Enhanced round robin CPU scheduling with burst time based time quantum

IOP Conf. Series: Materials Science and Engineering **263** (2017) 042038 doi:10.1088/1757-899X/263/4/042038

Enhanced round robin CPU scheduling with burst time based time quantum

J R Indusree and B Prabadevi

Vellore Institute of Technology, Vellore-632014 Tamilnadu, India.
Email: prabadevi.b@vit.ac.in

Abstract. Process scheduling is a very important functionality of Operating system. The main-known process-scheduling algorithms are First Come First Serve (FCFS) algorithm, Round Robin (RR) algorithm, Priority scheduling algorithm and Shortest Job First (SJF) algorithm. Compared to its peers, Round Robin (RR) algorithm has the advantage that it gives fair share of CPU to the processes which are already in the ready-queue. The effectiveness of the RR algorithm greatly depends on chosen time quantum value. Through this research paper, we are proposing an enhanced algorithm called Enhanced Round Robin with Burst-time based Time Quantum (ERRBTQ) process scheduling algorithm which calculates time quantum as per the burst-time of processes already in ready queue. The experimental results and analysis of ERRBTQ algorithm clearly indicates the improved performance when compared with conventional RR and its variants.

1. Introduction

Operating system (OS) is a collection or package of numerous soft-wares which act as an intermediary between the computer hardware and end user. It provides a means for the user to work with the computer .Operating system performs major functions like memory management, process scheduling and resource management.

Process scheduling: Process scheduling is a very important functionality of OS which is responsible for allocating CPU to a process. The new processes which are to be executed are placed in a queue called ready-queue from where the OS module called scheduler picks up a process for execution [1]. Scheduling algorithms are the strategies which help scheduler to choose a process from ready queue.

Scheduling Algorithms: Present day requirements are aligned towards multiprocessing systems and continual efforts are being carried out towards developing efficient and optimal CPU scheduling strategies. There are different algorithms [1] for CPU scheduling in practice, the most prominent ones being First Come First Serve (FCFS) algorithm, Shortest Job First (SJF) algorithm and Round Robin (RR) algorithm. Of these, Round Robin algorithm is advantageous than others since, every process gets to share the CPU for a fixed time quantum without having to wait for complete execution of other processes.

Round Robin (RR) algorithm: RR algorithm executes the processes in the order of arrival time for a fixed time quantum in a circular fashion. So, all processes in the ready queue takes turn to execute for this predefined time quantum. If the burst-time of a process is less than the value of the time quantum, the next process in the queue starts execution. Thus there is pre-emption in RR algorithm. This way, every process gets a fair portion of the CPU without having to wait long for its execution.

The advantages of RR algorithm are as follows:

- Starvation of processes can be avoided since every process takes turn.

- Processes with smaller burst time get advantage of execution earlier compared to other algorithms.
- RR algorithm is pre-emptive.

1.1 Motivation:
Conventional RR algorithm uses a random time quantum (most often of the range 20-50ms). The efficiency of scheduling algorithm is measured by the following performance [2] factors:

- *Throughput:* The number of processes executed in unit time
- *Waiting- time:* The amount of time a process has to wait for CPU in ready-queue.
- *Turn-around time:* The total amount of time required by a process to complete execution.
- *Response time:* The time taken from the generation of a request and the first response.
- *Context Switching:* The process of switching CPU between processes. This is otherwise known as preemption.
- *CPU Utilization:* A measure of maximum usage of CPU / how effectively CPU is made busy.

The performance and effectiveness of RR algorithm is largely dependent on the value of time quantum selected [3]. If the value of time quantum is too small then the number of context switches will be more and algorithm will not be effective. If the value of the time quantum is too large, then the algorithm will work more or less like FCFS algorithm. Choosing an optimum time quantum can significantly decrease the number of context switches, maintaining the RR nature and also can improve performance. The number of context switches can further be reduced if there is a strategy to execute processes with smallest remaining burst- times.

This paper is organized as follows: The section 2 explains details about the existing algorithms which are variants of RR algorithm. Section 3 briefs the proposed methodology, algorithmic procedure, pseudo code and flow chart. Section 4 illustrates with example, the working of the proposed algorithm. Section 5 details the comparison of the proposed algorithm with other existing RR variants. Section 6 and 7 contains conclusion and future enhancements respectively.

2. Related Work

Augmented Dynamic Round Robin scheduling (ADRR) [4]: In this algorithm, processes are executed according to their arrival times. Once a process is executed for the defined time quantum, instead of switching to the next process, checks whether the resultant burst-time of the current process is less than or equal to time-quantum value. If yes, the same process is executed. Otherwise, process execution happens as in normal RR. This strategy reduces number of context switches as compared to conventional RR.

Improved Mean Round Robin with Shortest Job First Scheduling (IMRRSJF) [5]: This algorithm has combined features of both RR and SJF, ie, the processes are arranged according to the burst-time values and time-quantum is calculated as the square root of the product of mean and highest burst time. Whenever a new process comes in, the processes are again sorted in ascending order and time quantum is calculated again.

Time Quantum Based Improved Scheduling Algorithm (TQBISA) [6]: This algorithm arranges the processes in ascending order of burst times and then calculates time quantum as the median of the process burst times. The processes are then executed as in case of normal RR.

Dynamic Quantum with Re-Adjusted Round Robin Scheduling Algorithm and Its Performance Analysis (DQRR) [7]: This algorithm first arranges processes in increasing order of burst times, calculates time quantum as the median of the burst-times. After each process execution, the processes are re-arranged such that process with least remaining burst-time will come first, then the process with highest remaining burst-time comes followed by the process with second least remaining burst-time and so on.

3. Proposed Methodology

Enhanced RR with Burst time based Time Quantum (ERRBTQ) works similar to RR with combined features of ADRR and TQBISA. The figures Fig.1., depicts the flowchart of the ERRBTQ.

3.1 Assumptions:

1. Set of processes comes into the ready queue and algorithm is applied when the ready-queue is fully occupied.
2. The burst-time values are assumed to be known.

3.2 Approach:

The processes in ready queue are arranged in increasing order based on their burst times. The time quantum is calculated as the median of burst times of these processes .

Median = $B\_T_{(m+1)/2}$; *if m is odd*

$(B\_T_{m/2} + B\_T_{(m+1)/2})/2$; *if m is even*

where m is the number of processes and B_T_m denotes the burst-time of m^{th} process. The first process from the ready-queue after arranging the processes according to burst-times (increasing order) is picked up by scheduler and it executes for the time- quantum. Once time- quantum completes, if the remaining burst-time value of the present process is less than or same as the time-quantum value, then the same process executes to completion and is moved out of the ready-queue. Then the next process is picked up and the procedure is repeated.

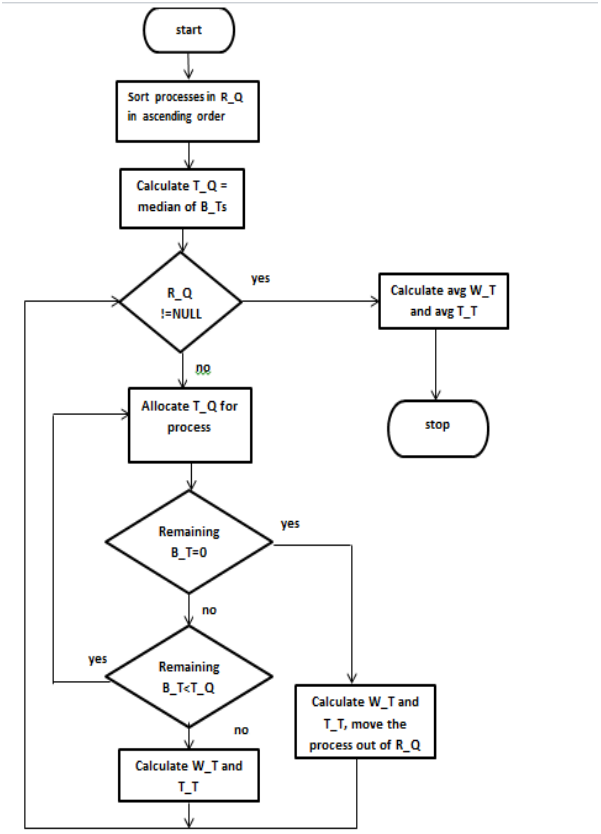


Fig. 1. Flowchart

3.3 Algorithm:

Notations:
R_Q: Ready-Queue
B_T: Burst-Time
T_Q: Time-Quantum
W_T: Waiting-Time
A_WT: Average Waiting-Time
T_T: Turn-around Time
A_TT: Average Turn-around Time

Step1: start
Step2: processes are sorted in increasing order of B_T
Step3: T_Q is calculated as median of the B_Ts.
Step4: do steps 5-8 until R_Q becomes empty.
Step5: For the process, allocate CPU for T_Q
Step6: If the remaining B_T is zero, calculate W_T and T_T and move the process out of the R_Q.
Step7: Else if remaining B_T of the current process is less than or equal to T_Q (not equal to 0), go to step5
Step8: Else, calculate the W_T and T_T of the process and move on to the next process in R_Q.
Step9: calculate average waiting-time and turn-around time.
Step10: stop

4. Illustrations

Consider 5 processes from p1 to p5 all arriving at same time say 0ms and having burst-times as shown in Table1.The processes are arranged in an increasing order of their burst -times as shown in Table2. Time quantum T_Q is the median of burst times (Table 2) which is 40. Now, first process in ready queue, P1 is executed for 26ms. Since it has completed execution, it is removed from ready queue and next process P4 is executed for 31ms and it is also deleted from ready queue. Next, process P5 executes and goes in similar way. Following this, process P3 executes for 40ms. The remaining burst time of P3 is 70-40=30ms; since 30 is less than T_Q 40ms, same process P3 executes to completion thereby reducing 1 context switch. Last, process P2 executes for 40ms. Remaining burst time of P2 is 82-40=42ms which is greater than T_Q ie, 40. But since there are no other process left in the ready-queue, P2 itself executes for 40ms. Now 2ms burst remains for P2 which is less than T_Q ie, 40ms. So P2 executes again to completion. If there was another process P6 then, after P2 executed for 40ms in the first context then process P6 would have got chosen.

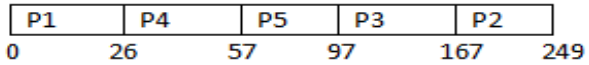
Table 1. Sample Processes with their burst-times

Process	**Burst-time(ms)**
P1	26
P2	82
P3	70
P4	31
P5	40

Table 2. Sorted Processes

Process	**Burst-time(ms)**
P1	26
P4	31
P5	40
P3	70
P2	82

The Gantt chart for ERRBTQ algorithm is as follows:

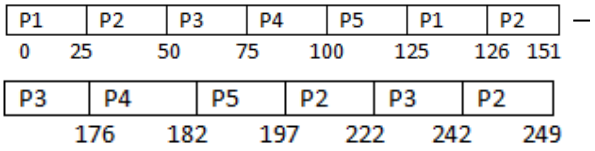


T_Q=40
Average turn-around time = (26+ 57+ 97+ 167+ 249) /5
= 119.2ms
Average waiting- time = 119.2- (26+82+70+31+40)
= 69.4ms
Number of context switch= 4

5. Experiments Aand Results

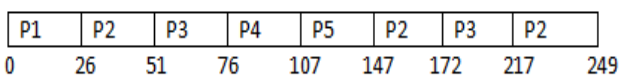
In this section we make a comparison of the performance measures of proposed algorithm with other RR algorithms.

Round Robin algorithm:
Let us take the time quantum value as 25ms. Conventional RR executes processes in the order of arrival time.
For the processes given in Table 1, the Gantt chart as per RR algorithm is as below:



Average waiting-time= 149.4ms
Average turn-around time=199.2ms

ADRR:
Here also we assume the time quantum to be 25ms. The Gantt chart of process execution is as follows:

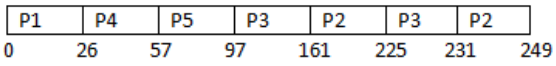


Average waiting-time= 99.4ms
Average turn-around time=149.2ms

IMRRSJF:

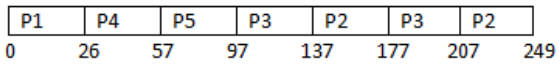
Here the processes are sorted in increasing order of burst- times and the time-quantum is:

T_Q = √(mean*highest burst-time)
= 64
Gantt chart is as below:



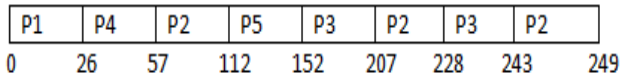
Average waiting-time= 82ms
Average turn-around time= 132ms

TQBISA:
Here also, the processes are sorted in increasing order of burst-time and time-quantum is the median:
T_Q= 40
Gantt chart is as follows:



Average waiting-time= 77.4ms
Average turn-around time= 127.2ms

DQRR:
In DQRR algorithm, the processes are alternately arranged in ascending and descending orders of burst-times respectively and time-quantum is the median of burst times. The time-quantum for p1 is 26ms; p4,p2,p5 and p3 takes 55ms; p2,p3 takes 21ms; p2 takes 6ms
Gantt chart is as follows:



Average waiting-time= 95.6ms
Average turn-around time= 145.4ms

5.1 Comparison of algorithms:
Based on the experimental analysis done in the previous section, we compare the algorithms with respect to the following performance measures:

1. Average waiting-time
2. Average Turn-around time
3. Number of context switching

Comparison based on performance measures:

Table 3. Performance measures

Algorithm / Perf measure	RR	ADRR	IMRRSJF	TQBISA	DQRR	ERR
Time Quantum	25	25	64	40	26,55,21,6	40
Number of context switches	12	7	6	6	7	4
Average waiting time	149.4	99.4	82	77.4	95.6	69.4
Average turnaround time	199.2	149.2	132	127.2	145.4	119.2

Graphical comparison of average waiting-time in various algorithms:

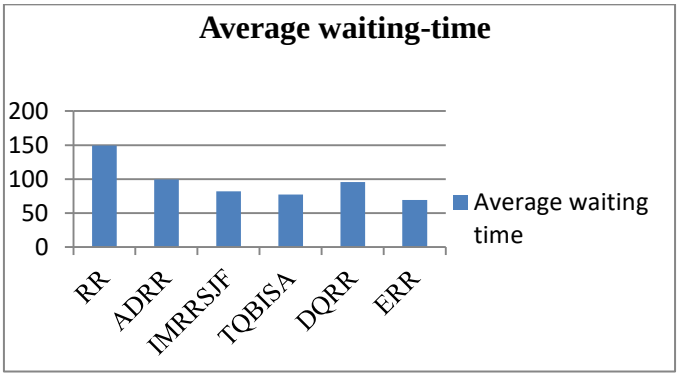


Fig. 2. A_WT comparison

Graphical comparison of average turn-around time in various algorithms:

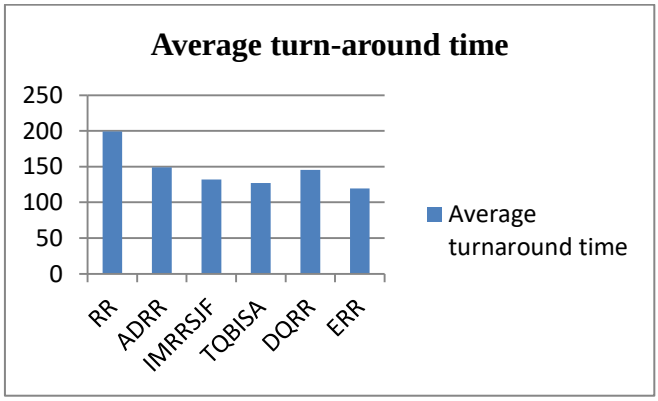


Fig. 3. A_TT comparison

Graphical comparison of context switching in various algorithms:

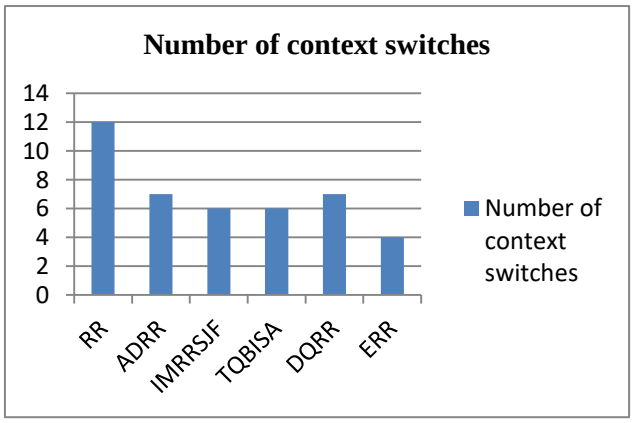


Fig. 4. Context switches comparison

6. Conclusion

The proposed algorithm provides improved performance as is evident from the experiment results of previous section. The number of context switches is very less compared to RR algorithm. It also reduces average waiting-time and turn-around time.

7. Future Work

ERRBTQ algorithm can be further enhanced by integrating appropriate strategy to control starvation [8]. It can also be modified to consider arrival-time of the processes.

References

[1] A. Silberschatz, P. B. Galvin, and G. Gagne, 2004 *Operating System Concepts*, 7th Edn., John Wiley and Sons, ISBN 0-471-694663,pp.944-947

[2] Ishwari Singh Rajput and Deepa Gupta, 2012 *A Priority based Round Robin CPU Scheduling Algorithm for Real Time Systems*,International Journal of Innovations in Engineering and Technology (IJIET), Vol. **1** Issue 3 Oct 2012,ISSN: 2319 – 1058,pp 1-4

[3] Saroj Hiranwal and Dr.K.C.Roy 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. 319-321.

[4] N Srinivasu, A.S.V Balakrishna and R.Durgalakshmi 2015 *An Augmented Dynamic Round Robin CPU Scheduling Algorithm* Journal Of Theoritical and Applied Information Technology, Vol.**76**, No 1, pp . 119-124.

[5] Radhe Shyam and Sunil Kumar Nandal 2014 *Improved Mean Round Robin with Shortest Job First Scheduling* International Journal of Advanced Research in Computer Science and Software Engineering, Vo.**4**, Issue.7, pp. 170-177.

[6] Lalit Kishor and Dinesh Goyal 2013 *Time Quantum Based Improved Scheduled Algorithm* International Journal of Advanced Research in Computer Science and Software Engineering, Vol .**3**, Issue. 4,pp. 955-962.

[7] H.S.Behera,R.Mohanty and Debashree Nayak 2010 *A New Proposed Dynamic Quantum with Re-Adjusted Round Robin Scheduling Algorithm and its Performance Analysis* International Journal of Computer Applications, Vol. **5**-No.5, pp. 10-14.

[8] G.Siva Nageswara Rao, S.V.N. Srinivasu, N. Srinivasu and G. Ramakoteswara Rao 2015 *Enhanced Precedence Scheduling Algorithm with Dynamic Time Quantum (EPSADTQ)* Research Journal of Applied Sciences, Engineering and Technology vol **10** No.8: 938-941 ISSN: 2040-7459,pp.931-941