"CHITTAGONG UNIVERSITY OF ENGINEERING&TECHNOLOGY"

Department of Computer Science and Engineering

Course Title : Operating Systems (Sessional)
Course No : CSE-336
Project Title : A Comparative Study of CPU Scheduling Algorithm

Project members:

Name	*ID*
SAYEDA TAHMINA	*1804034*
MD KAF SHAHRIER	*1804035*
ASHIK BILLAH FAHIM	*1804036*
SUHANA BINTA RASHID	*1804037*
HASAN MESBAUL ALI TAHER	*1804038*
MD OSAMA	*1804039*

Supervised by :

Md. Shafiul Alam Forhad
Assistant Professor,
Dept of CSE, CUET.

Hasan Murad
Lecturer,
Dept of CSE,CUET.

REMARKS

Abstract

The primary goal of this project is to create an efficient method for the round-robin CPU scheduling algorithm that reduces response time in real-time operating systems as compared to other current CPU scheduling algorithms. The dynamically suggested SJF, round-robin, and priority scheduling algorithms are used in the priority-based round-robin CPU scheduling technique to lessen famine and the idea of aging by giving the processes new priorities, but the response remains larger. Due to the larger response time, existing round-robin CPU scheduling algorithms cannot be utilized efficiently in the real-time operating system. For this, we have tried to implement such an idea that lessens the average response time in comparison to the existing round-robin.

Introduction

CPU Scheduling is a technique that makes full use of the CPU by allowing one process to run while another is delayed because no resources, such as I/O, are available. The operating system must choose one of the processes in the line of ready-to-launch processes whenever the CPU is idle. The Scheduler chooses between memory processes that are prepared to run and allows the CPU to one of them. CPU scheduling's primary responsibility is to make sure that whenever the CPU is idle, the OS at least chooses one of the processes in the ready queue to be executed.

Round-robin CPU scheduling is a form of preemptive scheduling that gives each process a specific amount of time (called a "time quantum") to complete its task. Round Robin processes all requests in a circular first-in-first-out (FIFO) order so that all processes and applications use the same resources at the same time and experience the same latency in each cycle and go precedence. The process scheduling mechanism known as "Priority Scheduling" prioritizes tasks, allowing the scheduler to choose which ones to do first. Processes having a higher priority, therefore, run first, followed by those with a lower priority.

This project represents the comparative analysis of the proposed algorithm with the existing CPU scheduling algorithm especially the round robin scheduling algorithm on the basis of varying time quantum, average waiting time, average turnaround time, and response time. It retains the advantage of round robin in reducing response time.

Related Works

Round Robin is the preemptive process scheduling algorithm. Each process is provided a fixed time slice to execute, it is called time quantum. Once a process is executed for a given period, it is preempted and another process executes for a given period. Context switching is used to save states of preempted processes.

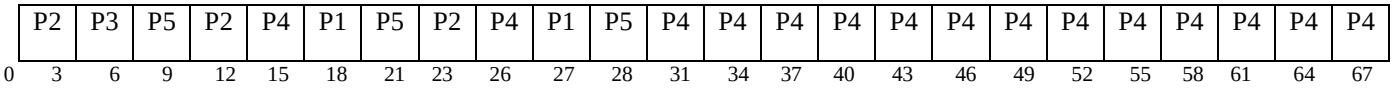
Round-robin (RR) scheduling algorithms are primarily designed for time-shared systems. This algorithm is similar to FCFS scheduling, but Round Robin (RR) scheduling adds preemption and allows the system to switch processes. This algorithm spends more time on context switches and offers a larger **response time**.

For example,

TQ = 3 units,

Process Id	Arrival Time	Burst Time	End Time	Turnaround Time	Waiting Time	Response Time
P1	5	4	27	20	18	10
P2	4	8	23	23	15	0
P3	2	3	6	4	1	1
P4	0	45	67	63	18	8
P5	3	7	28	25	18	3

Gantt chart:



Average Response Time: 4.4

Average Turnaround Time: 27.4

Average Waiting Time: 14

Methodology

In our proposed algorithm we have focused on reducing the Average Response Time of processes and thus overcoming the STARVATION problem. Unlike the Round Robin algorithm, our proposed one gives chance to those processes (which are in Ready Queue) to be executed in CPU that have accessed CPU for the minimum number of times. Thus if a process is ahead of another process in Ready Queue, but there is another process that has accessed the CPU lesser than the previous one, will get a chance to be executed in the CPU. Through this, the **Response Time** of processes will decrease drastically and the chance of being in **STARVATION** will be minimized.

The algorithm performs the following steps:

Step 1: Unlike the Round Robin algorithm, our proposed one gives chance to those processes (which are in Ready Queue) to be executed in CPU that have accessed CPU for the minimum number of times.

Step 2: After completion of first step following steps are performed:

i. If a process is ahead of another process in Ready Queue, but there is another process that has accessed the CPU lesser than the previous one, will get a chance to be executed in the CPU The processes are executed according to the new priorities based on the remaining CPU bursts, and each process gets the control of the CPU until they finished their execution.
ii. Continue step I until all the processes execution time is finished.

Step 3: If all the processes are completed terminate the program and exit.

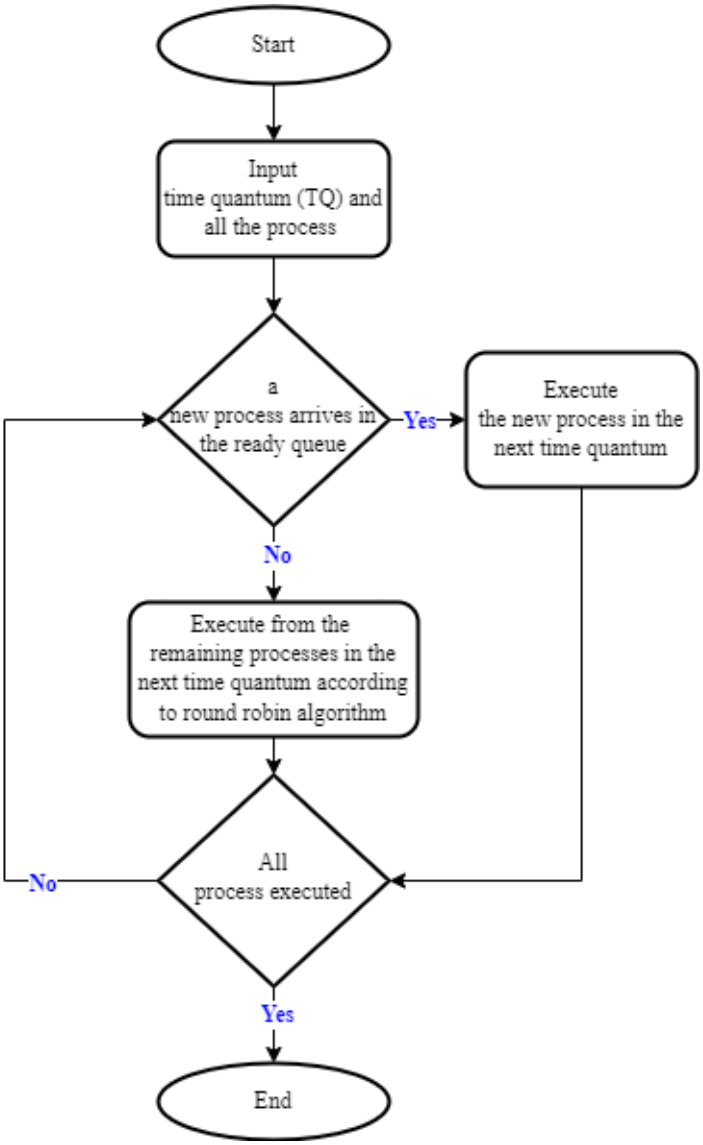


Figure 01: Flow chart of our proposed algorithm

Experiment

Five processes have been defined with CPU burst time and their arrival time, these five processes are scheduled in round robin procedures and also according to the proposed algorithm. The average response time, average waiting time, and average turnaround time has been calculated and the results were compared.

Test Case:

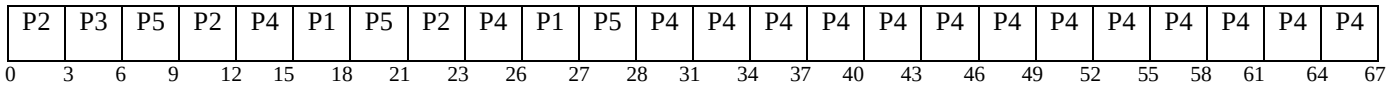
Table 01: Number of Processes, Burst time, and response time

Process Id	Arrival Time	Burst Time	End Time	Turnaround Time	Waiting Time	Response Time
P1	5	4	25	20	16	7
P2	0	8	27	27	19	0
P3	2	3	6	4	1	1
P4	4	45	67	63	18	5
P5	3	7	28	25	18	3

Now consider the time quantum 3 units

TQ = 3 units

Gantt chart for simple Round Robin CPU scheduling algorithm:



Average Response Time: 4.4

Average Turnaround Time: 27.4

Average Waiting Time: 14

Gantt chart for Proposed CPU scheduling algorithm:

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

0 3 6 9 12 15 18 21 23 26 27 28 31 34 37 40 43 46 49 52 55 58 61 64 67

Average Response Time: 3.2

Average Turnaround Time: 27.8

Average Waiting Time: 14.4

```
Number of process,n: 5

Enter time quantum: 3
Enter the arrival time of P1: 5
Enter the burst time of P1: 4

Enter the arrival time of P2: 0
Enter the burst time of P2: 8

Enter the arrival time of P3: 2
Enter the burst time of P3: 3

Enter the arrival time of P4: 4
Enter the burst time of P4: 45

Enter the arrival time of P5: 3
Enter the burst time of P5: 7

------------------------------------------------Proposed_Algorithm------------------------------------------------
Gantt Chart
0 P2 3 P3 6 P5 9 P4 12 P1 15 P2 18 P5 21 P4 24 P1 25 P2 27 P5 28 P4 31 P4 34 P4 37 P4 40 P4 43 P4 46 P4 49 P4 52 P4 55 P4 58 P4 61 P4 64 P4 67
Process: P1  Finish time: 25  Response time: 7 Waiting time: 16  Turnaround time: 20
Process: P2  Finish time: 27  Response time: 0 Waiting time: 19  Turnaround time: 27
Process: P3  Finish time: 6  Response time: 1 Waiting time: 1  Turnaround time: 4
Process: P4  Finish time: 67  Response time: 5 Waiting time: 18  Turnaround time: 63
Process: P5  Finish time: 28  Response time: 3 Waiting time: 18  Turnaround time: 25

Average waiting time : 14.4
Average turnaround time : 27.8
Average Response time : 3.2
Idle time: 0
Press any key for the home page
```

Figure 02: Proposed algorithm for time quantum 3 units

Results

The performance of the two algorithms can be compared by considering the response time, average waiting time, and average turnaround time. The proposed algorithm performs better over simple round robin for varying time quantum. Here we notice that the response of the proposed algorithm is reduced in comparison to round robin.

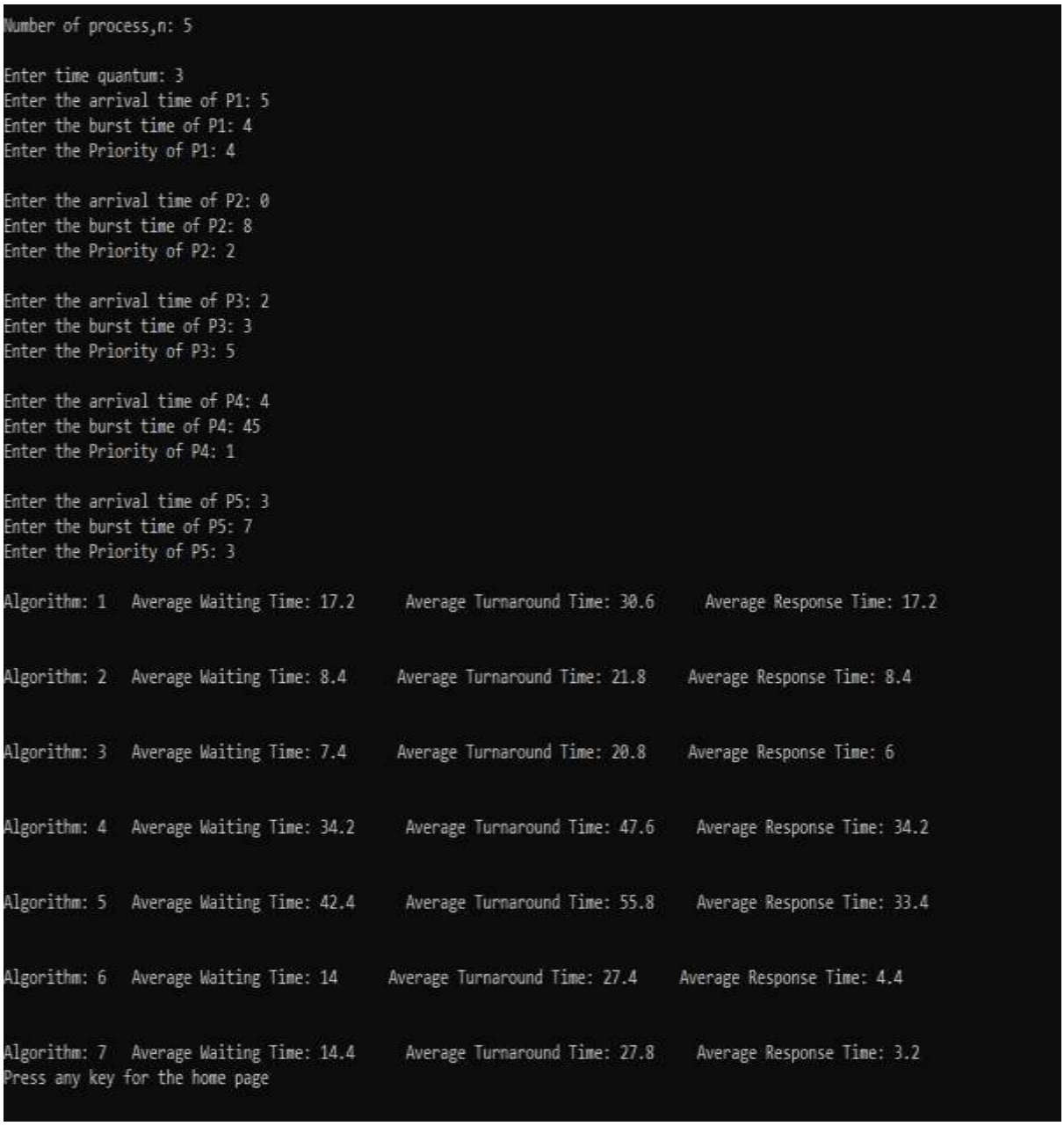


Figure 03: Comparison of the proposed algorithm with other existing algorithms.

Graphical representation of response time with other scheduling algorithms:

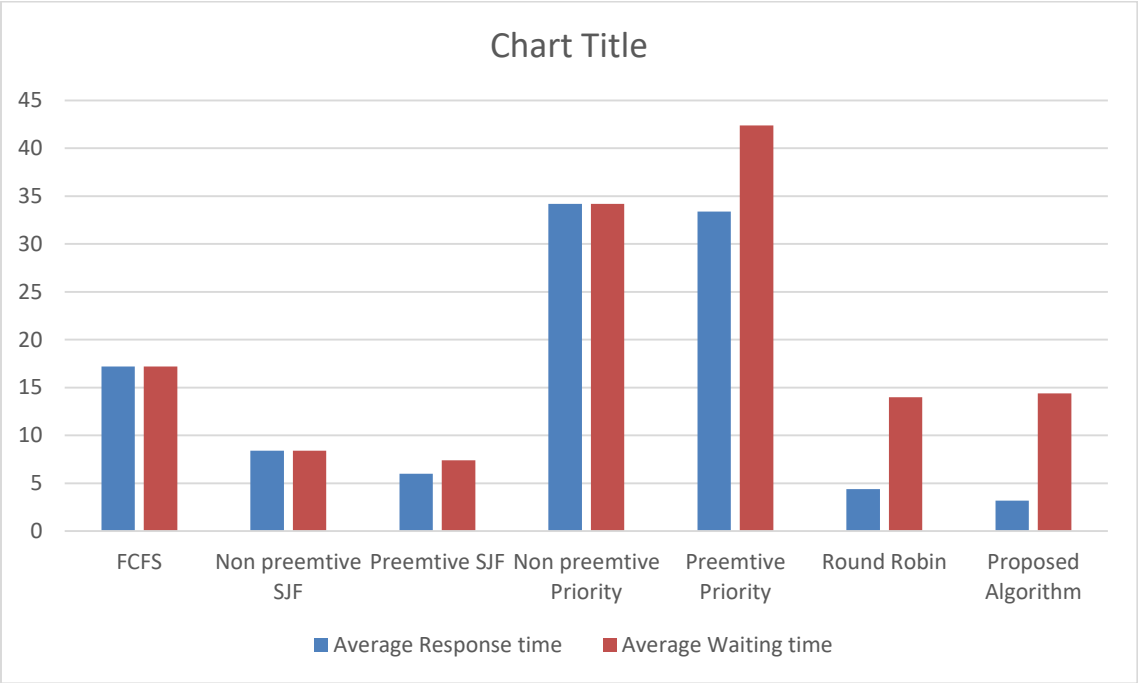


Figure 04: Comparison of the proposed algorithm with other existing algorithms.

From the comparison charts, we say that our proposed algorithm doing better performance compared to a simple round-robin algorithm by reducing the response time. It minimizes average response time.

Conclusion

Throughout this project, we have implemented 6 famous scheduling algorithms and one proposed algorithm. All the scheduling algorithms are applied to find the scheduling criteria such as – average Response Time, average Turnaround Time, average waiting time, etc. Moreover, we have also proposed a completely new algorithm that intends to reduce the average Response Time of the processes drastically. Apart from that this proposed algorithm also helps to overcome the Starvation problem. Though this algorithm does not perform its best in all cases but performs much effect when cases of Starvation arise.