COEN 283: Operating Systems

Winter 2016

Final Report

Topic: Analysis of various scheduling algorithms

Mentor: Prof. Amr Elkady

Title of Project: Process Scheduler Simulator

Team Name: Phoenix(7)

Team Members: Rama Gupta

Saurabh Sathaye

Shubham Goyal

Table of Contents:

List of figures and Graphs:

Abstract:

Scheduling algorithm is a very important and highly researched concept of an operating system. In multiprogramming system scheduling is very complex process as there are multiple processes waiting for services of the CPU. System's response time, throughput capacity and CPU utilization for some processes depends on the scheduling algorithm used by the system. So it is important to implement a proper scheduling algorithm for efficient use of resources.

This project work in the field of CPU scheduling includes carefully studying and implementing all popular scheduling algorithms thereby proposing a new algorithm Multilevel FeedBack Queue with Shortest Job first. The major problem leading to starvation in multilevel Queue was reduced by proposing a model into the MLQ. The model was designed and tested, thereby suggesting ways by which Multilevel Feedback queue waiting time and average response time can be improved.

1. Introduction:

Today most of the computers perform multiprogramming however, the processor can execute only one instruction at a time and hence only one process can get the CPU control at a time. To achieve multiprogramming CPU allocation has to be managed and proper process scheduling should be done so that each process gets an opportunity to run enabling maximum throughput and CPU utilization with minimum waiting time and turn around time. No single scheduling algorithm is best for every situation , hence there is a need to analyze various scheduling algorithms and come up with the algorithm that proves to be most efficient in a particular situation. Moreover, a tool is required that gives a graphical representation of the order of execution/completion, turn around time and burst time which in turn assists in the study and evaluate these Scheduling Algorithms.

Any CPU scheduling algorithm relies on the following criteria. They are:

(A) Response time : Response time is the time from the arrival of a request up until the first response is produced.

(B) Turnaround Time: The time interval from the time of arrival of a process to the time of completion is the turnaround time. Total turnaround time is calculation is the sum of the time periods spent waiting to get into memory, waiting in the ready queue, executing on the CPU, and doing I/O.

(C) Waiting Time: The waiting time is not the measurement of time when a process executes or does I/O completion; it affects only the amount of time of submission of a process spends waiting in the ready queue. We keep average waiting time should be less.

1.1 Objectives:

The objectives of this work is to minimise the overall waiting time and response time of the processes and schedule them according to the priority and to Build a JavaFx based simulator for CPU scheduling that can display the results in the form of various charts. It performs a comparative study between different algorithms under different situations and requirements of the system. This tool is used to display the outcomes of the algorithms and measure their performance. We can use the tool to see the process states in the CPU at a particular time unit and visualize

1.2 Project Contribution:

Rama: Research about all the algorithms, Implementing First Come First Serve, Shortest Job First Scheduling, Improved Round Robin Scheduling, Extensive Testing of the application (30-40% work).

Saurabh: Implementing MultiLevel Feedback Queue, MultiLevel Feedback Queue with Shortest Job First, Designing the UI of the application using JavaFX.(30-40% work).

Shubham: Implementing the Charts, Improving the algorithms by including the IO time, handling the dynamic Chart Gantt Chart (30-40% work).

1.3 Summary:

A simulator is designed to measure the performance of the algorithms based on the average response time and turn around time. In addition, We modified the MLFQ scheduling algorithm to improve the performance. Traditional MLFQ uses Round Robin scheme for the jobs in the same queue. Instead of Round Robin selecting the next job in the queue, the performance of the algorithm improved when we picked the shortest job first. This tool can be used to analyze the performance of the following CPU scheduling algorithms:

1) First Come First Serve (FCFS)
2) Shortest Job First Scheduling (SJF)
3) Improved Round Robin Scheduling (RR)
4) MultiLevel FeedBack Queue (MLFQ)
5) MultiLevel FeedBack Queue with SJF (MLFQSJF)

1.4 Data Required:

Following data are required for the visualization and performance analysis :

1) Number of processes
2) The Arrival time of each process
3) Burst Time 1 for each process
4) Time quantum for Round robin and Multilevel feedback queue
5) I/O time for each process
6) Burst Time 2 for each process

1.5 Theory:

Process scheduling algorithms has been an interesting field of study in Operating Systems. Scheduling is a key concept in computer multitasking, multiprocessing and real-time operating system designs.

The process state is as follows:

1. **Start**: Processes enters the system.
2. **Ready**: It is a stage where queues are used to order the manner in which jobs arrived.
3. **Running**: Scheduler and Dispatcher, based on selected algorithm moves Ready jobs to running.
4. **Waiting**: On a request for an I/O and later rescheduled on completion of request.
5. **End/Exit**: Process terminates.

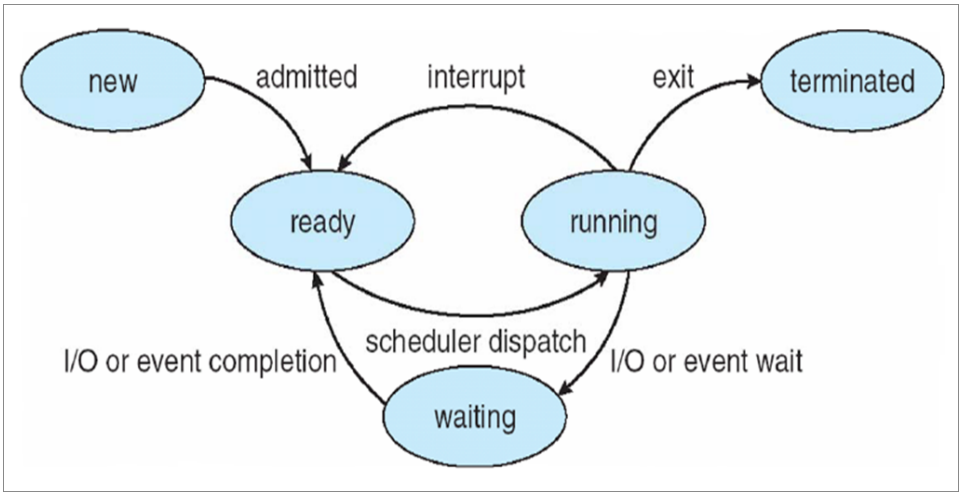

Figure 1: Process state diagram

2. Discussion :

The following scheduling algorithms are used in the system:

2.1 First Come, First Serve (FCFS):

It is the simplest of the Scheduling algorithms. The processes are assigned the CPU in the order of which come. It is a Non preemptive algorithm. the main advantage of FCFS is that it is easy to understand .A linked list is used to store ready processes. But it has its own disadvantages like high waiting and turnaround time. If there are jobs with very long waiting time the shorter jobs will face high waiting time.

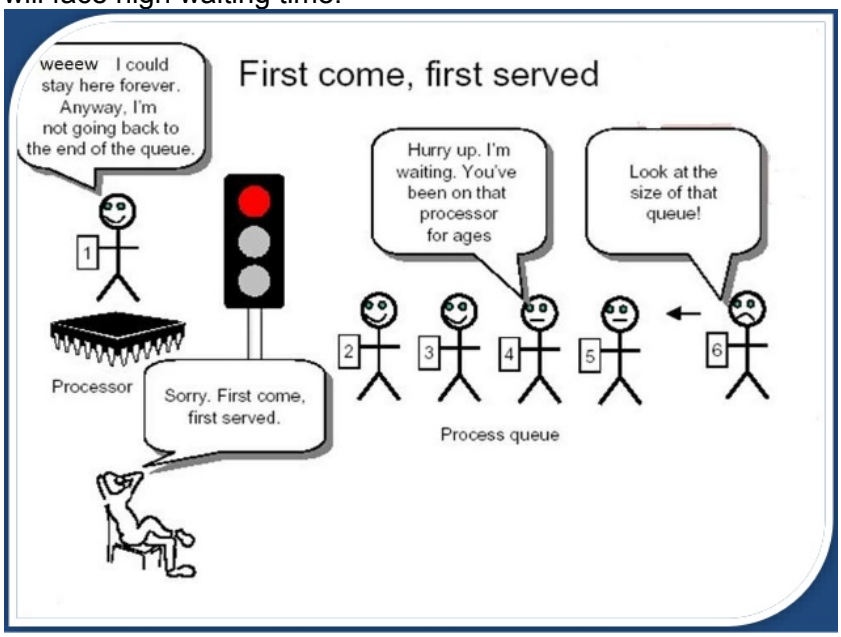

Figure 2. First-come-First-serve

2.2 Shortest Job First (SJF):

The job which has shortest burst time is allocated the CPU first. it is also a Non Preemptive scheduling algorithm. the advantage of using SJF is that the throughput is high and less average waiting time. the disadvantage is that it requires future knowledge of the burst time.

2.3 Round-Robin (RR):

It is a Preemptive algorithm. Each process runs for a fixed amount of time which is the time Quantum. If the time quantum is over, the CPU is preempted and next process runs. It is a fair algorithm and there is no chance of starvation of any process. the turnaround time is better than FCFS. But it has disadvantages like if the time slice is too short it may lead in significant overhead as there is a lot of context switching. With the increase in time quantum the average waiting time improves as there is less switching overhead. but it also results in a high response time.further, a too large time quantum makes it a FCFS algorithm.

To resolve the problem of selecting the time quantum we implemented Improved Round Robin where time quantum is taken as the average of the Burst time of the processes.

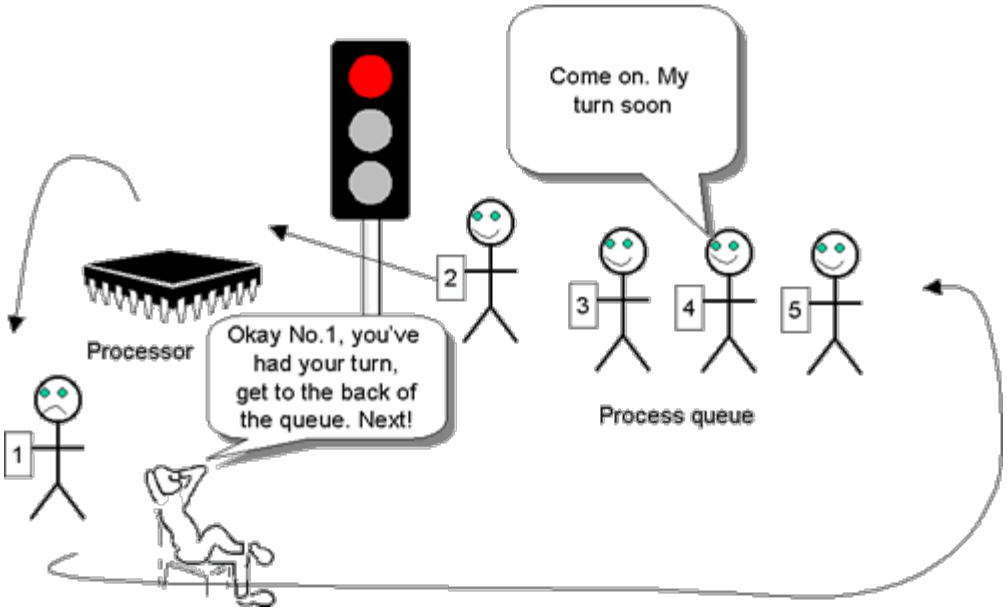

Figure 3: Round Robin scheduling

2.4 Multilevel Feedback Queue(MLFQ):

The Multilevel Feedback Queue (MLFQ) scheduler was first described by Corbato et al. in 1962. It was initially described in Compatible Time-Sharing System (CTSS). The same algorithm was then in implemented in Multics. An ideal scheduler algorithm should tend to minimize the turn-around-time and maximize the response time. MLFQ tries to achieve these by using multiple ready queues. MLFQ tries to approximate the running time of a particular process using multiple queues. Higher level queues have higher priority than lower queues. If there are jobs in higher queue then jobs in the lower queue will be blocked. Important part of the MLFQ is that it changes the priority of the processes depending upon their behavior. This helps improve response time and prevents starvation of jobs in the lower queues. When there are multiple jobs in the queue the jobs are scheduled using round robin scheduler scheme. The decision about the priority of a particular is taken by the algorithm depending on several conditions. The decision of the priority is done in two cases:

- the processes utilizes the complete time quantum and still has processing time left. In such a case the priority of the process is reduced.
- the process relinquishes the processor without using the entire time quantum. In this case the process might go into I/O state. In such condition the priority of the process is not changed and the process continues to stay in the same queue.

This is how the MLFQ is able to approximate the total running time of the process. If the process uses up the time quantum then it is interpreted as a long process. In such
situation the process lowers its priority and makes way for other “shorter” process. This avoids long burst process eating up all the CPU time thus avoiding starvation of shorter processes.
In second condition, if the process gives up CPU before time slice ends, this is interpreted as the process might be doing I/O or waiting for a user feedback. Such a process can be an interactive process. Hence it is kept in same queue without lowering its priority. This makes this scheduler ideal for responsive applications as processes that are responsible for user interaction are always kept in a higher priority queue.

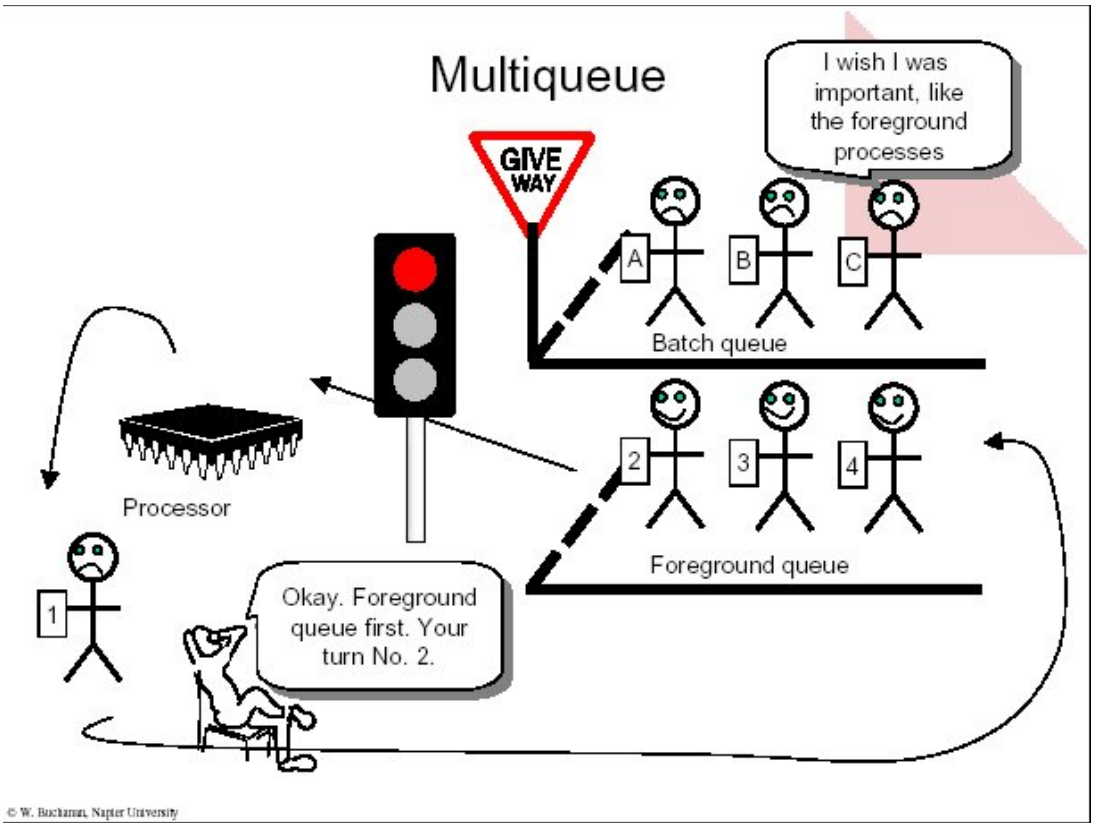

Figure 4: MultiLevel Feedback Queue source:http://homepages.uel.ac.uk/u8902383/scheduling_policies.htm

3. Implementation & Results:

The project was implemented in Java. The visualization was rendered using JavaFX library. Currently the tools takes input for 5 processes with their respective stats. The stats mainly include first burst time, then I/O time for each process and second burst time if required. One of the future works include taking in random number of processes which will help to stress test the application and check the performance of each algorithm against large inputs. Along with these a priority for each process is also taken from the user. This is used in multilevel feedback queue algorithms. The time quantum has currently been fixed to 3 units of a time. On inputting the numbers the user has to hit the 'Calculate' button to calculate all the numbers. The self explanatory button viz. "Turn Around Times", "Waiting Times" display the respective values in form of graphs for each process per scheduling algorithm and the average for all the algorithms. The tool allows to visualize a dynamic Gantt chart and a static Gantt chart. The dynamic Gantt shows the progress for each algorithm and how each process is getting schedule. One thread is assigned per algorithm to carry out the calculations.

3.1 Inputs :

Arrival Time	1st Burst Time	I/O Time	2nd Burst Time	Priority
0	4	4	4	0
2	8	1	8	2
3	2	1	2	1
7	1	1	1	0
19	10	2	10	1

3.2 Turnaround Time:

Process	FCFS	SJF	RR	MLFQ	MLFQ-SJF
Process 1	19	22	22	23	17
Process 2	25	28	43	38	38
Process 3	26	14	11	5	9
Process 4	23	11	12	6	6
Process 5	33	33	31	31	31
Average	25	21	23	20	20

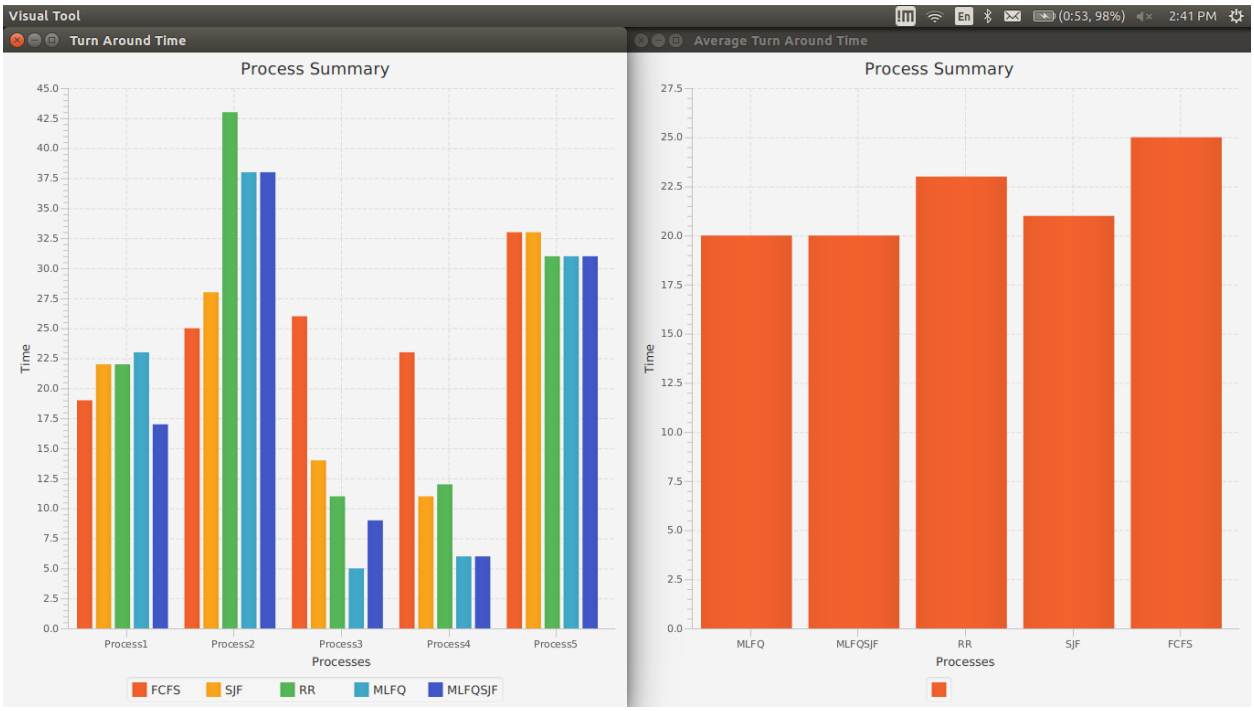

Figure 5: Turn Around Time

3.3 Waiting Time:

Process	FCFS	SJF	RR	MLFQ	MLFQ-SJF
Process 1	7	10	5	11	5
Process 2	8	11	16	21	6
Process 3	21	7	3	0	1
Process 4	20	8	4	2	1
Process 5	11	11	2	9	6
Average	13	9	6	8	3

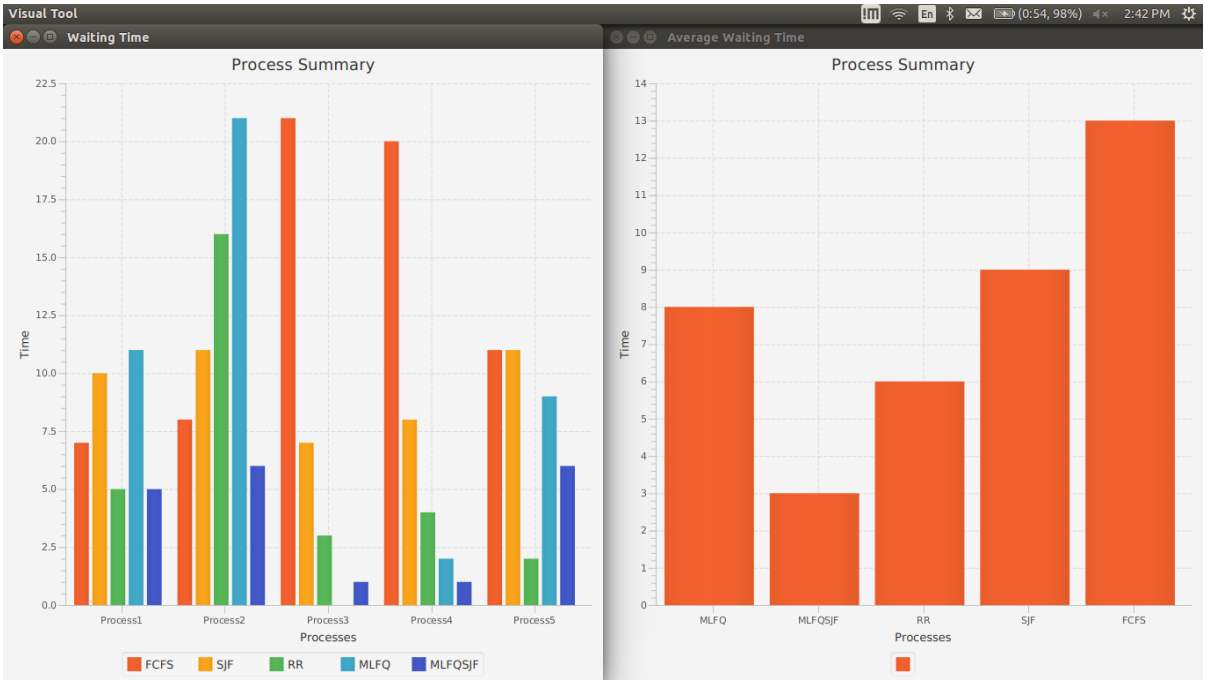

Figure 6: Waiting Time

3.4 Response Time:

Process	FCFS	SJF	RR	MLFQ	MLFQ-SJF
Process 1	0	0	0	0	0
Process 2	2	4	1	7	4
Process 3	9	1	3	0	1
Process 4	7	7	4	1	2
Process 5	11	11	3	0	1
Average	5	4	2	1	1

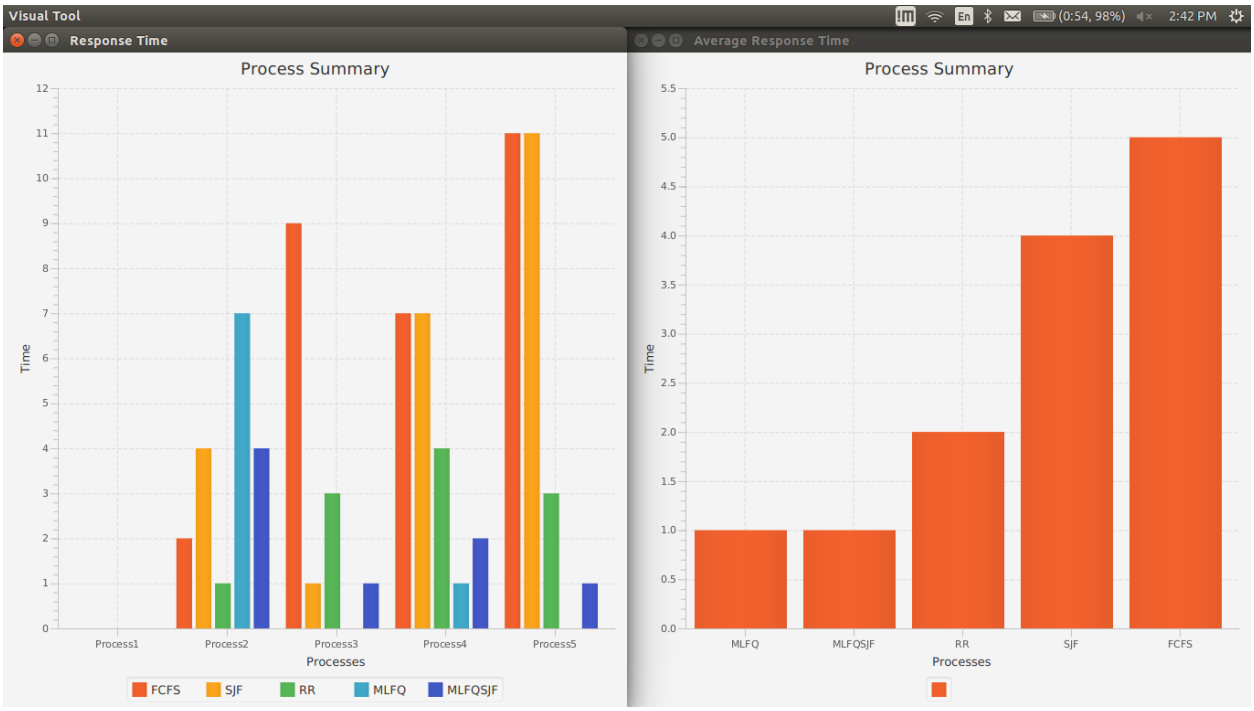

Figure 7: Response Time

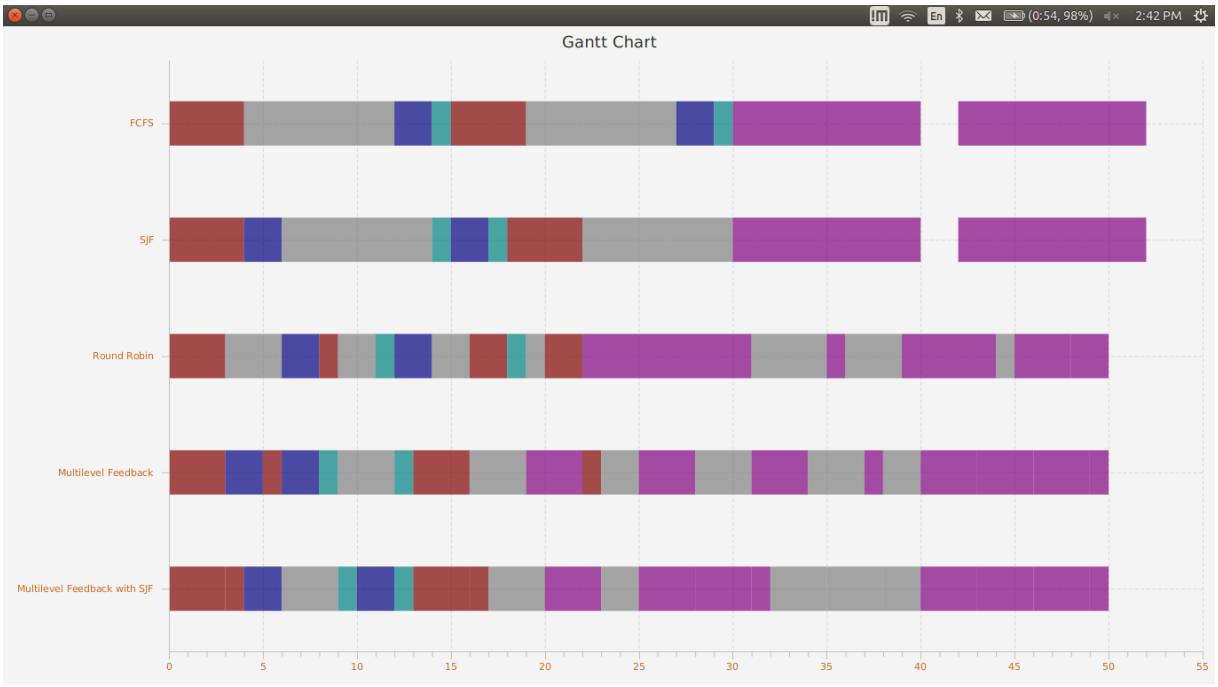

Figure 8: Static Gantt Chart

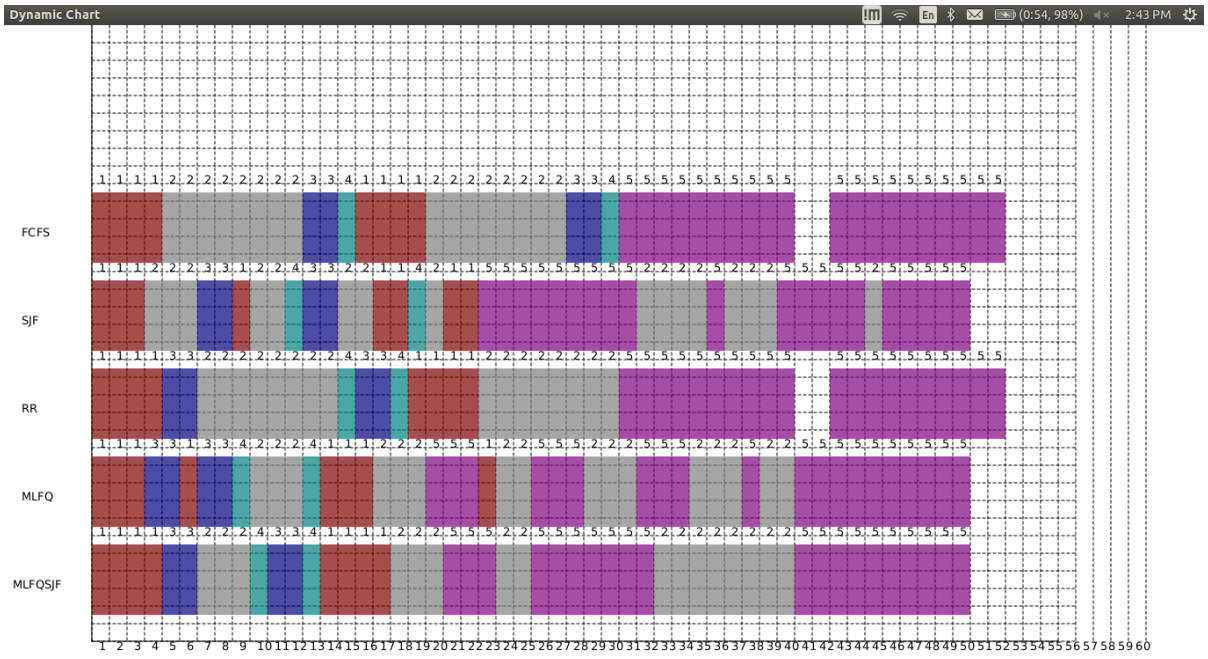

Figure 9: Dynamic Gantt Chart

We can see that when we replace the FCFS in the standard MLFQ with SJF the Turn Around Time and Waiting time for the processes improved significantly.

4. Conclusion and Future Work:

In this project we implemented an improved version of Multilevel Feedback Queue by scheduling the shortest job in the queue first. The change improved the turnaround time and waiting time significantly.
However, it is rather difficult to determine the burst time of a particular process before hand. In order to obtain an efficient scheduling algorithm, a rule-based fuzzy decision maker can be used to overcome this shortcoming. Using fuzzy decision maker the average waiting time and the average turnaround time can be improved close to that obtained from preemptive shortest-job-first (SJF) scheduling. Improving efficiency of round robin scheduling using neuro-fuzzy approach by increasing throughput and decrease waiting time as well as turnaround time for a process is also one area of research for future work.

5. References:

1. H.S. Behera, Reena Kumari Naik, Suchilagna Parida “Improved Multilevel Feedback Queue Scheduling Using Dynamic Time Quantum and Its Performance Analysis”. (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 3 (2) , 2012,3801-3807
2. Rami J. Matarneh, “Self-Adjustment Time Quantum in Round Robin Algorithm Depending on Burst Time of the Now Running Processes”, American Journal of Applied Sciences 6 (10): 1831-1837, 2009 ISSN 1546-9239 © 2009 Science Publications.
3. Saleem U, Javed M.Y “Simulation of CPU Scheduling algorithms”, TENCON 2000,IEEE Vol. 2 pages 562-567 (2).
4. Modern Operating System, Andrew S Tannenbaum, Herbert Bos 4th Edition.
5. http://stackoverflow.com/questions/27975898/gantt-chart-from-scratch
6. Bashir Alam, M. N. Doja, R. Biswas, M. Alam, “Fuzzy Priority CPU Scheduling Algorithm” Department JCSI International Journal of Computer Science Issues, Vol. 8, Issue 6, No 1, November 2011 ISSN (Online): 1694-0814
7. Shatha Jawad “Design and Evaluation of a Neurofuzzy CPU Scheduling Algorithm”, Software Engineering Department, The Eastridge Group of Staffing Companies.