2581-4575
International Journal For Recent Developments in Science & Technology
Crossref
A Peer Reviewed Research Journal

MODIFIED ROUND ROBIN ALGORITHM FOR TASK SCHEDULING

JOEL JOSEPHSON P[1], MATTA JAGADEESH CHANDRA PRASAD[2]

[1]Assistant Professor, Department of Electronics and Communication Engineering, Narsimha Reddy College of Engineering, Hyderabad, T. S., India

[2]Professor and Head of the Department of Electronics and Communication Engineering, Narsimha Reddy College of Engineering, Hyderabad, T. S., India

E-Mail ID: joeljosephsonp@gmail.com, jagadishmatta@gmail.com

ABSTRACT

Cooperative booking calculation is a preemptive CPU planning calculation which switches between the procedures when static time quantum terminates. In Cooperative booking calculation the time quantum is fixed and process are planned for such a way, that no procedure get CPU time more than one time quantum in one go. Cooperative planning calculation is structured particularly for time sharing working framework however it has its hindrances that are its More drawn out Normal Holding up Time, Higher Setting Switches and Higher Turnaround Time. In this booking calculation the principle thought is to modify the time quantum powerfully so that perform preferable execution over Cooperative planning calculation.

Keywords: Operating System, Round Robin, Average Mid Max Round Robin, Turnaround time, Waiting time, Context Switch.

1. INTRODUCTION

A working framework is a program that deals with the PC equipment. It gives a stage where client can interface with equipment and execute program in an effective way. Current working frameworks have gotten increasingly perplexing, they have developed from a solitary assignment to a performing various tasks condition in which procedures run in a simultaneous way. In performing multiple tasks and multiprocessing condition the manner in which the procedures are relegated to run on the accessible CPUs is called planning. Assigning CPU to a procedure requires cautious attention to guarantee equity and keep away from process starvation for CPU. The principle objective of the planning is to augment the diverse presentation measurements viz. CPU usage, throughput and to limit reaction time, holding up time and turnaround time and the quantity of setting switches.

2. PRELIMINARIES

A program in execution is known as a procedure. The ideal opportunity for which a procedure holds the CPU is known as burst time. The time at which a procedure lands for execution is its appearance time. Turnaround time is the measure of time to execute a specific procedure, while holding up time is the measure of time a procedure has been holding up in the prepared line. Time lapsed from the accommodation of a solicitation by the procedure till its first reaction is characterized as the reaction time. Scheduler chooses a procedure from lines in a way, for its execution with the end goal that the heap balance is successful. In non-

2581-4575

International Journal For Recent Developments in Science & Technology

Crossref

A Peer Reviewed Research Journal

acquisition, CPU is appointed to a procedure, it holds the CPU till its execution is finished. In any case, in seizure, running procedure is compelled to discharge the CPU by the recently shown up process. In time sharing framework, the CPU executes different procedures by exchanging among them quick. The occasions CPU changes starting with one procedure then onto the next is called as the quantity of setting switches. At the point when procedure goes into the primary memory and are prepared and standing by to execute are kept in the information structure called Prepared Line. At the point when a procedure appoint to the CPU, it execute or while hanging tight for some occasion to happen. The procedures which are sitting tight for I/O demand are kept in Gadget Line. The Long haul scheduler or employment scheduler select procedure from work pool and burden them into principle memory for execution. Momentary scheduler or CPU scheduler select from among the procedures that are prepared to execute and assigns the CPU to one of them. Medium term scheduler is utilized in time sharing framework. The fundamental favorable position of Medium term scheduler is now and again it expel forms from principle memory and in this way decrease level of multiprogramming. Later the procedure can be reintroduced into memory, and its execution can be proceeded with the latest relevant point of interest. This plan is called as swapping.

3. SCHEDULING ALGORITHM

CPU booking calculation chooses which of the procedures in the Ready Queue (RQ) are to be apportioned to the CPU. There are a wide range of CPU planning calculations utilized like First Come First Serve (FCFS), Shortest Job First (SJF),Round Robin (RR),Priority booking calculation and Short Remaining Time Next (STRN) calculation. The procedures are booked by the given burst time, appearance time, time quantum and need.

3.1 RR Scheduling Algorithm

Cooperative effort (Round Robin) is the most seasoned, easiest and most broadly utilized relative offer booking calculation. It is like FCFS booking, yet seizure is added to switch between forms. In Round Robin calculation a little unit of time cut are required which is called Time Quantum (TQ). The CPU scheduler circumvents Ready Queue and assigns the CPU to every procedure by the assistance of Dispatcher for a period interim of up to 1 Time Quantum (TQ). On the off chance that new procedure shows up, at that point it is added to the tail of Circular Queue. The CPU scheduler picks the primary procedure from the Ready Queue sets a clock to hinder after one Time Quantum and dispatches the procedure. After TQ is lapsed, the CPU seizes the procedure and the procedure is added to the tail of the Circular Queue. In the event that the procedure completes before the finish of the TQ, the procedure itself seizes the CPU eagerly. In this paper, we attempted to take care of the Time Quantum issue by altering the Time Quantum Dynamically regarding the existed set of procedures in Ready Queue.

4. PERFORMANCE METRICS

The proposed calculation is intended to meet all booking criteria, for example, most

International Journal For Recent Developments in Science & Technology
2581-4575
Crossref
A Peer Reviewed Research Journal

extreme CPU usage, greatest throughput, turnaround time, least holding up time and setting switches. Here we are thinking about three execution criteria for each situation of our analysis.

Turnaround Time (TAT)=Finish Time– Arrival Time. Normal Turnaround Time ought to be less

Holding up Time (WT)= Start Time- Arrival Time. Normal Waiting Time ought to be less.

Setting Switch The quantity of setting Switch ought to be less

5. RELATED WORK

Over the most recent couple of years various methodologies are utilized to build the exhibition of Round Robin planning like Adaptive Round Robin Scheduling utilizing Shortest Burst Approach Based on Smart Time Slice[1], Multi-Dynamic time Quantum Round Robin (MDTQRR)[5].Min-Max Round Robin (MMRR)[2], Self-Adjustment Time Quantum in Round Robin (SARR)[10], Dynamic Quantum with Re-balanced Round Robin (DQRRR)[11],Average Max Round Robin Algorithm (AMRR)[8].

6. PROPOSED APPROACH

Let's expect that the burst time of the procedures is taken as arranged expanding request with the goal that it will give better turnaround time and holding up time. For the most part in Round Robin calculation the presentation relies on the size of fixed or static Time Quantum (TQ). On the off chance that TQ is excessively huge, at that point Round Robin calculation rough to First Come First Served (FCFS). On the off chance that the Time Quantum is excessively little, at that point there will be numerous setting exchanging between the procedures. Along these lines, our methodology tackled this issue by taking a dynamic TQ. Where TQ is the Average of the summation of Mid and Max process.

Mid=(Min + Max)/2 TQn=(Mid+ Max)/2

7. PROPOSED ALGORITHM

In our proposed calculation, forms are as of now present in the Ready Queue (RQ). Naturally, Arrival Time (AT) is alloted to zero. The quantity of procedures „n" and CPU Burst Time (BT) are acknowledged as info and Average Turnaround Time (ATT), Average Waiting Time (AWT) and number of Context Switch (CS) are created as yield. Let TQn be the new time quantum. The pseudo code for the calculation is displayed in Figure 1 and the flowchart of the calculation is exhibited in Figure 2.

All the processes in the Ready Queue are sorted in ascending order.

```
Where
// N= Number of Processes in the ready
queue
// BT= Burst Time of the Processes
While(RQ != NULL)
// RQ= Ready Queue
//Mid= Mid of all the processes in the
Ready Queue
//Min= First process in the Ready
Queue having minimum Burst Time
// Max=Last process in the Ready
Queue having maximum Burst Time
Mid=(Min + Max)/2
//(Use Use round off function in Mid)
TQn=(Mid+ Max)/2
//TQn=New Time Quantum
//(Use round off function in TQ)
```

International Journal For Recent Developments in Science & Technology
2581-4575
Crossref
A Peer Reviewed Research Journal

(Remaining Burst time of the process)
//If one process is there then after calculation TQn is equal to BT itself
// Assign TQ to (1 to n) processes
for i=0 to N loop
{
Pi->TQn
}
End of for
//Assign TQn to all the available processes
Calculate the remaining Burst time of the processes.
If (new process arrived and BT!=0 or new process is arrived and BT==0)
then go to step1
else if (new process
is not arrived and
BT!=0) then go to
step 2
elsego to step 6 end of if end of while
Calculate ATT,AWT,CS
//ATT=Average Turnaround time
//AWT=Average waiting time
//CS=Number of context switch
End

Figure 1 Pseudo code for average mid max round robin Algorithm

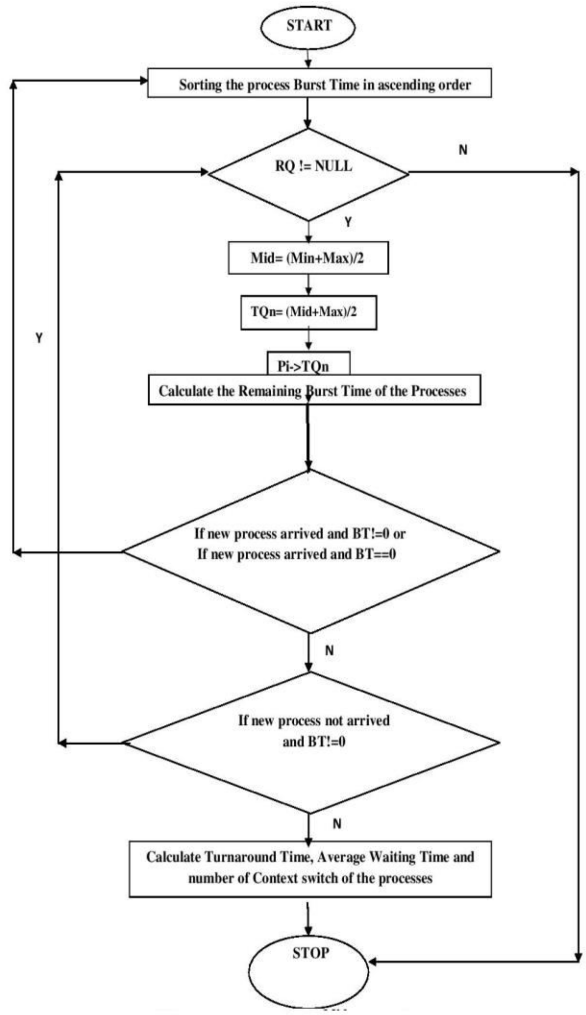


Figure 2 Flow Chart of average Mid Max Round Robin (AMMRR)

8. ILLUSTRATION

Given the burst time of the process in unsorted sequence:(P1=57,P2=69,P3=45,P4=20,P5=85) taking arrival time=0. Initially the burst time of all the processes were sorted in ascending order which resulted in sequence : (P4=20,P3=45,P1=57,P2=69,P5=85). Then the burst time of Mid process is find which is P1=57 by using the formula Mid=(Min+Max)/2. After that TQn is calculated. Where TQn is the Average

International Journal For Recent Developments in Science & Technology

2581-4575

Crossref

A Peer Reviewed Research Journal

of the summation of Mid processes and Max process i.e TQn=(Mid+Max)/2. So TQn=(57+85)/2=71. After first iteration the remaining CPU burst time sequence is P4=0,P3=0,P1=0,P2=0 and P5=14. In this case, processes P4,P3,P1 and P2 are deleted from the Ready Queue. Now there is only one process left P5 with burst time. So TQn=14. After second iteration P5=0. Now there is no process in the RQ, it completes its execution and TAT, AWT and CS are calculated. In this case, TAT=134.8, AWT=79.6, CS=4.

9. EXPERIMENTAL ANALYSIS

In every case we will compare the result of the proposed Average Mid Max Round Robin (AMMRR) method with Round Robin (RR) scheduling algorithm. Here we have taken 20 as the static time quantum (TQ) for RR algorithm

CASE 1:-Let‟s consider five processes with Burst time (P1=15,P2=35,P3=55,P4=85,P5=95) and Arrival Time =0 as shown in the Table 1. Table 2 shows the output using RR algorithm and AMMRR algorithm. Figure 3 and Figure 4 shows Gantt chart of both RR and AMMRR algorithm respectively.

Table 1. Processes with Burst Time

Processes	Arrival Time	Burst Time
P1	0	15
P2	0	35
P3	0	55
P4	0	85
P5	0	95

Table 2: Comparison between RR algorithm and our new proposed AMMRR algorithm (CASE 1).

Algorithm	Time Quantum	Turnaround Time	Average Waiting Time	Context Switch
RR	20	173	116	15
AMMRR	75,20	144	87	6

TQ=20

P1	P2	P3	P4	P5	P2	P3	P4	P5	P3	P4	P5	P4	P5	P4	P5

Fig.3: Gantt chart of RR from Table 1 of CASE 1.

TQ=75 TQ=20

P1	P2	P3	P4	P5	P4	P5

Fig.4: Gantt chart of AMMRR from Table 1 of CASE 1.

CASE 2:-Let‟s consider five processes with Burst time (P1=31, P2=32, P3=33,P4=34,P5=35) and Arrival Time =0 as shown in the Table 3. Table 4 shows the output using RR algorithm and AMMRR algorithm. Figure 5 and Figure 6 shows Gantt chart of both RR and AMMRR algorithm respectively

Table 3. Processes with Burst Time

Processes	Arrival Time	Burst Time
P1	0	51
P2	0	52
P3	0	53
P4	0	54
P5	0	55

Table 4: Comparison between RR algorithm and our new proposed AMMRR algorithm (CASE 2).

Algorithm	Time Quantum	Turnaround Time	Average Waiting Time	Context Switch
RR	20	237	180	14
AMMRR	54,1	104	157	4

TQ=20

P1	P2	P3	P4	P5	P1	P2	P3	P4	P5	P1	P2	P3	P4	P5

Fig.5: Gantt chart of RR from Table 3 of CASE 2.

TQ=54 TQ=1

P1	P2	P3	P4	P5	P5

Fig.6: Gantt chart of AMMRR from Table 3 of CASE 2.

2581-4575

International Journal For Recent Developments in Science & Technology

Crossref

A Peer Reviewed Research Journal

CASE 3:-Let‟s consider five processes with Burst time (P1=10,P2=20,P3=40,P4=80,P5=160) and Arrival Time =0 as shown in the Table 5. Table 6 shows the output using RR algorithm and AMMRR algorithm. Figure 7 and Figure 8 shows Gantt chart of both RR and AMMRR algorithm respectively

Table 5. Processes with Burst Time

Processes	Arrival Time	Burst Time
P1	0	10
P2	0	20
P3	0	40
P4	0	80
P5	0	160

Table 6: Comparison between RR algorithm and our new proposed AMMRR algorithm (CASE 3).

Algorithm	Time Quantum	Turnaround Time	Average Waiting Time	Context Switch
RR	20	134	72	11
AMMRR	100,60	114	52	4

TQ=20

P1	P2	P3	P4	P5	P3	P4	P5	P4	P5	P4	P5	P5	P5	P5	P5

Fig.7: Gantt chart of RR from Table 5 of CASE 3.

TQ=100 TQ=60

P1	P2	P3	P4	P5	P5

Fig.8: Gantt chart of AMMRR from Table 5 of CASE 3.

CASE 4:-Let‟s consider five processes with Burst time (P1=20,P2=30,P3=35,P4=50,P5=70) and Arrival Time as (P1=0,P2=5,P3=15,P4=25,P5=30) shown in the Table 7. Table 8 shows the output using RR algorithm and AMMRR algorithm. Figure 9 and Figure 10 shows Gantt chart of both RR and AMMRR algorithm respectively.

Table 7. Processes with Burst Time

Processes	Arrival Time	Burst Time
P1	0	20
P2	5	30
P3	15	35
P4	25	50
P5	30	70

Table 8: Comparison between RR algorithm and our new proposed AMMRR algorithm (CASE 4).

Algorithm	Time Quantum	Turnaround Time	Average Waiting Time	Context Switch
RR	20	112	71	10
AMMRR	53,17	84	43	4

P1	P2	P3	P4	P5	P2	P3	P4	P5	P4	P5	P5

Fig.9: Gantt chart of RR from Table 7 of CASE 4.

P1	P2	P3	P4	P5	P5

Fig.10: Gantt chart of AMMRR from Table 7 of CASE 4.

CASE 5:-Let‟s consider five processes with Burst time (P1=10,P2=40,P3=55,P4=80,P5=90) and Arrival Time as (P1=0,P2=5,P3=25,P4=30,P5=40) shown in the Table 9. Table 10 shows the output using RR algorithm and AMMRR algorithm. Figure 11 and Figure 12 shows Gantt chart of both RR and AMMRR algorithm respectively.

Table 9. Processes with Burst Time

Processes	Arrival Time	Burst Time
P1	0	10
P2	5	40
P3	25	55
P4	30	80
P5	40	90

Table 10: Comparison between RR algorithm and our new proposed AMMRR algorithm (CASE 5).

Algorithm	Time Quantum	Turnaround Time	Average Waiting Time	Context Switch
RR	20	145	88	13
AMMRR	73,17	119.6	48.6	6

TQ=20

P1	P2	P3	P4	P5	P2	P3	P4	P5	P3	P4	P5	P4	P5	P5

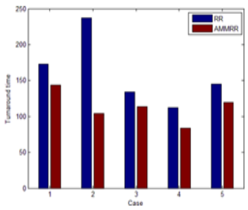
Fig.11: Gantt chart of RR from Table 9 of CASE 4.

TQ=73 TQ=17

P1	P2	P3	P4	P5	P4	P5

Fig.12: Gantt chart of AMMRR from Table 9 of CASE

International Journal For Recent Developments in Science & Technology
2581-4575
Crossref
A Peer Reviewed Research Journal



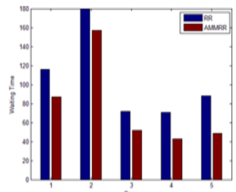


Fig.13: Comparison of average turnaround time of RR and AMMRR taking arrival time into consideration. *Fig.14. Comparison of average waiting time of RR and AMMRR taking arrival time into consideration.*

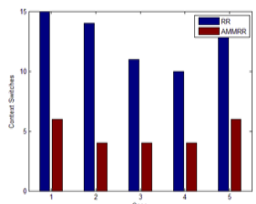


Fig.15. Comparison of Context Switches of RR and AMMRR taking arrival time into consideration.

10. CONCLUSION

From the above study we conclude that the proposed Average Mid Max Round Robin Scheduling algorithm shows better performance as compared to Round Robin Scheduling algorithm by decreasing the Total Turnaround Time, Average Waiting Time and Number of context switching .This is achieved by increasing the Time Quantum Dynamically.

REFERENCES

1. “Tarek Helmy, Abdelkader Dekdouk” Burst Round Robin: As a Proportional-Share Scheduling Algorithm, IEEE Proceedings of the fourth IEEE-GCC Conference on towards Techno-Industrial Innovations, pp. 424-428, 11-14 November,2007
2. “Yaashuwanth .C & R. Ramesh” Inteligent time slice for round robin in real time operating system, IJRRAS 2 (2), February 2010.
3. Pallab banerjee, probal banerjee, shweta sonali dhal,”Comparative Performance Analysis of Average Max Round Robin Scheduling Algorithm (AMRR) using Dynamic Time Quantum with Round Robin Scheduling Algorithm using static Time Quantum”,IJITEE,ISSN: 2278-3075, Volume-1, Issue-3, August 2012.
4. J. Nieh, C. Vaill and H. Zhong, “Virtual-Time Round-Robin: An O(1) Proportional Share Scheduler”, Proceedings of the USENIX
5. Sarojhiranwal and D.r. K.C.Roy“Adaptive Round Robin Scheduling using Shortest Burst Approach Based on Smart Time Slice”.volume 2,issue 3.
6. Sanjay Kumar Panda and Saurav Kumar Bhoi, “An Effective Round Robin Algorithm using Min-Max Dispersion Measure” ISSN : 0975-3397 ,Vol. 4 No. 01, January 2012.
7. “Tanebaun, A.S., 2008” Modern Operating Systems. 3rd Edn., Prentice Hall, ISBN: 13:9780136006633, pp: 1104.
8. “Silberschatz, A., P.B. Galvin and G. Gagne, 2008” Operating Systems Concepts. 7th Edn., John Wiley and Sons, USA., ISBN: 13: 978-0471694663, pp: 944.
9. H. S. Behera, Rakesh Mohanty, Sabyasachi Sahu and Sourav Kumar Bhoi.” Comparative performance analysis of multi-dynamic time quantum round robin (mdtqrr) algorithm with arrival time”, ISSN : 0976-5166, Vol. 2, No. 2, Apr-May 2011.
10. R. J. Matarneh, “Seif-Adjustment Time Quantum in Round Robin Algorithm Depending on Burst Time of the Now Running Proceses”, American Journal of

Applied Sciences 6 (10), pp. 1831-1837, 2009.

11. H. S. Behera, R. Mohanty, and D. Nayak, "A New Proposed Dynamic Quantum with Re-Adjusted Round Robin Scheduling Algorithm and Its Performance Analysis," vol. 5, no. 5, pp. 10-15, August 2010.
    A. Bhunia, "Enhancing the Performance of Feedback Scheduling", IJCA, vol. 18, no. 4, pp. 11-16, March 2011.
12. "Prof. Rakesh Mohanty, Prof. H. S. Behera, Khusbu Patwari, Manas Ranjan Das, Monisha Dash,Sudhashree" Design and Performance Evaluation of a New Proposed Shortest Remaining Burst RoundRobin (SRBRR) Scheduling Algorithm, Am. J. Applied Sci., 6 (10): 1831-1837, 2009.