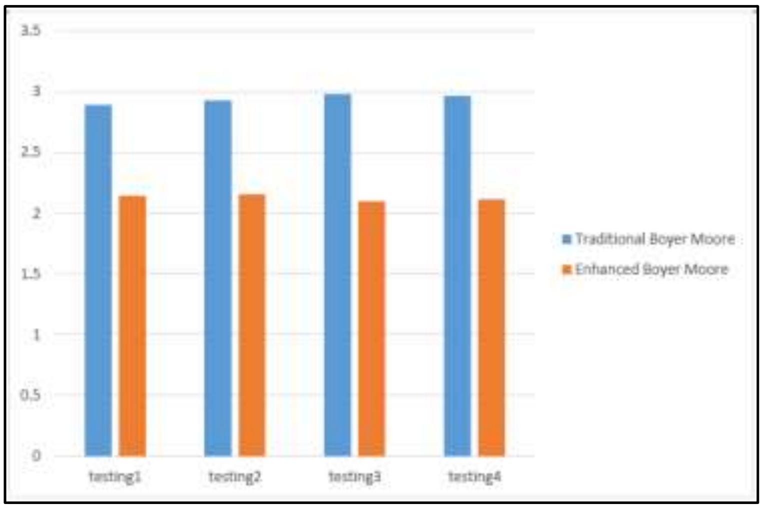


Figure 9. The Time Duration for Searching String C "doc"

Figure 10 shows the difference in duration time between the traditional and enhanced BM search algorithms in every test for String D "Performance".

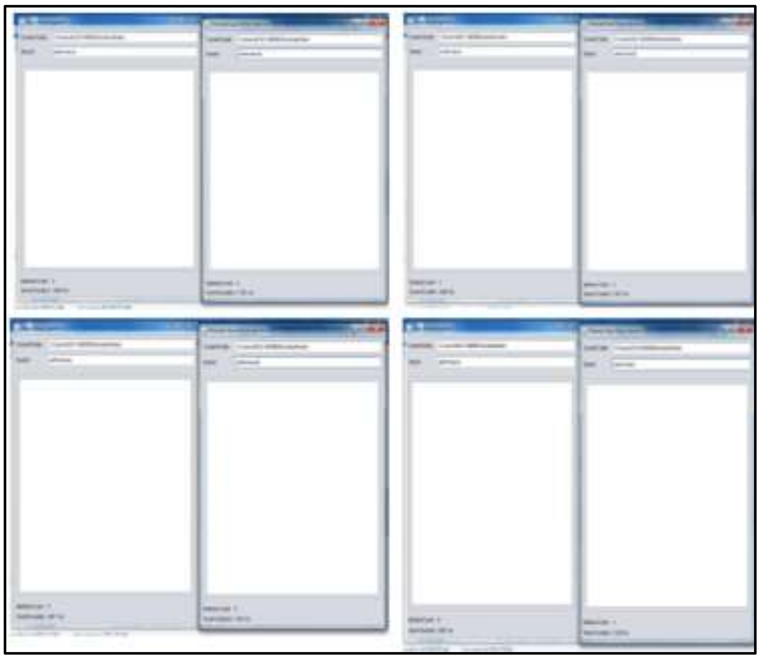


Figure 10. String D Results (Performance)

In the fourth run of the simulation, string D "Performance" was entered. In the first test, the traditional BM search algorithm takes 2.906 seconds to perform the search, while the enhanced BM search algorithm takes 2.101 seconds. In the second test with the same string "Performance", the traditional BM search algorithm takes 2.926 seconds, while the enhanced BM search algorithm takes 2.151 seconds. In the third test, the traditional BM search algorithm takes 2.917 seconds, while the enhanced BM search algorithm takes 2.130 seconds. In the fourth test, the traditional BM search algorithm takes 2.926 seconds, while the enhanced BM search algorithm takes 2.120 seconds. The average duration time of the traditional BM search algorithm to perform the search is 2.92 seconds, while the average duration time of the enhanced BM search algorithm is 2.13 seconds. This simply shows that the enhanced BM search algorithm performs the search faster than the traditional BM search algorithm. Figure 11 shows the time durations of the four tests conducted on the string "Performance" for both the traditional and enhanced BM search algorithms.

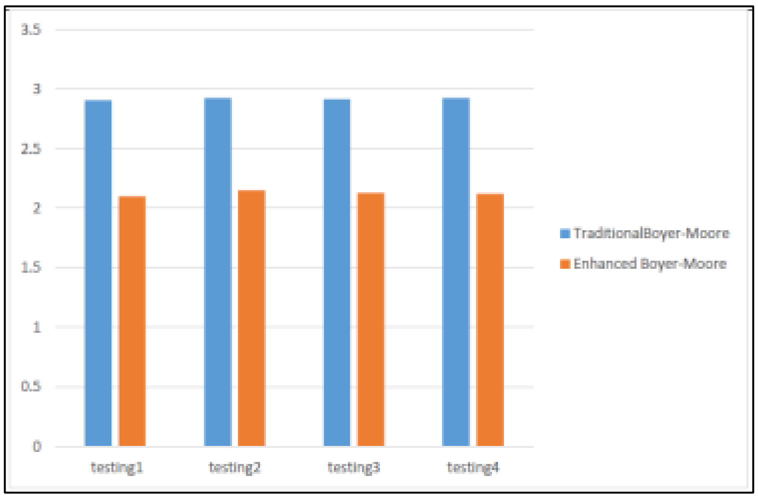


Figure 11. The Time Duration for Searching String D "Performance"

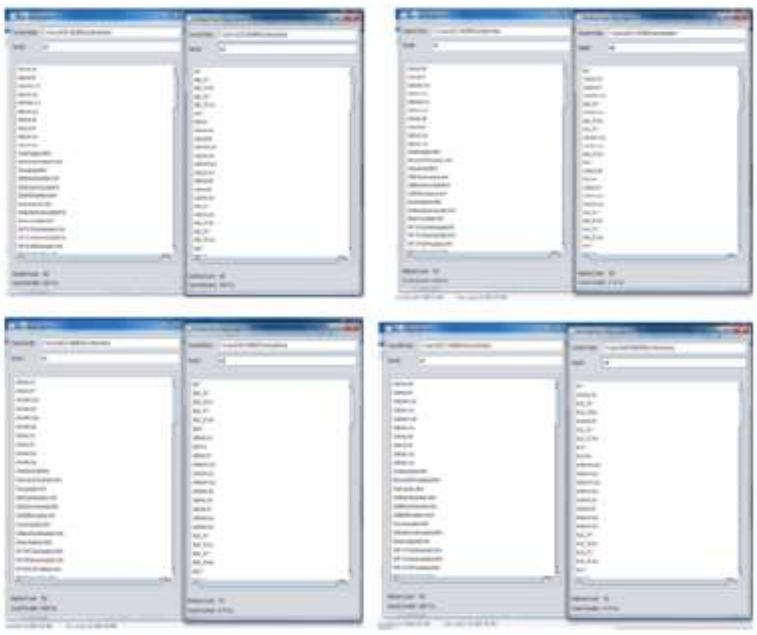


Figure 12. String E Results (Lab)

Figure 12 shows the difference in duration time between the traditional and the enhanced BM search algorithm in every test for String E "Lab".

In the fifth run of the simulation, the string "Lab" was entered as String E. In the first test, the traditional BM search algorithm takes 2.927 seconds to perform the search, while the enhanced BM search algorithm takes 2.115 seconds. A second test for the same string was conducted, and the traditional BM search algorithm takes 2.947 seconds, while the enhanced BM search algorithm takes 2.115 seconds. In the third test, the traditional BM search algorithm takes 2.958 seconds, while the enhanced BM search algorithm takes 2.175 seconds. In the fourth test, the traditional BM search algorithm takes 2.912 seconds, while the enhanced BM search algorithm takes 2.188 seconds. The average duration time for performing searches with the traditional BM search algorithm is 2.44 seconds, while the average duration time for the enhanced BM search algorithm is 2.13 seconds. This simply shows that the enhanced BM search algorithm performs the search faster than the traditional BM search algorithm. Figure 13 shows the time durations of the four tests conducted on the string "Lab" for both the traditional and enhanced BM search algorithms.

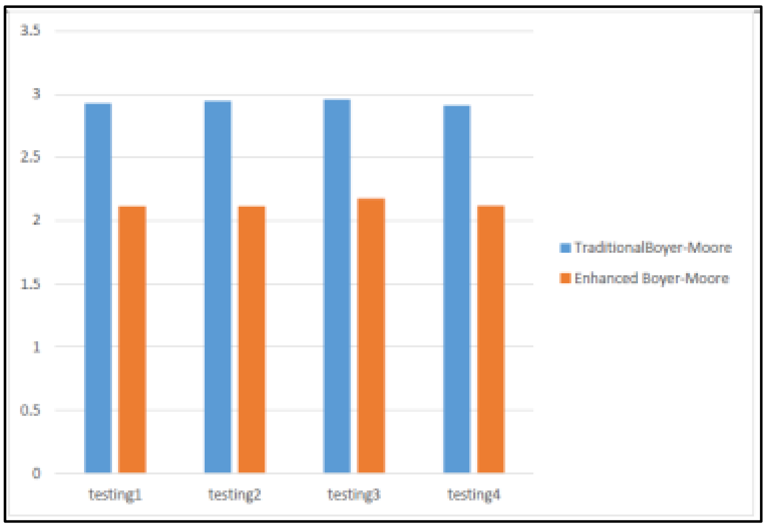


Figure 13. The Time Duration for Searching String E "Lab"

5. Conclusion

This paper has analyzed the performance of the enhanced BM string searching algorithm to form the basis for a local search engine for desktop PCs. The enhanced algorithm runtime depends on the size of the queue, that is, the number of index values in the queue, which shows the number of expected patterns in text *T*. As compared with the traditional BM algorithm, the enhanced BM search algorithm for local desktop PCs is capable of reducing the runtime length as well as the time duration while performing a search on the contents of a file.

In the future, a prototype implementation of the enhanced BM search algorithm will be designed to further investigate its robustness and efficiency as compared with other string searching algorithms. The algorithm will be implemented using a Java program.

References

[1] BeyondVerse, "*Algorithms for String Manipulation and Matching*", Medium, www.medium.com/@beyond_verse/algorithms-for-string-manipulation-and-matching-2f9a450ffd7b (Accessed November 2, 2020)

[2] J. Simoes, "*What is a Computer Search Engine*", lookeen.com, www.lookeen.com/blog/what-is-a-computer-search-engine (Accessed November 2, 2020).

[3] B. Cole, "*Search Engines Tackle the Desktop*", Computer, vol. 38, no. 3, March 2005, pp.14-17, doi: 10.1109/MC.2005.103.

[4] J. Y. Lemos, A. R. Joshi, "*Search engine optimization to enhance user interaction*", in Proc. of 2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Palladam, India, February 10-11, 2017, doi: 10.1109/I-SMAC.2017.8058379.

[5] Google Search, "*Explore the world of Google Search*", www.google.com/competition/howgooglesearchworks.html (Accessed November 2, 2020).

[6] M. Gou, "*Algorithms for String matching*", student.montefiore.ulg.ac.be, www.student.montefiore.ulg.ac.be/~s091678/files/OHJ2906_Project.pdf (Accessed November 2, 2020).

[7] H. F. Mahmood, "The naive algorithm for pattern searching", Educative, www.educative.io/answers/the-naive-algorithm-for-pattern-searching (Accessed November 2, 2020).

[8] D. E. Knuth, J. H. Morris Jr., V. R. Pratt, "*Fast Pattern Matching in Strings*", SIAM Journal on Computing, vol. 6, no. 2, 1977, pp.323–350, doi: 10.1137/0206024.

[9] R. S. Boyer, J. S. Moore, "*A Fast String Searching Algorithm*", Communications of the ACM, vol. 20, no. 10, October 1977, pp.762-772, doi: 10.1145/359842.359859.

[10] A. Hume, D. Sunday, "*Fast string searching*", Software: Practice and Experience, vol. 21.no. 11, November 1991, pp.1221-1248, doi: 10.1002/spe.4380211105.

[11] Programiz, "*Rabin-Karp Algorithm*", www.programiz.com/dsa/rabin-karp-algorithm (Accessed November 2, 2020).

[12] N. Horspool, "*Practical fast searching in strings*", Software: Practice and Experience, vol. 10, no. 6, June 1980, pp. 501-506, doi: 10.1002/spe.4380100608.

[13] B. Langmead, "*Boyer-Moore*", www.cs.jhu.edu/~langmea/resources/lecture_notes/boyer_moore.pdf (Accessed November 2, 2020).