**Analysis of Round Robin Algorithm with the Implementation of Alternative Sorting Algorithm for Time Efficiency**

As-il, Jhaeus Herzian T.

Governor Pack Road, Baguio City

Philippines 2600

jhaeus8@gmail.com

Baleña, Julius Bernadin S.

Governor Pack Road, Baguio City

Philippines 2600

juliusbalena9@gmail

Castillo, John Ariel Dominic J.

Governor Pack Road, Baguio City

Philippines 2600

nikcastillo11@gmail.com

**ABSTRACT**

The Round Robin Algorithm is a preemptive process Scheduling Algorithm used in CPUs. This study was conducted to analyze whether or not implementing a sorting algorithm in the Round Robin Scheduling Algorithm could enhance its Time Efficiency. By implementing an Alternative Sorting Algorithm in the Round Robin Algorithm, it could theoretically improve the algorithm. A Dynamic Time Quantum-based Burst Time is also used to help with the time splicing for the processes. The algorithm had gone through three tests where it processes Burst Time in ascending and descending order, and high values. The results show that implementing the Alternative Sorting Algorithm does not enhance the Time Efficiency of the Round Robin Algorithm. The results show, however, that the Sorting Algorithm had an effect on the Round Robin Algorithm, be it negative.

**Keywords**

Round Robin Scheduling Algorithm; Alternating Sorting Algorithm; Time Efficiency

1. **INTRODUCTION**

The researchers will make use of past studies on Round Robin Algorithms where other Algorithms were implemented within it to make the algorithm more efficient. (Manuel, Baquirin, Guevara, & Tandingan Jr., 2019) (Shyam & Nandal, 2014)

The researchers will make use of similar elements and concepts for their test cases, notably the number of processes, the arrangement of variables and the values of the Burst Times. (Mishra & Rashid, 2014) (Pati, Pranav, & Pallav, 2017) (Dash, Sahu, & Samantra, 2015)

In order to make the Round Robin Algorithm time efficient, the researchers will add the Alternative Sorting Algorithm and manipulate the Time Quantum of the algorithm.

By basing the Time Quantum on the process’ minimum and maximum Burst Time, the context switching could be reduced. (H.S.Behera, 2010)

If done correctly the Round Robin Algorithm’s criteria for scheduling will all be improved. A static Time Quantum will affect the time efficiency of the algorithm since it will affect the criteria of the algorithm. (J R Indusree, 2017)

The researchers plan to make the Round Robin Algorithm time efficient by adding the Alternative Sorting Algorithm. The researchers will also add a Burst Time-based Dynamic Time Quantum.

1. **LITERATURE REVIEW**

Combining the Round Robin with other Algorithms can enhance the algorithm. The order of the queues affects the work time for each process and decrease the arrival for each processes. (Hyytiä & Aalto, 2016)

Minimizing Waiting Time is one of the important criteria. By minimizing the minimizing both the average Waiting Time and Turnaround Time, the context switching can be lessened which means work on each process will be done faster. (Samih Mohemmed Mostafa, 2010)

Using the right Time Quantum for the Round Robin Scheduling Algorithm can help improve the context switches of the processes. If done correctly, this can also improve the Average Waiting Time and Turnaround Time.

Optimizing Round Robin Scheduling Algorithms by using a Dynamic Quantum that is based on Burst Time, can drastically decrease the number of context switching, also reducing the Average Waiting and Turnaround Time.

1. **METHODOLOGY**

The researchers will be able to improve the time efficiency by adding the Alternative Sorting to Round Robin Algorithm and as well as implementing a Dynamic Time Quantum for time splicing.

Alternative Sort()

       For i = 0 to #OfProcesses/ 2

            If i is 0 Then

                Array[ i ] Gets  Processes[n - (i + 1)];

                Array[i + 1]  Gets Processes[i];

            Else

                Array[i + counter] Gets Processes[n - (i + 1)];

                Array[i + (counter + 1)] Gets Processes[i];

            End If

            counter += 1;

            If Total number of Processes is an Odd

                Array[n - 1] Gets Processes[(n / 2)];

End If

The Alternating Sort has a time complexity of [3 + 5(n/2)] || [4 + 5(n/2)],  O.G (n)

and space efficiency of 10 bytes

Quantum Time()

        If #OfProcesses is greater than 2 Then

 QuantumTime Gets (MaxBurstTime - MinBurstTime) / 2;

       Else

            QuantumTime  Gets MinBurstTime;

       End If

1. **FIGURES/CAPTION**

The researchers assumed 5 processes for each test cases. Test case 1 had its Burst Time values arranged in an ascending manner. Test case 2 had its Burst Time values arranged in a descending order. Lastly, test case 3 had no arrangement but had relatively high Burst Time Values.

All results were computed manually and mathematically in order to remove the bias of a used CPU and processor.

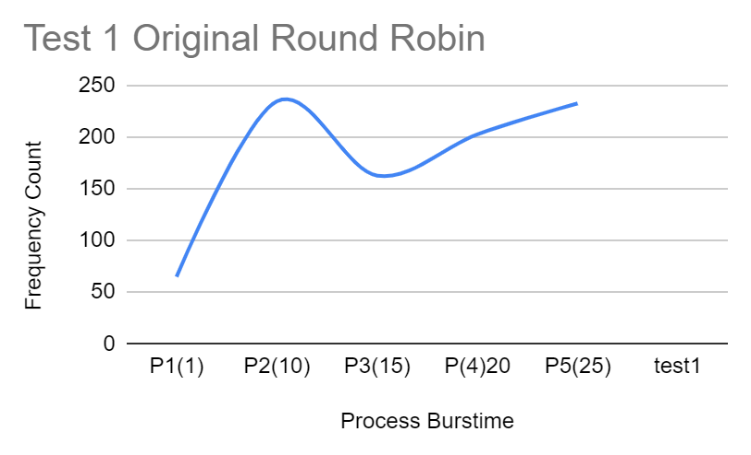
**Test Case 1 (Ascending Burst Time)**

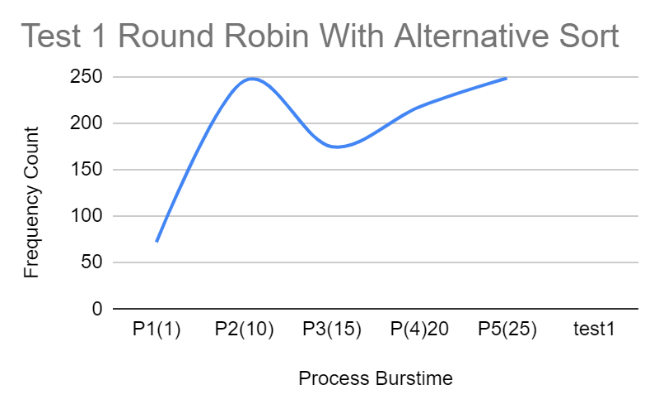
The results in Table 1 show that the Round Robin Algorithm with the implemented Alternative Sorting performed relatively worse when compared to the original.

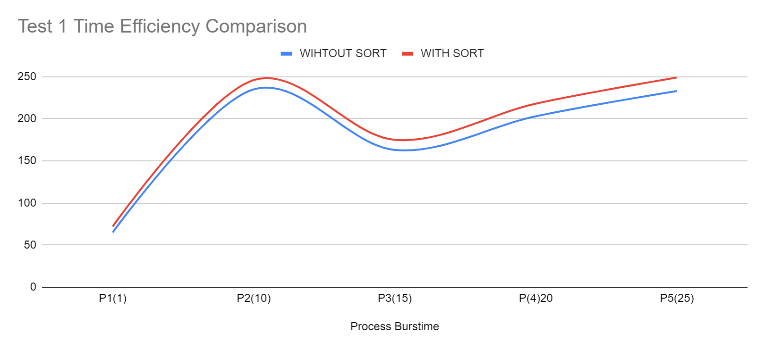
The arrangement of the Burst Times in ascending order does not affect the outcome of the values in a positive manner.

**Table 1** Test Case 1

|  |  |  |  |
| --- | --- | --- | --- |
| Process | Burst Time | Original | With sorting |
|  | Process Burst Time | Frequency Count | Frequency Count |
| p1 | P1(1) | 65 | 72 |
| p2 | P2(10) | 235 | 246 |
| p3 | P3(15) | 163 | 175 |
| p4 | P4(20) | 203 | 218 |
| p5 | P5(25) | 233 | 249 |







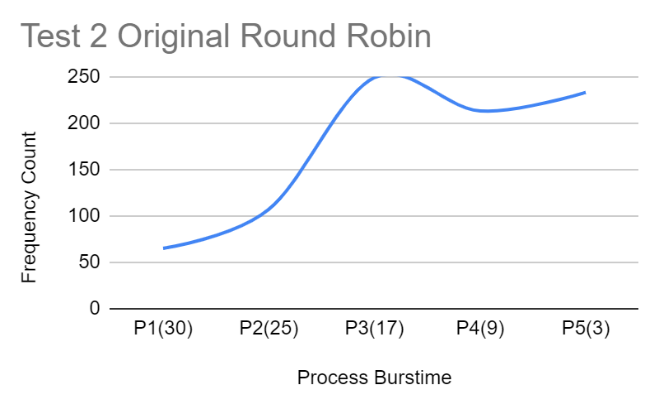
**Test Case 2 (Descending Burst Time)**

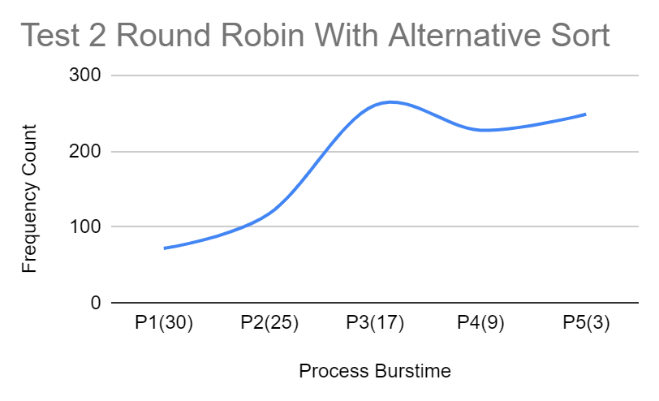
The results in Table 2 show that the implemented Alternative Sorting Algorithm also performs worse compared to the original.

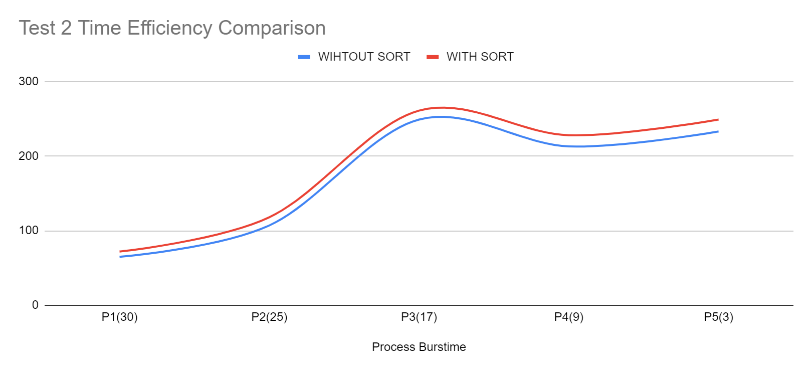
Arrangement of the Burst Time values in descending order also does not positively affect the outcome of the values.

**Table 2** Test Case 2

|  |  |  |  |
| --- | --- | --- | --- |
| Process | Burst Time | Original | With sorting |
|  | Process Burst Time | Frequency Count | Frequency Count |
| p1 | P1(30) | 65 | 72 |
| p2 | P2(25) | 107 | 118 |
| p3 | P3(17) | 249 | 261 |
| p4 | P4(9) | 213 | 228 |
| p5 | P5(3) | 233 | 249 |

****

****

****

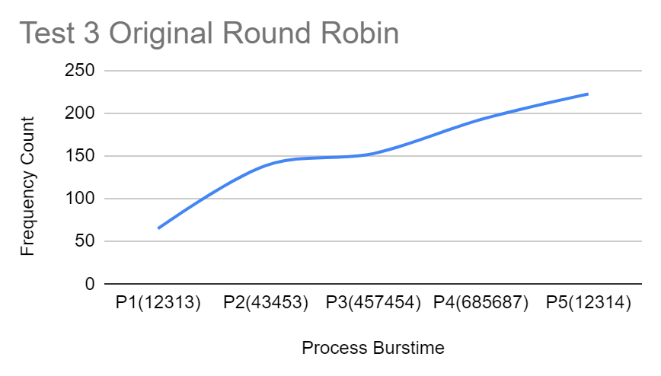
**Test Case 3 (High Value Burst Time)**

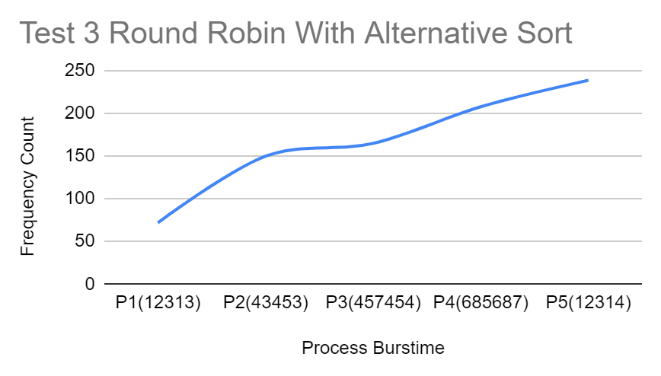
The final test case also shows that implementing the Alternative Sorting Algorithm to the Round Robin Algorithm does not make it perform better than the original.

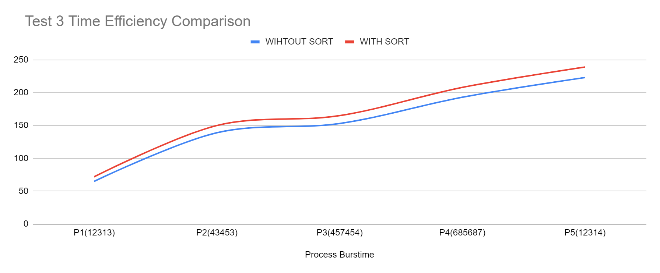
The intensity of the values used for the Burst Time also do not affect the outcome in a positive manner.

**Table 3**

|  |  |  |  |
| --- | --- | --- | --- |
| Process | Burst Time | Original | With sorting |
|  | Process Burstime | Frequency Count | Frequency Count |
| p1 | P1(12313) | 65 | 72 |
| p2 | P2(43453) | 139 | 150 |
| p3 | P3(457454) | 153 | 165 |
| p4 | P4(685687) | 193 | 208 |
| p5 | P5(12314) | 223 | 239 |

****

****

****

1. **CONCLUSION**

The implementation of the Alternative Sorting Algorithm does not enhance the Time Efficiency of the Round Robin Algorithm. It is relatively slower when compared to the original when processing.

The researchers recommend fixing the Time Quantum used in the algorithm. It is not completely dynamic since it only computes the Time Quantum in its very first instance. This lengthens the processing time since it does not adapt to the current number of processes.

The researchers would also recommend implementing other types of sorting algorithms. Doing so may affect the outcomes of the test cases.

1. **ACKNOWLEDGEMENT**

The researchers would also like to thank Sir Jeffrey S. Ingosan for aiding them in the creation of this paper.

1. **REFERENCES**

Manuel, Baquirin, Guevara, & Tandingan Jr. 2019

Fittest Job First Dynamic Round Robin (FJFDRR) scheduling algorithm using dual queue and arrival time factor: a comparison https://iopscience.iop.org/article/10.1088/1757-899X/482/1/012046/pdf

(2019)

Shyam & Nandal, 2014

Improved Mean Round Robin with Shortest Job First

Scheduling

http://ijarcsse.com/Before\_August\_2017/docs/papers/Volume\_4/7\_July2014/V4I7-0103.pdf

(July, 2014)

Mishra & Rashid, 2014

An Improved Round Robin CPU Scheduling Algorithm with Varying Time Quantum

https://www.academia.edu/9461392/An\_Improved\_Round\_Robin\_CPU\_Scheduling\_Algorithm\_with\_Varying\_Time\_Quantum

(August, 2014)

Mythili.N Swaraj Pati; Korde Pranav ; Dey Pallav 2017

An advanced approach to traditional round robin CPU

scheduling algorithm to prioritize processes with residual

burst time nearest to the specified time quantum

https://iopscience.iop.org/article/10.1088/1757-899X/263/4/042001/pdf

(2017)

Dash, Sahu, & Samantra, 2015

An Optimized Round Robin CPU Scheduling Algorithm with Dynamic Quantum

https://www.academia.edu/34442483/AN\_OPTIMIZED\_ROUND\_ROBIN\_CPU\_SCHEDULING\_ALGORITHM\_WITH\_DYNAMIC\_TIME\_QUANTUM

Indursree, J. R. and Prabadevi, B. 2017

Enhanced Round Robin CPU Scheduling with Burst Time-based Time Quantum

https://iopscience.iop.org/article/10.1088/1757-899X/263/4/042038/pdf

(2017)

Behera, H. S., Mohanty R., and Nayak, D. 2010

A New Proposed Dynamic Quantum with Re-Adjusted Round Robin Scheduling Algorithm and Its Performance Analysishttps://www.researchgate.net/publication/45601967\_A\_New\_Proposed\_Dynamic\_Quantum\_with\_ReAdjusted\_Round\_Robin\_Scheduling\_Algorithm\_and\_Its\_Performance\_Analysis

(August 2010)

Hyytiä & Aalto, 2016

On Round-Robin routing with FCFS and LCFS scheduling

https://www.sciencedirect.com/science/article/pii/S0166531616000031#s000160

(March, 2016)

Rida, S., Hamad, S., and Mostafa, S. 2010

Improving Waiting Time of Tasks Scheduled Under Preemptive Round Robin Using Changeable Time Quantum

https://www.researchgate.net/publication/45908729\_Improving\_Waiting\_Time\_of\_Tasks\_Scheduled\_Under\_Preemptive\_Round\_Robin\_Using\_Changeable\_Time\_Quantum

(Mar. 2010)