Journal Homepage: -www.journalijar.com
INTERNATIONAL JOURNAL OF ADVANCED RESEARCH (IJAR)
Article DOI:10.21474/IJAR01/8506
DOI URL: http://dx.doi.org/10.21474/IJAR01/8506

RESEARCH ARTICLE

A MODIFIED ROUND ROBIN CPU SCHEDULING ALGORITHM WITH DYNAMIC TIME QUANTUM.

Md. Sohrawordi[1], U. A. Md. Ehasn Ali[1], Md. Palash Uddin[1] and Md. Mahabub Hossain[2].

1. Department of Computer Science and Engineering, Hajee Mohammad Danesh Science and Technology University, Dinajpur-5200, Bangladesh.
2. Department of Electronics and Communication Engineering, Hajee Mohammad Danesh Science and Technology University, Dinajpur-5200, Bangladesh.

Manuscript Info

Manuscript History
Received: 05 December 2018
Final Accepted: 07 January 2019
Published: February 2019

Keywords: -
CPU scheduling, Multitasking systems, Round Robin, Time quantum, Context switching, Turnaround time, Waiting Time.

Abstract

CPU scheduling is one of the basic factors for performance measure of multitasking operating system which makes a commuter system more productive by switching the CPU among the processes**.** The performance of the CPU scheduling algorithms depends on minimizing waiting time, response time, turnaround time and context switching, and maximizing CPU utilization**.** Round Robin **(**RR**)** is the most widely used CPU 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 becomes too high**.** On the other hand, the shorter time quantum raises the amount of context switch among 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 burst time of the set of waiting processes in the ready queue**.** The experimental results show that the proposed algorithm solves the fixed time quantum problem and decreases the average waiting time and turnaround time compared to traditional RR algorithm**.**

Introduction:-

An operating system is a program that manages the computer hardware and provides a basis for application programs and acts as an intermediary between the computer user and the computer hardware (Silberschatz et al. (2005))**.** The CPU executes multiple jobs by switching among them in multi-programmed system (Silberschatz et al. (2005))**.** The objective is to increase the CPU utilization by allowing new jobs to take over the CPU whenever the presently running job needed to wait (Banerjee et al. (2017**)).** In multiprogramming operating system, process scheduling is an important way that increases system performance by selecting a waiting process from ready queue and assigns the CPU to this selected process**.**

Corresponding Author: -Md. Sohrawordi.
Address**: -**Department of Computer Science and Engineering, Hajee Mohammad Danesh Science and Technology University, Dinajpur-5200, Bangladesh.

There are different kinds of algorithms implemented in CPU scheduling of operating system. Among them, Round Robin (RR) algorithm is most widely used in multiprogramming operating system (Noon et al. (2011), Yang et al. (2003), Padhy&Nayak (2014)).Various approaches are introduced for enhancing the performance of RR scheduling.Goyal et al. (2011) have arranged the processes in the ready queue in ascending order of burst time and taken the median of the burst time of the processes as time quantum for each cycle. Other reports proposed a fixed time quantum is taken for the waiting process in the ready queue only for first cycle, and then SJF is used to select next process (Yadav et al. (2010)).After arranging the processes in ascending order of burst time, the CPU burst of the mid process in case of odd number of processes is selected as time quantum; otherwise the average CPU burst of all running processes is taken time quantum in (Hiranwal et al. (2011)). Researchers, moreover, have taken the average of minimum and maximum burst time of the processes in the ready queue as time slice (Behera et al. (2011), Nayak et al. (2012)). More specifically, Behera et al. (2011) proposed an algorithm that calculates the original time slice which is suitable to the burst time of each processes and then dynamic ITS (Intelligent Time Slice) is found out in conjunction with the SRTN algorithm (Silberschatz et al. (2005)) while Nayak et al. (2012) calculated the optimal time quantum from the average of highest CPU burst time and the median and assigned it to each process after every cycle of execution. Meanwhile, Noon et al. (2011)have proposed an algorithm known as AN, based on anew approach called dynamic-time-quantum, that take the mean average of burst time of waiting processes as time quantum and updated this time quantum after one process completes its execution through one time quantum.The CPU is assigned to the first process of the ready queue for a time interval of up to one time quantum in an improved Round Robin (IRR) CPU scheduling algorithm (Mishra et al (2012)). After completing its one-time quantum, if the remaining burst time of the currently running process is less than one-time quantum, the CPU is again allocated to the same process. Otherwise, CPU is assigned to another waiting process. Later, IRRVQ (IRR with Varying time Quantum) algorithm is proposed by Mishra & Rashid (2014) for CPU scheduling. This method arranges the processes in the ready queue in the ascending order of their remaining burst time and set the time quantum value equal to the burst time of first process in the ready queue.

The CPU scheduler chooses a process to allocate the CPU for a fixed length of time quantum for all processes of the ready queue in RR (Silberschatz et al. (2005), Noon et al. (2011), Yang et al. (2003), Roy et al. (2017), Nitu et al. (2015)). This process continues repeatedly until ready queue becomes empty. Here, the quantum is a very small-time value which is set for preemption of the processes (Roy et al. (2017)). By this algorithm, the processes can execute concurrently (Silberschatz et al. (2005), Roy et al. (2017)). If the time quantum is too large, the response time of the processes becomes too high. On the other hand, if this time quantum is too small, it decreases response time but increases the amount of context switch among the processes (Noon et al. (2011)). Hence, RR with fixed time quantum may decrease the performance of a multitasking system in several times. This paper introduces a modified algorithm named Round Robin with Dynamic Time Quantum (RRDTQ) that uses dynamic time quantum in CPU scheduling along with RR. This time quantum is equivalent to the integer average value of the burst time of the remaining waiting processes in the ready queue. After each cycle, the time quantum is updated by calculatingthe integer average value of the burst time of the processes in the ready queue. The experimental results show that the proposed algorithm minimizesthe average waiting time, average turnaround time and amount of context switch among the processes with comparison that of the traditional RR, and recently proposed AN (Noon et al. (2011)) and IRRVQ (Mishra & Rashid (2014)) algorithms. Hence, the commuter system will be more productive by implementing the proposed algorithm.

Proposed RRDTQ CPU Scheduling Algorithm

The proposed RRDTQ algorithm uses dynamic time quantum in CPU scheduling. It takes the integer average value of the burst time of the remaining waiting processes in the ready queue as time quantum. After each cycle, the time quantum is updated by calculating the integer average value of the burst time of the processes in the ready queue. If the ready queue contains only one waiting process, then the time quantum will be the burst time of that processes. When the execution of the processes is completed, then it will be removed from ready queue and otherwise it will be added at the tail of the ready queue with the remaining required burst time. Following is the proposed RRDTQ CPU scheduling algorithm.

1. Create a ready queue RQUEUE of the processes that are ready for execution.
2. Repeat following steps 3to7 while RQUEUE is not empty.
3. Set the time quantum value equal to the integer of average burst time of the processes in the RQUEUE.
4. Repeat steps 5to7 until all processes in the ready queue get the CPU time interval up to 1 time quantum.
5. Pick the process that requires smallest CPU burst time among processes in RQUEUE and allocate CPU to this process for a time interval of up to 1 time quantum.

6. If the currently running process has finished its execution and the remaining CPU burst time of the currently running process is zero, then remove it from ready RQUEUE.
7. Else, put it at the tail of the ready queue with the remaining required burst time.

Experimental Result Analysis: -

Assumptions

For experimental analysis, all processes are treated having equal priority ina single processor environment. The time for context switching from one process to another is zero and all processes are CPU bound. There are noprocesses that are I/O bound. The number of processes and their burst time are known before submitting the processes for the execution. The time quantum for the proposed algorithm is taken in milliseconds.

Performance Evaluation

Two different cases have been taken for performance evaluation of the proposed RRDTQalgorithm. In the case 1, all processes arrival time is assumed same and in the case 2, processes arrival time isassumed different.

CASE 1 – Same Arrival Time

In this case, all processes are arrived at same time in ready queue. Now, consider a ready queue with five processes P1, P2, P3, P4, and P5 as shown in Table 1, assumed to have arrived at time 0 millisecond (ms) with the length of the CPU burst given in milliseconds.

Table1:-Processes with their arrival and burst time (CASE 1)

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

Figure 1, Figure 2 and Figure 3 show Gantt charts for simple round robin algorithm with time quantum of 10, 5 and 6 milliseconds respectively**.** Figure 4 and Figure 5 show the Gantt chart representation of AN and IRRVQ algorithms**.** Figure 6 shows the Gantt chart representation of the proposed RRDTQ algorithm with dynamic time quantum and the comparison result of RR, AN, IRRVQ and the proposed RRDTQ algorithm is shown in Table 2**.**

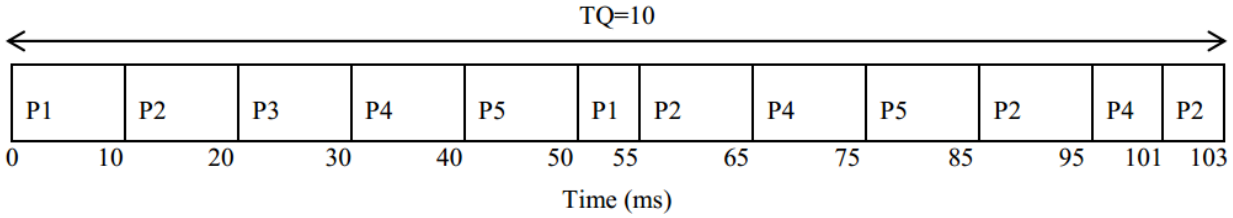

Figure 1: - Gantt chart representation of RR with TQ = 10

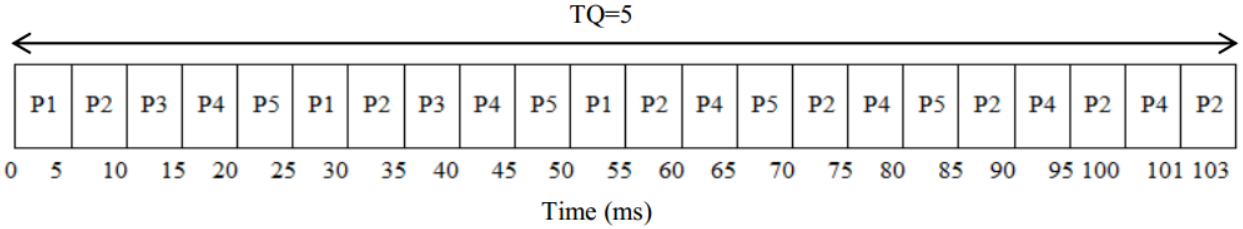

Figure 2: - Gantt chart representation of RR with TQ =5

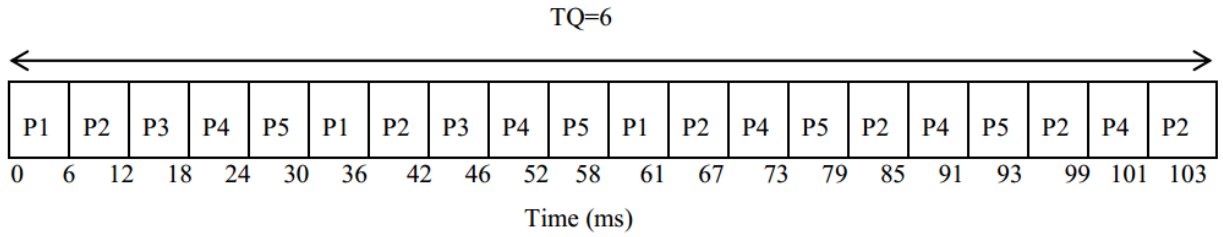

Figure 3: - Gantt chart representation of RR with TQ = 6

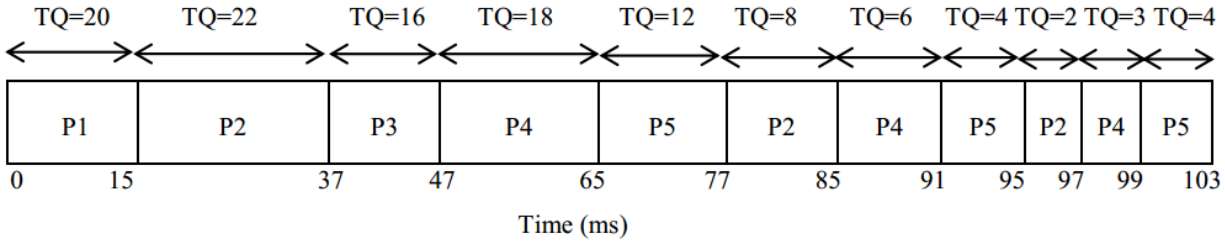

Figure 4: - Gantt chart representation of AN

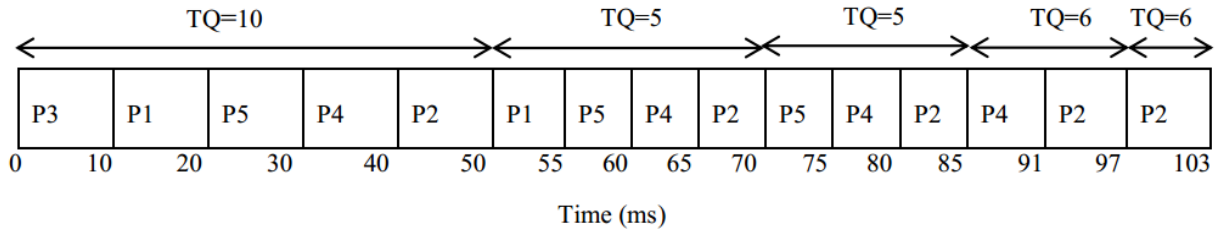

Figure 5: - Gantt chart representation of IRRVQ

From Table 1, processes P1, P2, P3, P4, and P5 have the length of the CPU burst time 15, 32, 10, 26 and 20 respectively. Then, the time quantum of the Proposed Round Robin will be the integer average value (103/5=20) of the CPU burst time. At first, the CPU is assigned to the process P3 for 10 milliseconds (ms), because it has minimum CPU burst time among the processes in the ready queue. After P3, the process P1 is executed through 15 ms, each P5, P4 and P4 processes run through 20 ms respectively in first time quantum. Now, the ready queue contains P2 and P4 processes with the CPU burst 12, 6 respectively. So, the time quantum will be upgraded to the integer average value (18/2=9) of the CPU burst of process P2 and P4. Finally, P2 runs through 6 ms, because it requires shorter CPU burst time then P4 and then P4 executes.

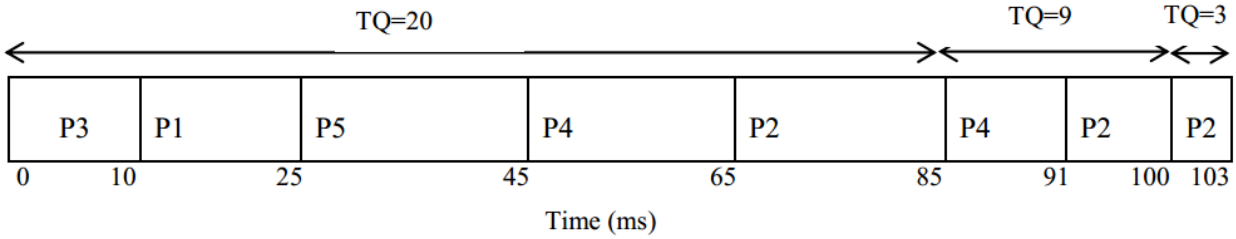

Figure 6: - Gantt chart representation of Proposed RRDTQ algorithm

Now, form the Gantt chart representation shown in Figure 6, it is seen that the waiting time using the proposed RRDTQ is 10 milliseconds for process P1, 71 milliseconds for process P2, 0 milliseconds for process P3, 71 milliseconds for process P4 and 25 milliseconds for process P5. Thus, the average waiting time is (10+71+0+71+25)/5=35.4 milliseconds. It is also seen that the turnaround time for the processes P1, P2, P3, P4, and P5 are 25, 103, 10, 91and 45 milliseconds respectively. Thus, the average turnaround time is (25+103+10+91+45)/5=54.8 milliseconds.

Table 2: - Comparison of RR, AN, IRRVQ and Proposed RRDTQ Algorithm.

Algorithm	Time Quantum (TQ)	Average Waiting Time (ms)	Average Turnaround Time (ms)	No. of Context Switch
RR	10, 5, 6	54.2, 56.2, 60.2	74.8, 76.8, 80.8	11, 21, 19
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
Proposed RRDTQ	20, 9, 3	35.4	54.8	6

Table 2 shows the comparative analysis of RR, AN, IRRVQ and Proposed RRDTQ Algorithm that are introduced in this paper. It is seen that the average waiting times are 54.2, 56.2 and 60.2 milliseconds for Round Robin with 10, 5 and 6 milliseconds time quantum respectively. The average waiting times are 51.6 and 46.2 milliseconds for AN and IRRVQ respectively. On the other hand, our proposed RRDTQ algorithm requires only the average waiting times of35.4 milliseconds for processes shown in table 1.

Besides, the average turnaround times are 74.8, 76.8 and 80.8 milliseconds for Round Robin with 10, 5 and 6 milliseconds time quantum respectively for the processes shown in table-1. It is also seen that the average turnaround times are 72.2 and 66.8 milliseconds for AN and IRRVQ respectively. On the other hand, our proposed RRDTQ algorithm requires only the average turnaround times of 54.8 milliseconds.

Table 2 also shows thatRound Robin algorithm with 10, 5 and 6 milliseconds time quantum acquire 11, 21 and 19 times context switching for executing the processes. AN, IRRVQ andRRDTQrequires 10, 14 and 6 times context switching for executing the processes. Hence the proposed algorithm decries the average waiting time, average turnaround time and amount of context switch among the processes than the traditional RR, AN and IRRVQ algorithm and enhances the CPU performance.

CASE 2 – Different Arrival Time

In this case, all processes are arrived at different time in ready queue. Now, consider a ready queue with five processes P1, P2, P3, P4, and P5 as shown in table 3, assumed to have arrived at various time with the length of the CPU burst given in milliseconds.

Table 3: - Processes with their arrival and burst time (CASE 2).

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

Figure 7 and Figure 8 show the Gantt charts for simple round robin algorithm with time quantum of 10 and 15 milliseconds respectively. Figure 9 and Figure 10 show the Gantt charts representation of AN and IRRVQ algorithms. Finally, Figure 11 shows the Gantt chart representation of the proposed RRDTQ algorithm with dynamic time quantum and the comparison result of RR, AN, IRRVQ and the proposed RRDTQ algorithm is shown in Table 4.

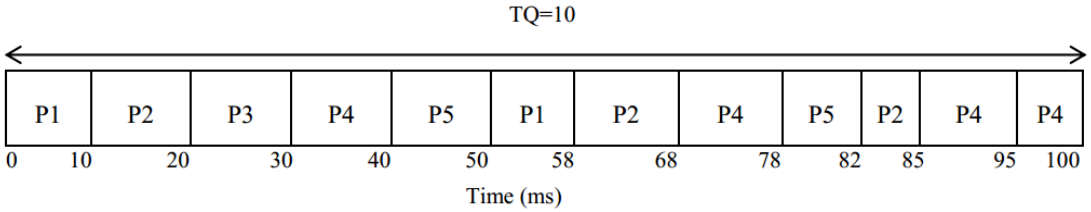

Figure 7: - Gantt chart representation of RR with TQ = 10

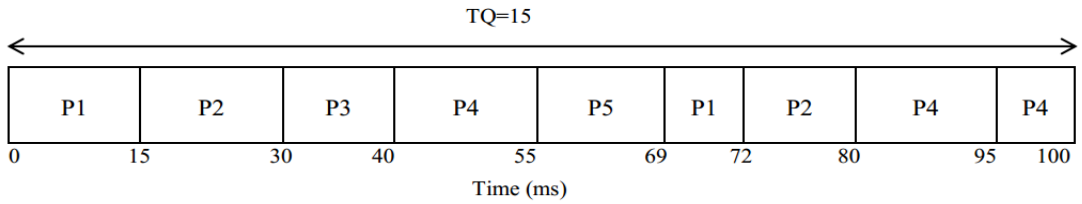

Figure 8: - Gantt chart representation of RR with TQ = 15

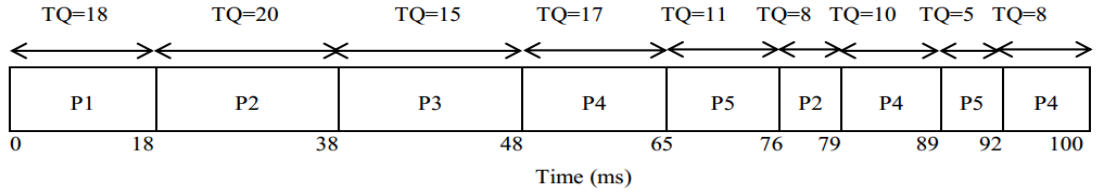

Figure 9: -Gantt chart representation of AN

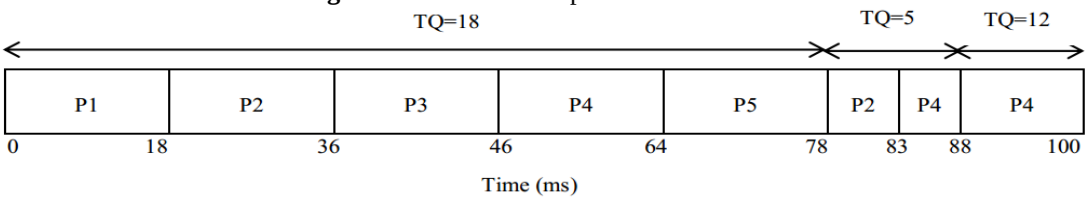

Figure 10: - Gantt chart representation of IRRVQ

From Table3, processes P1, P2, P3, P4, and P5 have the length of the CPU burst 18, 23, 10, 35, 14 respectively and arrived at 0, 3, 4, 8 and 10 respectively in ready queue. At 0 millisecond, the ready queue contains only P1 process. Thus, the time quantum for the proposed RRDTQ algorithm will be the burst time of P1 process. Hence, the CPU is assigned to the process P1 for 18 milliseconds. Now, the ready queue contains P2, P3, P4, and P5 processes. The time quantum of the proposed RRDTQ algorithm will be the integer average value (82/4=20) of the CPU burst time. At first, the CPU is assigned to the process P3 for 10 milliseconds (ms), because it has minimum CPU burst time among the processes in the ready queue. After P3 the process P5 is executed through 14 ms and then P1 is executed through 18 ms. Finally, each P2 and P4 processes run through 20 ms in first time quantum. Now, the ready queue contains P2 and P4 processes with the CPU burst 3, 15 respectively. In this manner, the time quantum will be upgraded to the integer average value (18/2=9) of the CPU burst of processes P2 and P4. Finally, P2 runs through 3 ms, because it has shorter CPU burst time then P4 and then P4 executes.

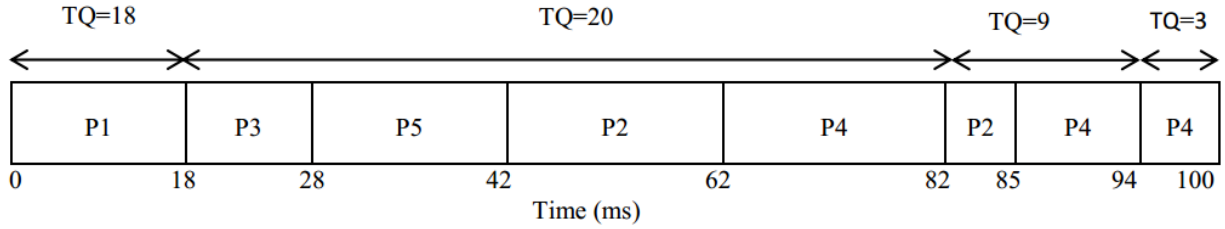

Figure 11: -Gantt chart representation of Proposed RRDTQ algorithm

Table 4: -Comparison of RR, AN, IRRVQ and Proposed RRDTQ Algorithm

Algorithm	Time Quantum (TQ)	Average Waiting Time (ms)	Average Turnaround Time (ms)	No. of Context Switch
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
Proposed RRDTQ	18, 20, 9, 3	29.6	49.2	6

Now, from the Gantt chart representation as shown in Figure 11, it is seen that the waiting time using the proposed RRDTQ are 0, 59, 14, 57 and 18 milliseconds for process P1, P2, P3, P4, and P5 respectively. It is also seen that the turnaround time for the processes P1, P2, P3, P4, and P5 are 18, 82, 24, 92 and 32 milliseconds respectively. Thus, the average turnaround time is (18+82+24+92+32)/5=49.2 milliseconds.

In this way, the experimental results shown in Table 4 prove that the proposed algorithm requires less average waiting time, less average turnaround time and less amount of context switch among the processes than the traditional RR, AN and IRRVQ algorithms**.** Hence, the CPU performance can be enhanced by implanting our proposed algorithm**.**

Conclusion:-

The allocation of CPU to the processes waiting for execution is the most important tasks of the operating system**.** Because the CPU is most valuable computer resource**.** There are many kinds of CPU scheduling algorithms that have been presented to manage the CPU with some advantages and disadvantages**.** This paper presents a new CPU scheduling algorithm with varying dynamic time quantum**.** The comparison results of round robin **(**RR**)**, AN, IRRVQ and the proposed algorithm RRDTQ have been investigated in this paper**.** This comparison proves that the proposed RRDTQ algorithm is more efficient than the conventional RR, AN and IRRVQ algorithms**.** Because it acquires less average waiting time, less average turnaround time and a smaller number of context switches than that of the RR, AN and IRRVQ**.** Therefore, the proposed algorithm is demonstrated to enhance the performance of multitasking operating systems**.**

References:-

1. Banerjee, P., Shree, R., and Verma, R. K. (2017): Generic Round Robin Scheduling for Real Time Systems, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 7, Issue 5, pp. 148-155.
2. Behera, H. S., Patel, S. and Panda, B. (2011): A new dynamic Round-robin and SRTN algorithm using variable original time slice and dynamic intelligent time slice for soft real time system, International Journal of Computer Applications, Vol. 2, No.1, pp. 54-60.
3. Goyal, L.K.D, Singh, R., and Sharma, P. (2011): Optimized Scheduling Algorithm, International Conference on Computer Communication and Networks CSI- COMNET-2011, Proceedings published by International Journal of Computer Applications, pp.106-109.
4. Hiranwal, S and Roy, k. C. (2011): Adaptive Round Robin Scheduling using Shortest Burst Approach Based on Smart Time Slice, International Journal of Computer Science and Communication, Vol. 2, No. 2, pp. 319-323.
5. Mishra, M. K. and Khan, A. K. (2012): An Improved Round Robin CPU Scheduling Algorithm, Journal of Global Research in Computer Science, Vol. 3, No. 6, pp. 64-69.
6. Mishra, M.K., and Rashid, F. (2014):AN IMPROVED ROUND ROBIN CPU SCHEDULING ALGORITHM WITH VARYING TIME QUANTUM, International Journal of Computer Science, Engineering and Applications (IJCSEA), Volume 4, No. 4, pp. 1-8, 2014.
7. Nayak, D, Malla, S, K. and Debadarshini, D. (2012): Improved Round Robin Scheduling using Dynamic Time Quantum, International Journal of Computer Applications, Vol. 38, No. 5, pp. 34-38.
8. Nitu, A. M., Uddin, M. P., Islam, M. M., and Islam, M. S. (2015): Developing a Simple, Interactive and Easy Simulation Tool for the Teachers/Undergrad Students with a view to Teaching/Learning CPU Scheduling Algorithms, Institutional Engineering and Technology (IET), Volume 5, No. 1, pp. 25-29.
9. Noon, A, Kalakech, A. and Kadry, S. (2011): A New Round Robin based Scheduling Algorithm for Operating Systems: Dynamic Quantum Using the Mean Average, International Journal of Computer Science Issues, Vol. 8, Issue 3, No. 1, pp. 224-229.
10. Padhy, C., and Nayak, D. R. (2014); Revamped Round Robin Scheduling Algorithm, Journal of Engineering Computers & Applied Sciences (JECAS), Volume 3, No.4, pp.51-59.
11. Roy, A. C., Mamun, M. A. A., Hossin, K, Islam, M. A, Uddin, M. P, Afjal, M. I. and Sohrawordi, M. (2017): Developing Operating System Simulation Software for Windows Based System by C# .NET Framework and an Android Application by JAVA and XML, Journal of Operating Systems Development & Trends, Volume 4, No. 1, pp. 9–18.

12. Silberschatz, A., Galvin, P. B., and Gagne, G. (2005): Operating System Concepts, 7th Edition, John Wiley and Sons Inc.
13. Yadav, R.K., Mishra, A.K., Prakash, N and Sharma, H. (2010): An Improved Round Robin Scheduling Algorithm for CPU Scheduling, International Journal on Computer Science and Engineering, Vol. 2, No. 4, pp. 1064-1066.
14. Yang, L., Schopf, J.M. and Foster, I. (2003): Conservative Scheduling: Using predictive variance to improve scheduling decisions in Dynamic Environments, Super Computing, Phoenix, AZ, USA.