AN ENHANCE ROUND ROBIN SCHEDULING ALGORITHM USING ADAPTIVE QUANTUM TIME

Aisha Umar Musa[1], Department of computer science
Dr. Ibrahim Abdullahi[2], Department of computer science
Professor A.E okeyinka[3], Department of computer science
Dr Adamu isah[4], Department of computer science
Dr. Adamu I abubakar[5], Department of computer science

Abstract

Scheduling is considered as an integral component of multi-processing, in recent years many scheduling algorithm are been design to tackle the limitation in multi-task environment and bad scheduling. Algorithm such as First Come First Serve, Priority Algorithm, Shortest Job first, Round Robin algorithm and the rest has been proposed to solve task scheduling. However, all the listed algorithm all aim at maximizing throughput that number of processes executed completely within a unit time, minimizing the Context switch, completion time, and waiting time of process. Based on this project emphasis is made on improving Round Robin algorithm. In this paper work we proposed a new Adaptive Quantum Time Round Robin Scheduling (thus the AQTRRSA) algorithm to achieve our aim. Analysis, simulation and result shows that the proposed algorithm is more efficient than that of the existing method in term of ATA, AWT, and number of Context Switch. in conclusion the AQTRRSA algorithm produce a better result.

INTRODUCTION

An operating system is a piece of computer software that helps in managing the hardware component of a computer system, including basic and complex operation of the computer system. I can also be explained as the window or interface between a computer system and the user [1]. The fundamental techniques process which is responsible for deciding and organizing jobs or task to be carry out by the system, is called scheduling. However, they are two main scheduling benchmarks for scheduling this include; Pre-emptive (thus, order process or task in the ready queues are block the a much higher priority process), and the non-pre-emptive algorithm (thus, process is executed based on first come first serve, no priority given to any task). The CPU scheduling process used within the operating system environment has been implemented using various proposed approach by researchers in the domain, this algorithm includes; the Shortest Remaining Time first, Fixed, priority scheduling, priority scheduling, Round Robin Scheduling algorithm, and multilevel Queue scheduling [2]. When an interrupt occurs the computer processor

experience a periodically interruption. hence, the process currently been executed by the processor is been place in the Ready Queue (RQ), and the next process schedule next in the Ready Queues are selected for execution. In this Method of execution, the algorithm is said to be time sharing or time slicing, because each process in the ready queue is been allocated with a certain amount of unit time (CPU clock) before it is been pre-empted. In the research work of (Khalil *et al.,* 2013) [3] which make it known that the most difficult aspect of Round Robin algorithm is the specific of CPU time (thus, amount of Quantum time) that will be allocated to task before it is being pre-empted. According to research it is known that RRA shows outstanding performance in term of scheduling process, in respect to other

scheduling algorithm mention earlier in terms of time-sharing operation system. However, the performance of deferent scheduling algorithm is based on some certain property or criteria, thus the Turnaround Time (TAT) of a process, which is define as the total time that is present between the arrival of a process and when a process will be completed, While the Waiting Time (WT) of a process is define as the time it take a task to be on hold until the CPU resources is available, Response time (RT) of CPU is the time taken to execute process in the ready queue, The CPU Throughput, and finally the CPU Utilization (thus, the busy time of a central processing unit measured in percentage) [2].

Furthermore, this paperwork aims in enhancing existing dynamic round robing algorithm using adaptive approach. Numeric analysis, with simulated code and result will be discussed in this paper work.

LITERATURE REVIEW

A novel scheduling algorithm based on Ant Lion Optimizer was introduce by (Dinkar & Deep, 2019). In the paper work it was introduced that within a multiprogramming operating system environment, the ant lion optimizer plays an important role in scheduling various process or task. The enhancement in number of throughput and minimal waiting time is achieved by efficiently placing processes in the ready queue. Based on the dynamic approach used for time slice enable the schedular to efficiently schedule process or task within a reasonable amount of time. The researcher introduces a new optimized techniques which is novel nature inspired, thus the Ant Lion Optimizer (ALO). This proposed algorithm shows good result in such a way the average waiting time of a process is drastically minimized [4].

Considering the paper work of [5] titled *comparison of various round robin scheduling algorithm.* In this paperwork various round robin scheduling algorithm are compare, this comparison is done to ensure that the improve type of the RRA is better than that of the traditional version. The researcher was able to carry out comparison on various RRA based on their average waiting time, average turn around time and finally the number of context switch. The intent of the researcher is to reveal the truth about the modify round robin algorithm and the traditional RRA [5].

An optimizer technique was introduced by [6] Kumar, 2014 to minimize context switching of round robin algorithm. The researcher specifies that almost all the computer resource are schedule prior before using. It was declared that all effort of other researcher is based on improving the CPU throughput, minimized waiting time and turnaround time of a process. Hence, to improve on the performance, and to minimize the overhead cost of the CPU or central processing unit, then the quantum time should be should be large based on the number of contexts switching. Otherwise, this would lead to small number of context switch [6].

In respect to the paper work Khalil, 2013 titled; *SWRR: the scheduling algorithm based on weighted round robin.* It is stated that they are rapid growth in people paying more attention to data, information or privacy protection on the internet. Tor is considered as the most widely and frequently used communication system due to is anonymous us. However, the system of communication can be efficiently adopted in protecting the privacy of users on the interment. The link scheduling (thus, Tor) approach proposed in recent year causes network congestion. Hence, the research present new techniques for link scheduling algorithm thus, SWW which is based on WRR (thus, the weighted round robin). A comparison experiment was made under various congestion condition, and the

proposed scheduling algorithm is verified to be efficient and effective in comparison to the existing one [3].

Srujana et al., 2019 proposed a *sorted round robin algorithm.* In the paper work it is explained that scheduling process is one of the most necessary and important tasks in an environment that is multiprogramming in nature. The process manager is in charge of managing each and every process, by deciding on which process to execute next using certain scheduling technique. With Round Robin algorithm individual task is been allocated with a specific CPU time slice for execution. In this paper work the researchers hybridize Shortest Job First (SJF) and the Round Robin algorithm (RR). Based on the merge techniques, it provides good advantages on the overall performance of the CPU and try to solve the drawback in using only RRA. The result of the research shows that the merge techniques outperformed the traditional RRA [7].

METHODOLOGY: ADAPTIVE QT ALGORITHM

This section present will discuss in detail the working structure of the proposed Adaptive Quantum time round robin scheduling algorithm. The proposed scheme use a dynamic or adaptive quantum based on attribute or condition, unlike the exiting tradition RRA that employ static or fixed CPU time slice to execute process. In us propose approach the quantum time adapt itself based on the available process and burst time of each process. The section will comprehensive explain the proposed *Adaptive Quantum time scheduling algorithm.*

Adaptive Quantum Time Algorithm

The following pseudocode explain the proposed Adaptive techniques, from the initialization of AT, WT, BT and ready Queue. Follow by the precomputation of quantum time for every context switching depicted with the for-loop block. Finally, the output such as number of context switch, average waiting time, average turnaround time are generated for every Context Switch.

AQTRRSA

```
1. Input: process, arrival_time, burst_time
2. Initialize local var:
3.      completion_time = 0
4.      turnaround_time = 0
5.       waiting_time = 0
6.      process_id = process_index
7.      adaptive_quantum_time = 0
8.      Initialize general var:
9.      average_waiting_time = 0
10.     average_turnaround_time = 0
11.     num_context_switch = 0
12.     ready_queue = null
13.      If all process BT == 0
14.             sort_process(using burst_time)
15.             Initialize ready_queue
16.      Else
17.             sort_process(using arrival_time)

18.     max_BT = sort_process[0] // process with the highest
burst_time
19.     Max2_BT = sort_process[1]// next highest process

20.     If number of process > 1
21.     adaptive_quantum_time = ((max_BT + max2_BT) / 2 ) +
(number_of_process/2)
22.     Else if number of process == 1
23.     adaptive_quantum_time = (max_BT/1) +
number_of_process/1
24.     Start_schedulling
25.     While process in ready_queue:
26.     Check if adaptive_quantum_time < process burst_time
27.     If YES
28.     Execute process
29.     burst_time=process burst_time - adaptive_quantum_time
30.     Process new burst_time = burst_time
31.     // process are not completly executed
32.     new_ready_queue.update(incomplete_process)
33.                     If NO
34.             Execute process
// process are completely executed

35.     Restart schedulling if ready_queue !=[]//restart process if
ready_queue is not empty
36.             start_schedulling
37.     Else
38.             // schedulling completed
39.             Compute
40.             average_turnaround_time =
total_turnaround_time / number_of_process
50.             average_waiting_time = total_waiting_time /
number_of_process
51.             PRINT:
```

52. AVG_TAT (average_turnaround_time)
53. AVG_WT (average_waiting_time)
54. QT (adaptive_quantum_time list)
55. CS (context_switch)
*56. Print: **EXECUTION COMPLETED***

In addition, AQTRRSA algorithm accept N number of process *p1*,*p2*,*p3*, and look into certain feature of the processes such as AT, BT. The algorithm uses the burst time to in arranging the process in an orderly manner. The main strength of this proposed algorithm is how the quantum time are re-computed, by dividing into two the sum of two process with the highest burst time. and finally add it up with the half to the total number of processes in the ready queue. The quantum time is recomputed for every context switch.

AQT (Adaptive Quantum Time)
$AQT_{(ms)} = (Max\_BT1 + Max\_BT2\ /2) + (No.\ process\ /\ 2)$
For one process in ready queue:
$AQT_{(ms)} = (Process\_BT\ /2) + (No.\ process)$

The mathematical expression shows condition of how the QT is computed when a process is one and more that one in the ready queue.

RESULT AND DISCUSSION

This section will show the mathematical analysis, computed simulation design using python language withing PyCharm environment. Two cases of sample process list will be considered in this paper work. The proposed algorithm with be numerical compared to existing approach such as the

i. AN algorithm [8].
ii. IRRVQ algorithm [8].
iii. RRDTQ algorithm [8].

CASE I

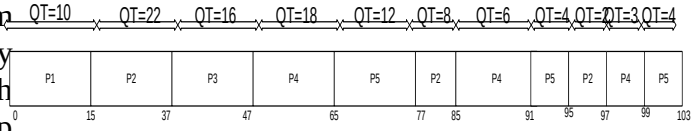
Considering the case each process (thus process 1 to 5) in the ready queue arrived at the same time.

Table. 1

Processes	Arrival Time	Burst Time (ms)
P1	**0**	**15**
P2	**0**	**32**
P3	**0**	**10**
P4	**0**	**26**
P5	**0**	**20**

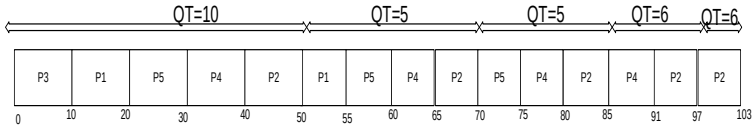
(Sohrawordi et al., 2019)

Gant Chart for AN (fig. 1)



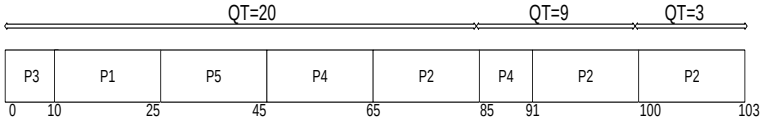
Time (ms)
AN algorithm
(Sohrawordi et al., 2019)

Gant Chart for IRRVQ Algorithm (Fig. 2)



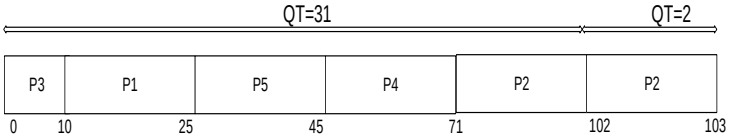
Time (ms)
IRRVQ Algorithm
(Sohrawordi et al., 2019)

Gant Chart for RRDTQ Algorithm (Fig. 3)



Time (ms)
RRDTQ algorithm
(Sohrawordi et al., 2019)

Gant Chart Proposed AQTRRSA (Fig. 4)



Time (ms)
Proposed: AQTRRA algorithm

According to the diagram above its clearly show that the number of context switch

reduces in the proposed adaptive quantum time algorithm.

Based on the Gantt chart in figure 4. Showing process p1 to p5 thus process to be execution, it also showcases the quantum time generated within a CS. In respect to the chart is can be concluded that execution of the all process is within two context switches, and the completion time for each process. The mathematical expression bellow explains how the time quantum is generated for each context switch, by using the two highest burst time and all floating value are round down to the nearest whole number. If process in the ready queue is more than one, the two highest burst time are added then they are divided by two, otherwise the single process in the ready uses is burst time only and add to one (thus, number of remaining process).

Quantum time (QT) = ((max_B1 + max_B2) / 2) + ((total No. of process)/2)
QT = ((32 + 26)/2) + (5/2) = 29 + 2 = 31ms (context switch=1)
For context switch two the ready queue contains only one process
QT = (1/1) + (1/1) = 2ms (context switch).

Execution Summary of proposed AQTRRSA algorithm

Table. 2

Process	AT	BT	CT	TAT	WT
P1	0	15	25	25	10
P2	0	32	103	103	71
P3	0	10	10	10	0
P4	0	26	71	71	45
P5	0	20	45	45	25
				Avg TAT=50.8	Avg WT=30.2

CT = can be easily seen on the Gantt chart
TAT (turnaround time) = CT (completion time) – AT (arrival time)
WT (waiting time) = TAT (turnaround time) – BT (burst time)
Average Turnaround Time (Avg TAT) = (25+103+10+71+45) / 5 = 254/5 = 50.8ms
Average Waiting Time (Avg WT) = (10+71+0+45+25) / 5 = 151/5 = 30.2ms

Comparative analysis of (RR, AN, IRRVQ, and AQTRRSA)

Table. 3

Algorithm	Time Quantum (TQ)	Arg Waiting Time (ms)	Avg Turnaround Time (ms)	Context Switch
Traditional RR	10, 5	54.2, 56.2	74.8, 76.8	11, 21
AN (Noon et al., 2011)	22, 22, 16, 18, 12, 8, 6, 4, 2, 3, 4	51.6	72.2	10
IRRVQ (Mishra & Rashid, 2014)	10, 5, 5, 6, 6	46.2	66.8	14
RRDTQ (Sohrawordi et al., 2019)	20, 9, 3	35.4	54.8	6
Proposed AQTRRA	31, 2	30.2	50.8	2

the table above show the comparative result of the proposed algorithm in respect to the existing round robin algorithm. And it's

clearly show that the proposed method outperforms the existing approach.

CASE II

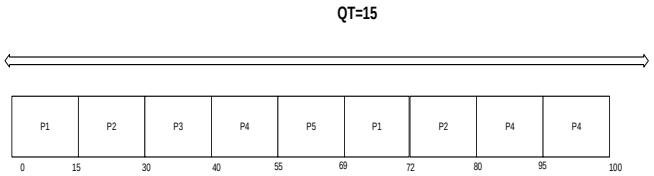
Considering this scenario where process arrive at different arrival time. Gantt chart and tabular comparisons are show bellow.

Table. 4

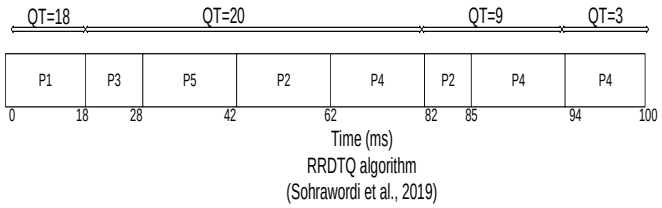
Processes	Arrival Time	Burst Time (ms)
P1	**0**	**18**
P2	**3**	**23**
P3	**4**	**10**
P4	**8**	**35**
P5	**10**	**14**

(Sohrawordi et al., 2019)

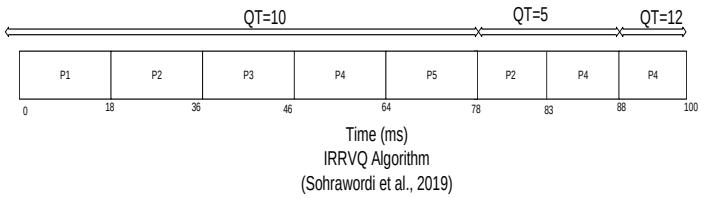
Gantt Chart for RR (fig. 5)



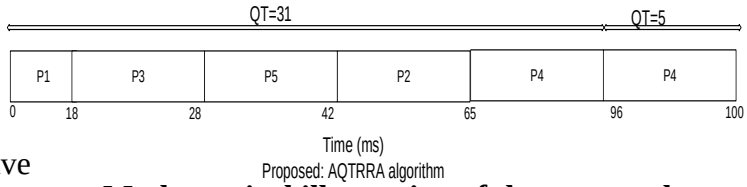
Gant Chart (RRDTQ) fig. 6



Gantt Chart (IRRVQ) fig. 7



Gantt Chart (Proposed AQTRRSA) fig. 8



Mathematical illustration of the proposed AQTRRSA algorithm

Quantum time (QT) = ((max_B1 + max_B2) / 2) + ((total No. of process)/2)
QT = ((35 + 23)/2) + (5/2) = 29 + 2 = 31ms (context switch=1)
For context switch two the ready queue contains only one process
QT = (4/1) + (1/1) = 5ms (context switch =2)

Execution Summary of proposed AQTRRSA algorithm

Table. 5

Process	AT	BT	CT	TAT	WT
P1	0	18	18	18	0
P2	3	23	65	62	39
P3	4	10	28	24	14
P4	8	35	100	92	57
P5	10	14	42	32	18
				Avg TAT=45.6	Avg WT=25.6

The table above shows the execution the AT, BT, CT, TAT, and WT of each process. With the overall average turnaround time and Average waiting time.

CT = can be easily seen on the Gantt chart
TAT (turnaround time) = CT (completion time) – AT (arrival time)
WT (waiting time) = TAT (turnaround time) – BT (burst time)
Average Turnaround Time (Avg TAT) = (18+62+24+92+32) / 5 = 228/5 = 45.6ms

Average Waiting Time (Avg WT) = (0+39+14+57+18) / 5 = 128/5 = 25.6ms

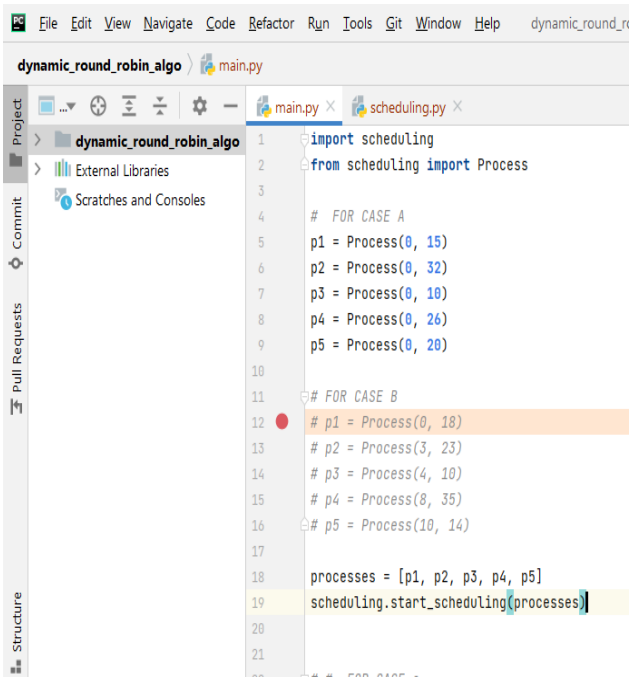
Tabular comparison of existing RR and the proposed AQTRRSA

Table 6.

Algorithm	Time Quantum (TQ)	Arg Waiting Time (ms)	Avg Turnaround Time (ms)	Context Switch
Traditional RR	10, 15	46, 49.2	66, 67.2	10, 7
AN (Noon et al., 2011)	18, 20, 15, 17, 11, 8, 10, 5, 8	40.4	62.4	8
IRRVQ (Mishra & Rashid, 2014)	18, 5, 12	42.8	60	6
RRDTQ (Sohrawordi et al., 2019)	18, 20, 9, 3	29.6	49.2	6
Proposed AQTRRA	31, 5	25.6	45.6	2

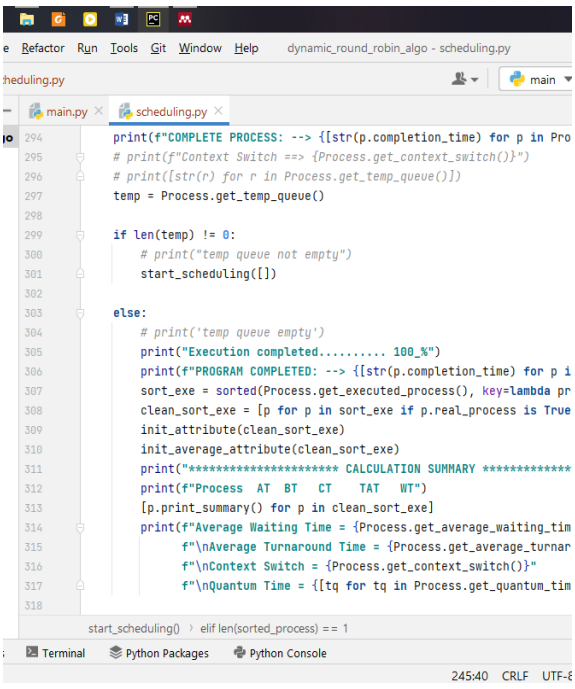
AQTRRSA Simulation Result using python.

Fig.9 Simulating AQTRRSA



The image above is showing process initialization and the method call for starting scheduling, withing the PyCharm environment.

Fig.10 Simulating AQTRRSA



The figure above shows fragment of the python method performing scheduling process

CONCLUSION

Considering the analysis and evaluation that is carry out on existing round robin algorithm and our proposed AQTRRSA algorithm, we can deduce that our proposed scheduling algorithm is more efficient than that of the existing algorithms.

REFERENCE

[1] A. Alsheikhyl, R. Ammar, and R. Elfouly, “An Improved Dynamic Round Robin Scheduling Algorithm Based on a Variant Quantum Time,” pp. 98–104, 2015.

[2] C. Algorithm, W. Dynamic, and T. Quantum, “A N O PTIMIZED R OUND R OBIN CPU S CHEDULING,” vol. 5, no. 1, pp. 7–26, 2015.

[3] Z. H. Khalil, A. Basim, and A. Alaasam, “Priority Based Dynamic Round Robin with Intelligent Time Slice and Highest Response Ratio Next Algorithm for Soft Real Time System,” vol. 2, pp. 120–124, 2013.

[4] S. K. Dinkar and K. Deep, *A Novel CPU Scheduling Algorithm Based on Ant Lion Optimizer*. Springer Singapore.

[5] A. Joshi and S. B. Goyal, “Comparison of Various Round Robin Scheduling Algorithms,” pp. 18–21, 2019.

[6] M. K. M. R, R. R. B, M. Sreenatha, and C. K. Niranjan, “AN IMPROVED APPROACH TO MINIMIZE CONTEXT SWITCHING IN ROUND ROBIN SCHEDULING ALGORITHM USING OPTIMIZATION TECHNIQUES,” pp. 804–808, 2014.

[7] R. Srujana, M. D. Sai, and K. Mohan, “Sorted Round Robin Algorithm,” *2019 3rd Int. Conf. Trends Electron. Informatics*, no. Icoei, pp. 968–971, 2019.

[8] I. J. A. Res, A. M. Round, R. Cpu, S. Algorithm, W. Dynamic, and T. Quantum, “Manuscript Info Abstract measure of multitasking operating system which makes a the processes . The performance of the CPU scheduling scheduling algorithm in multitasking operating system . The efficiency of a multitasking systemcomprising with Round Robin CPU scheduling relies onthe selection of the optimal time quantum . If the time quantum is longer , the response time of the processes . In this paper , a modified CPU scheduling algorithm , called Round Robin with Dynamic Time Quantum (RRDTQ) is introduced for enhancing CPU performance using dynamic time quantum with RR . This time quantum is calculated from the Introduction : - ISSN : 2320-5407,” vol. 7, no. 2, pp. 422–429, 2019, doi: 10.21474/IJAR01/8506.