Optimized Time Quantum for Dynamic Round Robin Algorithm

[1]Sumit Mohan, [2]Rajnesh Singh

[1]Student-M.Tech, [2]Professor, IEC College of Engineering & Technology, Greater Noida, India,

[1]*sumitmohan600@gmail.com*, [2]*rajneshsingh.cs@ieccollege.com*

Abstract - Round robin is the only choice in time-sharing but we cannot implement in a real-time system because it has large waiting time, more no. of context switches hence we need some improvement in existing round robin to get expected performance. In order to get it, we must focus on the choice of time quantum value because optimal time quantum can improve the performance. In this algorithm, processes get equal time to process its task in a cyclic way. Selecting the optimal time quantum is the main problem because of everything depends on it. If we select time quantum small then CPU will be spending a large amount of time in process switching and if it is large then response time increased which can't be tolerated in time sharing. This paper shows an optimal way of selecting the time quantum which will improve the performance of round robin, the performance of Optimized Time Quantum Round Robin better than existing Round Robin algorithms.

Keywords — Round Robin, Context Switching, Burst Time, Turnaround Time, Waiting Time, Time Quantum.

I. INTRODUCTION

Improper use of CPU can affect the efficiency of a multi-programmed system so we need to schedule task properly in which we can utilize system resources efficiently. Hence Scheduling algorithms are used for proper utilization of system resources, we have many algorithms and each algorithm has a specific environment. We can select the algorithm as per our requirements. In contrast of time sharing environment, we have round robin scheduling algorithm here every process gets equal opportunity to execute its task [3]. Round robin has many problems with its static version

Every process has following parameter CPU time, Arrival time and priority. CPU time is the of the process which needs CPU resources or I/O operations, the priority of a process tells CPU to its importance and Arrival time of any process when a process submit a request to the ready queue, Turnaround Time is the difference of submission of a request to completion.

The Waiting Time of a process is the amount of time in which process spends for its execution. Context Switching when the processor switches from current process to other processes, Effectiveness of CPU scheduling algorithm always depends on waiting, turnaround time and context switches [14].

Basic Scheduling Algorithms:

First Come First Serve (FCFS)

In FCFS, Processes scheduled according to the arrival time of the processes. FCFS is simple and easy to implement and its nature is non-preemptive. The main problem with this algorithm is avg. waiting time. In this algorithm, short burst time process may wait for a long time. If long burst time process gets first then other processes must wait for execution this situation can slow down the speed of CPU and also leads to convey [14]. Every process gets chance to run, so no starvation.

Shortest Job First (SJF):

In this algorithm, a process scheduled first which has minimum CPU time. If processes having equal arrival time and burst time, then it follows FCFS algorithm, CPU time of processes must be known by the processor which is not possible. The throughput of this algorithm quite good as compare to other but starvation is also there if process having longer burst time. It can be used for the interactive environment when past patterns are available [11].

Shortest Remaining Time First (SRTF)

The improved version of SJF is SRTF, in this algorithm job scheduler pick the process which has minimum burst time and gives CPU, current process can execute its task until next request or processes are not available in the ready queue, next process will be scheduled which has minimum CPU time at that time. If we have many short burst time processes then longer burst time processes will wait for a long time which leads to starvation situation. The throughput of this algorithm is good because a short process can execute their task with minimum waiting time [13].

Priority Scheduling Algorithm:

It provides the priority to each process and highest priority process will get priority over other processes. In this algorithm waiting time of low priority processes is much more as compared to higher priority processes; it can suffer the starvation problem. We can solve this problem by increasing the priority of processes which are waiting for long a time. In this algorithm, we can assign the priority to processes as per requirement [3] [10].

 DOI : 10.18231/2454-9150.2018.0343

Round-robin Scheduling Algorithm:

It is mostly used as the algorithm in time sharing. In this algorithm, every process gets equal time to run and preempted after quantum time value in a cyclic way. Scheduler picks process and set counter to interrupt after one -time quantum and give processor to other processes. if the process has burst time greater than the time quantum value, then counter will go off after one-time quantum values exceeded and it interrupts the current process and gives CPU to other process and the interrupted process will get CPU again in which process interrupted first [14][15].

CPU scheduling algorithm performance criteria are following:

Processor Utilization: Sum of work done by processor We can estimate the performance of a system by processor utilization. Always we want processor utilization as much as possible.

Throughput: Measure the work of a processor done in one unit of time.

Throughput = (Completed Processes) / (One Unit Time)

Turnaround Time: The time which is spent by a process in the waiting queue for the ready queue.

TAT = Completion Time – Request Submission Time

Waiting Time (WT): When process wait for its turn in the ready queue is called Waiting Time.

WT=Turnaround Time –Burst Time

Response Time (RT): when processor starts processing the request of the process

RT = First Response – Arrival Time

II. RELATED WORK

There are many problems with static round robin because in this algorithm we choose the time quantum randomly; sometimes it gives a good performance. But we cannot say that always [4] it will give a good performance because time quantum is the main key of the performance of round robin. We cannot choose time quantum very small and too large it must be optimal because in the small time quantum more no. of context switches occur and response time of the process will be increased if we choose time quantum too much large[3]. Recently many approaches proposed for selecting Time Quantum. Some of the approaches are very optimal and these approaches are adaptive Scheduling using Shortest Burst Approach Based on Smart Time Slice [1] in this algorithm all processes are arranged in increasing (burst time) order, shortest burst time process will get first priority to run its task and time quantum calculated by average of all process if no. of processes are even otherwise mid process burst time will be selected as a time quantum. Min-Max Round Robin [2] in this algorithm time calculated by maximum burst time of process – minimum burst time of the process. MDTRR [5] this algorithm gave another approach for time quantum here time quantum selected median of all processes and this time quantum assigned till the median after that time quantum will be assigned by the first process of the first quarter. Intelligent Time Slice for Round Robin [7] here time quantum depends on various factor OTS, Priority Component and context switches component. [6]SRBRR in this paper time quantum calculated by the median of the processes multiplied by highest burst time process then applied sqrt function and then finally take the ceil value of calculated s result.

III. PROPOSED ALGORITHM

The Optimized Time Quantum for Dynamic round Robin Algorithm mainly focuses on performance parameters of the scheduling algorithm. In case of RR, time quantum is the key parameter of performance because everything depends on the selection of it. This algorithm gave the best optimal way of selecting the time quantum; our result analysis shows that how it is better than other algorithms. Our algorithm firstly find the square of all available processes then divided by no. of processes then apply sqrt mathematical function then we have a most important variable(adjustment factor) which is **α**(alpha) and its value is fixed 0.9 multiply with obtained value then finally choose the floor value.

TQ = floor [{sqrt ($\sum p_i^2/n$)*α}]

The proposed (OTQRR) algorithm follows:

1. Initialize n= No. of Processes and α=0.9.
2. While (ready queue! = NULL)
   TQ = floor[{sqrt($\sum p_i^2/n$)*α}]
   For i=1 to n…// i loop variable

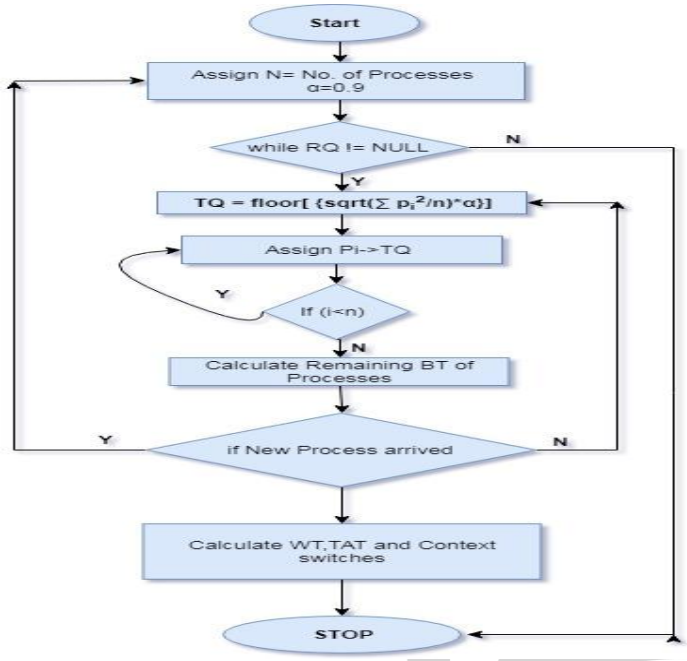
   Where α is the Adjustment Factor.
3. Assign TQ to I_{th}Process Pi→TQ
4. If (i<n) then go to step 3.
5. Calculate the remaining burst time of processes
6. If other processes (new) come then
   Update n and goto: 1
   End of while
7. Calculate TAT, WT and No.of CS.
8. End

Firstly we will check the status of ready queue if its status is not null then initialize the value of n equal to no. of processes available in ready queue and also set the value of α equal to 0.9 and then calculate the value of time quantum by using given formulae and then assign calculated time quantum value to Pi_{th} process check the condition if i_{th} value less than n loop will continue after the loop iteration calculate the remaining burst time of processes, during the processing if any new process is coming then update the value of n and goto first step and then follow same procedure. For loop will run i to n(no. of processes) if the counter value less than n then assign TQ to next process until the i_{th} value less n. after the for loop body calculate the remaining burst time all available processes. if any new coming then goto the first step of the algorithm and update the value of n and check the ready queue status if it is not

 DOI : 10.18231/2454-9150.2018.0343

NULL then follow the above procedure.

Calculate the Avg. Waiting Time, Turnaround Time and No. of Context Switches.



Flow Chart of Proposed Algorithm

IV. EXPERIMENTS & RESULTS

Assumptions

I assumed that Attributes of a process (CPU Time, Priority) must be known before submission of processes, and CPU bound. The Arrival Time of all processes must be same and this experiment performed in a single processor system, the processes are independent of each other.

Case: 1

Let's take 5 Processes in increasing order at arriving time=0
As shown below.

Table 1: Burst Time of Processes

Process	CPU Time
P_1	37
P_2	58
P_3	76
P_4	115
P_5	137

TQ calculated by the proposed formula
TQ = floor sqrt $(\sum [(P_i^2)/n])*\alpha$
TQ=floor $[\{sqrt\ (37^2+58^2+76^2+115^2+137^2)/5\}* \alpha]$
= (42503) ÷5=8500.6
= √8500.6 = 92.19
= 92.19 *0.9 = 82.97
= floor 82.97 = 82
TQ= 82

Gantt chart Case: 1

P_1	P_2	P_3	P_4	P_5	P_4	P_5

0 37 95 171 253 335 368 423

No. of Context Switches = 6
Avg. Waiting Time = (0+37+95+253+286) / 5 = 671/5=134.2
Avg. Turnaround Time = (37+95+171+368+423) / 5 =1094/5=218.5

Table 2: Comparative Analysis RR, OTQRR algorithm (case – 1)

Algorithm	TQ	Avg. TAT	Avg. W T	C S
RR	50	272.4	187.8	10
OTQRR	82	218.5	134.2	6

Case: 2

Let's Take Same Processes in Decreasing order and processes at t=0 Arriving Time
As shown below

Table 3: Burst Time of Processes

Process	CPU Time
P_5	137
P_4	115
P_3	76
P_2	58
P_1	37

TQ = floor $(\sum$ sqrt $[(p_i^2)/n])*\alpha$
TQ=floor $[\{sqrt\ (137^2+115^2+76^2+58^2+37^2)/5\}* \alpha]$
= (42503) ÷5=8500.6
= √8500.6 = 92.19
= 92.19 *0.9 = 82.97
= floor 82.97 = 82
TQ= 82

Gantt chart Case: 2

P5	P4	P3	P2	P1	P5	P4

0 82 164 240 298 335 390 423

Table 4: Comparative Analysis RR, OTQRR algorithm (case – 2)

Algorithm	TQ	Avg. TAT	Avg. W T	C S
RR	50	360.4	275.8	10
OTQRR	82	337.2	252.6	6

Case: 3

Let's Take Same Processes in Random order and processes at t=0 Arriving Time
As shown below

 DOI : 10.18231/2454-9150.2018.0343

Table 5: Burst Time of Processes

Process	CPU Time
P_2	58
P_1	37
P_4	115
P_3	76
P_5	137

TQ=floor [{sqrt (58^2+37^2+115^2+76^2+137^2)/5}* α]
= (42503) ÷5=8500.6
= √8500.6 = 92.19
= 92.19 *0.9 = 82.97
= floor 82.97 = 82
TQ= 82

Gantt chart Case: 3

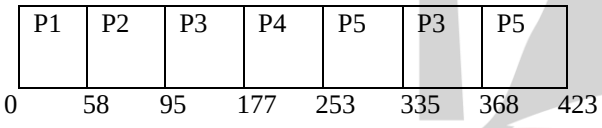


Table 6: Comparative Analysis RR, OTQRR algorithm (case – 3)

Algorithm	TQ	Avg. TAT	Avg. W T	C S
RR	50	292.4	207.8	10
OTQRR	82	239.4	154.8	6

V. CONCLUSION AND FUTURE WORK

Problem statement was the choice of optimal time quantum for dynamic round robin there this experiment on data shows that selected time quantum value has better performance than static round robin thus we can say that proposed approach is optimal for selecting the time quantum. My above-proposed algorithm comparison shows that Optimized Time Quantum Round Robin has better performance than Round Robin in case of waiting, turnaround time and no. of context switches. This proposed algorithm gave the best optimal approach for time quantum which leads to better performance in term of context switches and avg. waiting time and turnaround time, In future work, there may another way of selecting time quantum value, which will give better performance than mine and it can be proposed for multicore processor system and also for the distributed environment.

REFERENCES

[1] Saroj hiranwal and D.r. K.C.Roy"Adaptive Round Robin Scheduling using Shortest Burst Approach Based on Smart Time Slice".volume 2,issue 3.

[2] 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.

[3] "Tannenbaum, A.S., 2008" Modern Operating Systems. 3rd Edn., Prentice Hall, ISBN: 13:9780136006633, pp: 1104.

[4] "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.

[5] 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.

[6] "Rakesh Mohanty, H.S. Behera and et. al, Design and Performance Evaluation of a new proposed Shortest Remaining Burst Round Robin(SRBRR) scheduling algorithm, Proceedings of the International Symposium on Computer Engineering and Technology(ISCET), March 2010.

[7] "Yaashuwanth .C & R. Ramesh" Intelligent time slice for round robin in a real-time operating system, IJRRAS 2 (2), February 2010

[8] Milan Milinkovic, "Operating Systems Concepts and Design", McGraw-Computer Science Series, second edition.

[9] Adore "Operating Systems", Technical Publications.

[10] M. Dietel, "Operating Systems", Pearson Education, Second Edition.

[11] "Real Time Task Scheduling ", NPTEL, IIT Kharagpur

[12] P. Balakrishna Prasad, "Operating Systems" Second Edition.

[13] https://en.wikipedia.org/wiki/Operating_system

[14] https://en.wikipedia.org/wiki/Scheduling_(computing)

BIOGRAPHIES:

"Sumit Mohan" Graduated in 2016 from Galgotia's College of Engineering & Technology and now pursuing his Master degree in Computer Science from Dr. APJ Abdul Kalam Technical University Lucknow (U.P.). He has also done three- year Polytechnic diploma in Computer Science. He has written in several international journals and conference. He is now active in writing papers and joining conferences.

"Prof. Rajnesh Singh" Presently working as a **Professor** and **HOD (CS)** in IEC College of Engineering and Technology, Greater Noida since 2009. He has done M.Tech from CDAC Noida GGSIPU New Delhi in 2006. He has published many research papers in the field of Networking in IEEE and 10 years of teaching experience.

 DOI : 10.18231/2454-9150.2018.0343