# DELHI TECHNOLOGICAL UNIVERSITY

# INNOVATIVE PROJECT REPORT

**OPERATING SYSTEMS DESIGN ( CO 204 )**

****

Submitted to – Mrs. Daya Gupta

Submitted by – Ritik Singh (2K19/CO/319)

**DELHI TECHNOLOGICAL UNIVERSITY**

**(Formerly Delhi College of Engineering)**

**Bawana Road, Delhi – 110042**

**An Optimized Round Robin Scheduling Algorithm**

**for CPU Scheduling**

**I. INTRODUCTION**

Scheduling is one of the core functions of an Operating System; it is the method that assigns processes to the CPU so that they can be executed. When more than one process is waiting to be executed, the scheduler utilizes a scheduling algorithm to make the decision of which process to run next. In an operating system, many different processes compete for CPU time at any given moment. Numerous algorithms have been developed to optimize the CPU time utilization, such as:

* First-Come, First-Served Scheduling Algorithm
* Shortest-Job-First Scheduling Algorithm
* Priority Scheduling Algorithm
* Round-Robin Scheduling Algorithm
* Multilevel Queue Scheduling Algorithm

The Round Robin Scheduling Algorithm is a type of scheduling algorithm mainly used by the operating system and applications that serve multiple clients that request to use resources. Each process is arranged in the ready queue in a first-come first-served manner, and the processor executes the task from the ready queue on the basis of a time slice. Even though the process might not have finished execution, once the time slice ends, the process is pushed to the back of the ready queue and the next process starts executing. It does this repeatedly until the jobs are finished.



Shows the Round Robin Algorithm taking processes from the ready queue and pushing them into the CPU for execution. When the quantum time slice expires, and the process is not complete, the process is pushed to the back of the ready queue. If the remaining time burst is equal to zero, the process terminates

In environments like the Real Time Operating System (RTOS), “the time slice should not be too small, since it could result in frequent context switches, and should be slightly greater than the average task computation time. The Round Robin Scheduling Algorithm is also “limited by high waiting and turnaround times and low throughput” making it unsuitable for a system like the RTOS .

According to Silberchatz, Galvin and Gagne, CPU scheduling play a vital role by switching the CPU among several process. The aim of operating system to allow a number of processes concurrently in order to maximize the CPU utilization. In a multi-programmed Operating system, a process is executed until it must wait for the completion of some input-output request.

Part of the reason for using multi-programming is that the operating system itself is implemented as one or more processes, so there must be a way for the operating system and application processes to share the CPU. Another main reason is the need for processes to perform I/O operations in the normal course of computation. Since I/O operations ordinarily require orders of magnitude more time to complete than do CPU instructions, multi-programming systems allocate the CPU to another process whenever a process invokes an I/O operation.

In RR a small unit of time is used which is called Time Quantum or Time slice. The CPU scheduler goes around the Ready Queue allocating the CPU to each process for a time interval up to 1 time quantum. If a process’s CPU burst exceeds 1 time quantum, that process is pre-empted

and is put back in the ready queue. If a new process arrives then it is added to the tail of the circular queue. Out of the above discussed algorithms RR provides better performance as compared to the others in case of a time sharing operating system. The performance of a scheduling algorithm depends upon the scheduling criteria viz. Turnaround time, Waiting time,

Response time, CPU utilization, and throughput.

**Turnaround time** is the time interval from the submission time of a process to the completion time of a process.

**Waiting time** is the sum of periods spent waiting in the ready queue. The time from the submission of a process until the first response is called Response time.

**CPU utilization** is the percentage of time CPU remains busy. The number of processes completed per unit time is called **Throughput**.

**Context switch** is the process of swap-out the pre-executed process from CPU and swap-in a new process to CPU. Context switch is the number of times the process switches to get execute. A scheduling algorithm can be optimized by minimizing response time, waiting time and turnaround time and by maximizing CPU utilization, throughput.

**II. Round Robin Scheduling**

* Round-robin scheduling is designed for time-sharing system.
* It is similar to the FCFS scheduling, but preemptive is added to switch between processes.
* A time quantum is typically 10 to 100 milliseconds.
* The ready queue is implemented in FIFO manner.
* If a process needs less than a time quantum, it releases the CPU voluntarily.
* If a process needs more than a time quantum, it is preempted from the CPU and placed at the back of the ready queue.

According to Silberchatz, Galvin, Gagne in operating system design and operating system by D M Dhamdhere, the simple RR scheduling algorithm is given by following steps:-

1. The schedular maintains a queue of ready

processes and a list of blocked and swapped out

processes.

2. The PCB of newly created process is added to end

of ready queue. The PCB of terminating process is

removed from the scheduling data structures.

3. The schedular always selects the PCB at head of

the ready queue.

4. When a running process finishes its slice, it is

moved to end of ready queue.

5. The event handler perform the following action

a) When a process makes an input -output request or

swapped out,its PCB is removed from ready queue to

blocked/swapped out list

b)When input-output operation awaited by a process

finishes or process is swapped in its process control block is

removed from blocked/swapped list to end of ready queue.

**III. Scheduling Objective**

A system designer must consider a variety of factors when developing a scheduling discipline ,such as what type of systems and what are user's

needs. Depending on the system, the user and designer might expect the scheduler to:

* ** Maximize throughput:** A scheduling discipline should attempt to service the maximum number of processes per unit of time.
*  **Avoid indefinite postponement and starvation:** A process should not experience an unbounded wait time before or while process service.
* **Minimize overhead:** Overhead causes wastage of resources. But when we use certain portion of system resources effectively,then overhead can

greatly improve overall system performance.

*  **Enforcement of priorities:** if system assign priorities to processes ,the scheduling mechanism should favour the higher-priority processes. Achieve balance between response and utilization:The scheduling mechanism should keep resources of system busy.
*  **Be predictable:** By minimizing statistical variance in process response times ,a system can guarantee that processes will receive predictable service levels.
*  **Favour processes exhibiting desirable behaviour.**
* ** Degrade gracefully under heavy load.**

**IV. Scheduling criteria:**

Different CPU-scheduling algorithms have different properties, and the choice of a particular algorithm may favor one class of processes over another. Which characteristics are used for comparison can make a substantial difference in which algorithm is judged to be best.

There are several different criteria to consider when trying to select the "best" scheduling algorithm for a particular situation and environment, including:

* Ideally the CPU would be busy 100% of the time, so as to waste 0 CPU cycles. On a real system CPU usage should range from 40% ( lightly loaded ) to 90% ( heavily loaded. )
* Number of processes completed per unit time. May range from 10/second to 1/hour depending on the specific processes.
* Time required for a particular process to complete, from submission time to completion. (Wall clock time.)
* is the sum of the times, processes spend in the ready queue waiting their turn to get on the CPU.
* Amount of time it takes from when a request was submitted until the first response is produced. Remember, it is the time till the first response and not the completion of process execution(final response).
* In general one wants to optimize the average value of a criteria ( Maximize CPU utilization and throughput, and minimize all the others. ) However some times one wants to do something different, such as to minimize the maximum response time.
* Sometimes it is most desirable to minimize the variance of a criteria than the actual value. i.e. users are more accepting of a consistent predictable system than an inconsistent one, even if it is a little bit slower.

**V. PROPOSED ALGORITHM**

Round robin Scheduling algorithm optimized by using dynamic time quantum. Implemented in C++.

* Round Robin scheduling algorithm is the widely used scheduling algorithm in multitasking and real time environment. It is the most popular algorithm due to its fairness and starvation free nature towards the processes, which is achieved by using the time quantum.
* Each scheduling algorithm has its own advantages and disadvantages. Similarly classical RR has a drawback which increases average waiting time, average turnaround time and minimizes the throughput, known as Context switch. The processes in RR are assigned with a time quantum which is static by nature.
* Dynamic time quantum refers to changing time quantum based on the burst times of processes present in ready queue.

**OPTIMIZED METHOD**

 In our proposed algorithm, we are arranging the processes in ascending order according to their burst time present in the ready queue.  For finding an optimal time quantum, median method is followed. The median can be calculated using the following formulae.

MEDIAN (M)= Y(n+2)/2 if n is odd

1/2(Y n/2 + Y (1+n)/2) if n is even

where n- no. of processes.

Y- The number in the middle when in ascending order.

Optimal time quantum (oqt) = (Highest Burst Time+Median)/2

**References**

[1] “What is Round Robin Scheduling (RRS)? - Definition from Techopedia,” Techopedia.com. [Online]. Available: <https://www.techopedia.com/> definition/9236/round-robin-scheduling-rrs. [Accessed: 27-Jun-2019].

[2] M. A. Mohammed, M. AbdulMajid, B. A. Mustafa and R. F. Ghani, "Queueing theory study of round robin versus priority dynamic quantum time round robin scheduling algorithms," 2015 4th International Conference on Software Engineering and Computer Systems (ICSECS), Kuantan, 2015, pp. 189-194.doi:10.1109/ ICSECS.2015.7333108.

[3] A. Silberschatz, P. B. Galvin, and G. Gagne, Operating System Concepts. John Wiley & Sons, Inc, 2013.

[4] S. Ghosh and C. Banerjee, "Dynamic Time Quantum Priority Based Round Robin for Load Balancing In Cloud Environment," 2018 Fourth International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN), Kolkata, India, 2018, pp. 33-37. doi: 10.1109/ICRCICN.2018.8718694

[5] A. Yasin, A. Faraz and S. Rehman, "Prioritized Fair Round Robin Algorithm with Variable Time Quantum," 2015 13th International Conference on Frontiers of Information Technology (FIT), Islamabad, 2015, pp. 314-319. doi: 10.1109/FIT.2015.62

[6] J. C. Villanueva, “Comparing Load Balancing Algorithms,” Comparing Load Balancing Algorithms. [Online]. Available: https://www.jscape.com/blog/load-balancing-algorithms..

[7] P. Krzyzanowski, “Process Scheduling,” Process Scheduling,

18-Feb-2015. [Online]. Available: <https://www.cs.rutgers.edu> /~pxk/416/ notes/

07-scheduling.html.

[8] A. Yasin, A. Faraz and S. Rehman, "Prioritized Fair Round Robin Algorithm with Variable Time Quantum," 2015 13th International Conference on Frontiers of Information Technology (FIT), Islamabad, 2015, pp. 314-319. doi: 10.1109/FIT.2015.62

[9] A. Alsheikhy, R. Ammar and R. Elfouly, "An improved dynamic Round Robin scheduling algorithm based on a variant quantum time," 2015 11th International Computer Engineering Conference (ICENCO), Cairo, 2015, pp. 98-104. doi: 10.1109/ICENCO.2015.7416332

[10] M. U. Farooq, A. Shakoor and A. B. Siddique, "An Efficient Dynamic Round Robin algorithm for CPU scheduling," 2017 International Conference on Communication, Computing and Digital Systems (C-CODE), Islamabad, 2017, pp. 244-248. doi: 10.1109/C-CODE.2017.791893

[11] <https://www.geeksforgeeks.org/round-robin-scheduling-with-different-arrival-times/>

[12] [https://www.idc-online.com/ technical\_references/pdfs/ information\_technology/ A%20Priority%20based.pdf](https://www.idc-online.com/technical_references/pdfs/information_technology/A%20Priority%20based.pdf)