*A Report on*

**Performance Analysis of two Process Scheduling Algorithms- FCFS and Round Robin for Different workloads**

BY

**Harsh Daryani 2018B1A70645H**

**Virag Lakhani 2018B3A70973H**

**Jui Pradhan 2018B3A70984H**

**Deepanshu Mody 2018B1A70949H**

**Rohit Sethi 2018B4A70611H**

**Saaketh Reddy Vaddi 2019A7PS1215H**

**Rohan Sachan 2018B3A70992H**

**Pranav Khabale 2018B1A70794H**

**Aaryan Gupta 2018B1A70775H**

**Group No.: 45**

Under the supervision of

**Dr. Barsha Mitra**

**SUBMITTED IN FULFILLMENT OF THE REQUIREMENTS OF**

**CS F372: OPERATING SYSTEMS**

**Assignment**



**BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE PILANI (RAJASTHAN)**

**HYDERABAD CAMPUS**

**(November 2021**)

**ACKNOWLEDGMENTS**

We wish to express our gratitude towards Prof. Barsha Mitra, Professor, Department of CSIS for her continuous and enthusiastic support, cooperation and help throughout the duration of the assignment. We would also like to thank Prof. Sridhar Raju, Associate Dean, Academic and Undergraduate Studies Division and Prof. Runa Kumari, Associate Dean, Timetable Division, BITS Pilani, Hyderabad Campus for giving us the opportunity to register.

**Introduction**

**Following are the Libraries included:**

<sys/types.h>

<string.h>

<stdio.h>

<stdlib.h>

<unistd.h>

<sys/types.h>

<sys/wait.h>

<pthread.h>

<errno.h>

<sys/ipc.h>

<sys/shm.h>

<stdio.h>

<sys/mman.h>

**The code uses the concepts of queue, threads, file handling, pthread, mutex, shared memory, pipes, clock\_gettime, sleep, thread scheduling etc.**

**Performance Analysis**

* The CPU scheduling algorithm doesn’t affect the amount of time during which a process executes or does I/O but only the amount of time that a process spends waiting in the ready queue so it affects the waiting and turnaround time.
* We can observe that the turnaround time is generally limited by the speed of the output device.
* From the graphs we can see that as the n value increases, the turnaround and waiting time both increase. It is almost a linear increase in both cases, FCFS and RR. This was obvious as the process gets more computationally complicated as we increase n,so it takes more time to complete the process.
* The process creation is in the order C1,C2,C3 so the average waiting times increase in that order for FCFS scheduling. The same trend follows for RR because of the same reason.
* We see some spikes in the graph which are the noise occurring due to the limitation of computers and other simultaneous processes running in the background.

**For Lower Workload (less value of n1,n2,n3)**

RR should have lower average time because if any one process is longer, the waiting time of the following processes increases significantly making FCFS’s waiting time more, but we on the contrary have found that FCFS is better than RR (for lower worldload)( has lower average waiting time) because the process C1 takes least time so process C2 and C3 do not have to wait for long. This may also be due to the value of quantums which may lead to multiple shifting which increases the overhead time.

**For the same values of input(for lower workload, n<1000 we observed that the average turnaround time and average waiting time of the 3 processes is less for FCFS as compared to Round robin.**

**For eg:**

For n1=100, n2=100, n3=100, time quantum =0.0002 we observe that the average turnaround time =2.001542 sec and average waiting time= 2.002655 sec for Round Robin whereas for FCFS, average turnaround time = 2.000695 sec and average waiting time= 2.001442 sec.

**As we can see from the above example.**

**This could be due to the fact that Round Robin involves many context switching due to low time quantum so has a greater overhead than fcfs, also for low workload, no process is consuming the processor for a longer period of time hence we get a lower waiting time and hence turnaround time for fcfs.**

**Hence, FCFS is performing better for low workload as compared to Round robin.**

**For Larger Workload (more value of n1,n2,n3)**

**For larger workloads, both FCFS and RR have almost similar Turnaround and waiting Time.** However the average Turnaround and waiting Time of RR in large workloads is slightly lower than that of FCFS. Therefore, RR is slightly better than FCFS for larger workloads.

The reason for this could be that for larger workloads processes have high burst time due to which waiting time of processes scheduled later increases and hence turn around time also increases, hence its values for greater than RR.

**Round robin scheduling analysis for Different time quantum value**

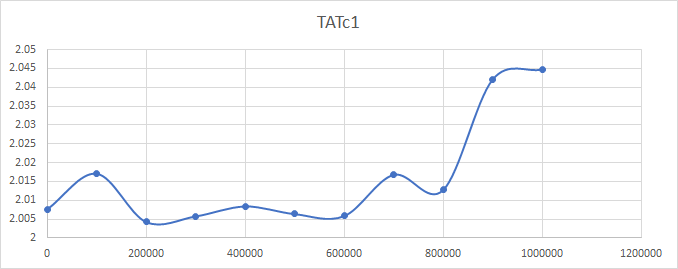
* The Round Robin scheduling algorithm works better for smaller inputs(but should be large enough as compared to context switching overhead) of time quanta. And, there is a lot of continuous process switching, but with less overheads.
* Whereas, for the larger values of time quanta the Turnaround Time becomes almost the same as FCFS. This is because when q is large, RR starts to act like FCFS; the large enough quanta makes no switching before the process terminates.
* Imagine an RR algorithm with a time slice so large that it is equal or larger than the burst time of the longest process, then each process will be executed in the manner they arrive into the ready queue. There will be no preemption. It will act just like the FCFS algorithm. The average waiting time gets almost constant by the time we reach process 3 as the large enough quanta makes no switching.
* Therefore, as we increase the values of time quanta the behaviour of both the algorithms is similar.

**GRAPHS**

**C1**

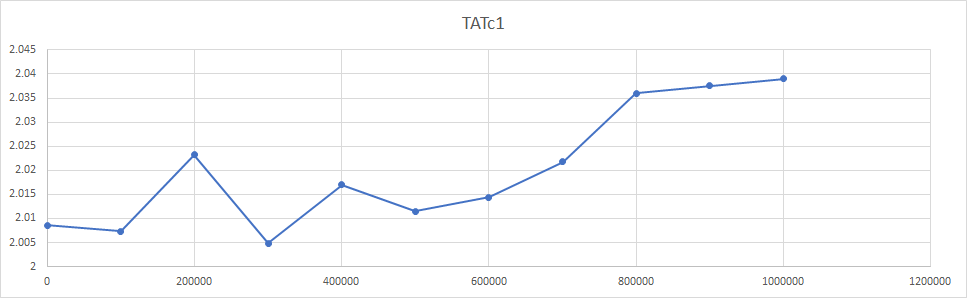
**FCFS Scheduling Algorithm**

**Graphs- Performance Analysis:** Turnaround Time vs Workload



**Round Robin Scheduling Algorithm**

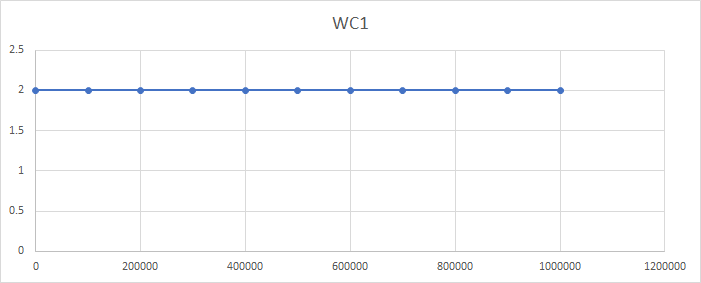
**Graphs- Performance Analysis:** Turnaround Time vs Workload



**C1**

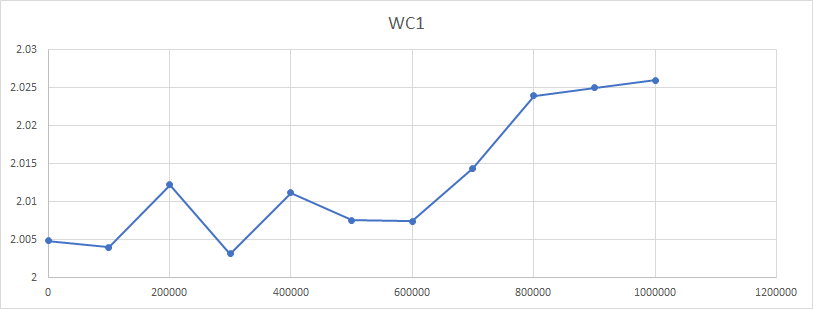
**FCFS Scheduling Algorithm**

**Graphs- Performance Analysis:** Waiting Time vs Workload



**Round Robin Scheduling Algorithm**

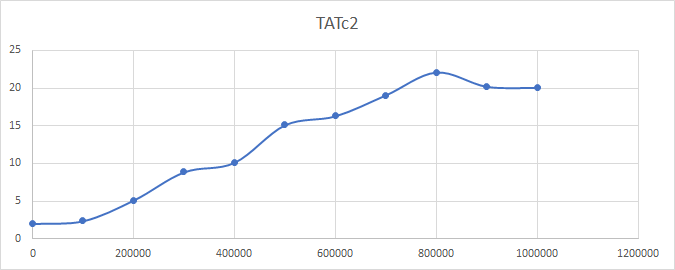
**Graphs- Performance Analysis:** Waiting Time vs Workload



**C2**

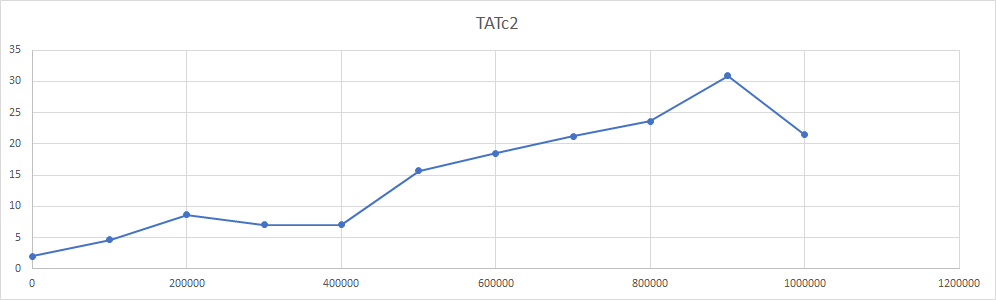
**FCFS Scheduling Algorithm**

**Graphs- Performance Analysis:** Turnaround Time vs Workload



**Round Robin Scheduling Algorithm**

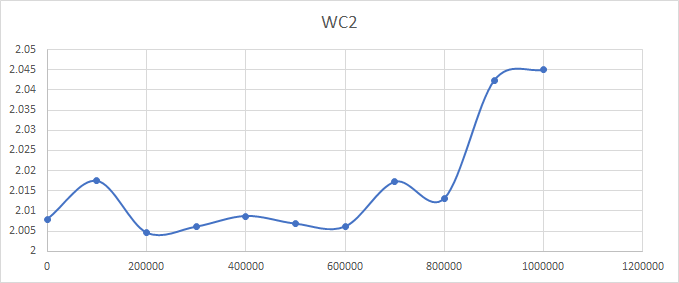
**Graphs- Performance Analysis:** Turnaround Time vs Workload



**C2**

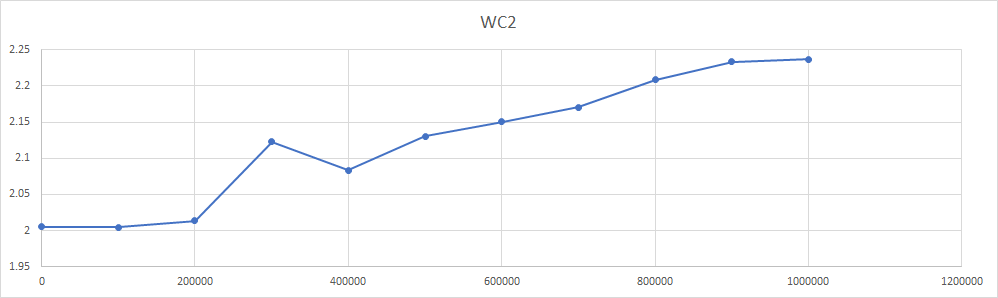
**FCFS Scheduling Algorithm**

**Graphs- Performance Analysis:** Waiting Time vs Workload



**Round Robin Scheduling Algorithm**

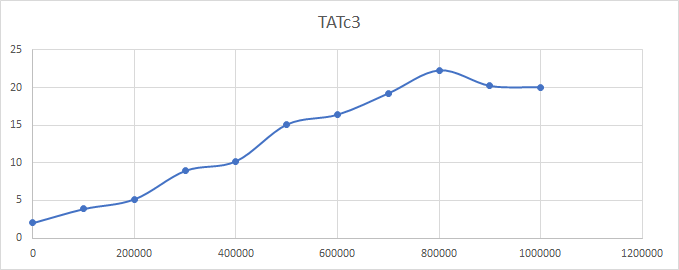
**Graphs- Performance Analysis:** Waiting Time vs Workload



**C3**

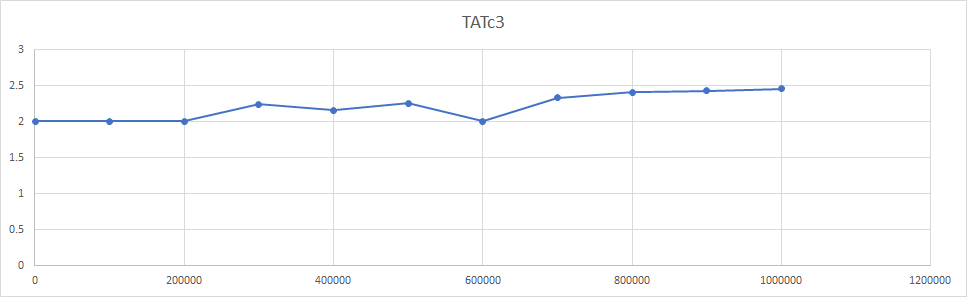
**FCFS Scheduling Algorithm**

**Graphs- Performance Analysis:** Turnaround Time vs Workload



**Round Robin Scheduling Algorithm**

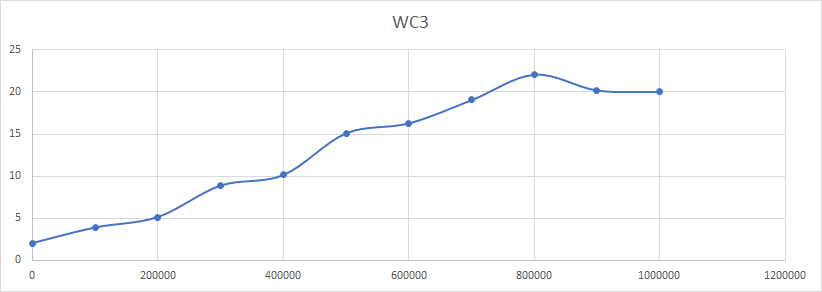
**Graphs- Performance Analysis:** Turnaround Time vs Workload



**C3**

**FCFS Scheduling Algorithm**

**Graphs- Performance Analysis:** Waiting Time vs Workload



**Round Robin Scheduling Algorithm**

**Graphs- Performance Analysis:** Waiting Time vs Workload

