CS-342 Operating Systems

Project 2

Multiprocessor Scheduling, Threads, Synchronization

Group Members:

- Elifsena Öz 22002245
- Mehmet Onur Uysal 22002609

Part A)

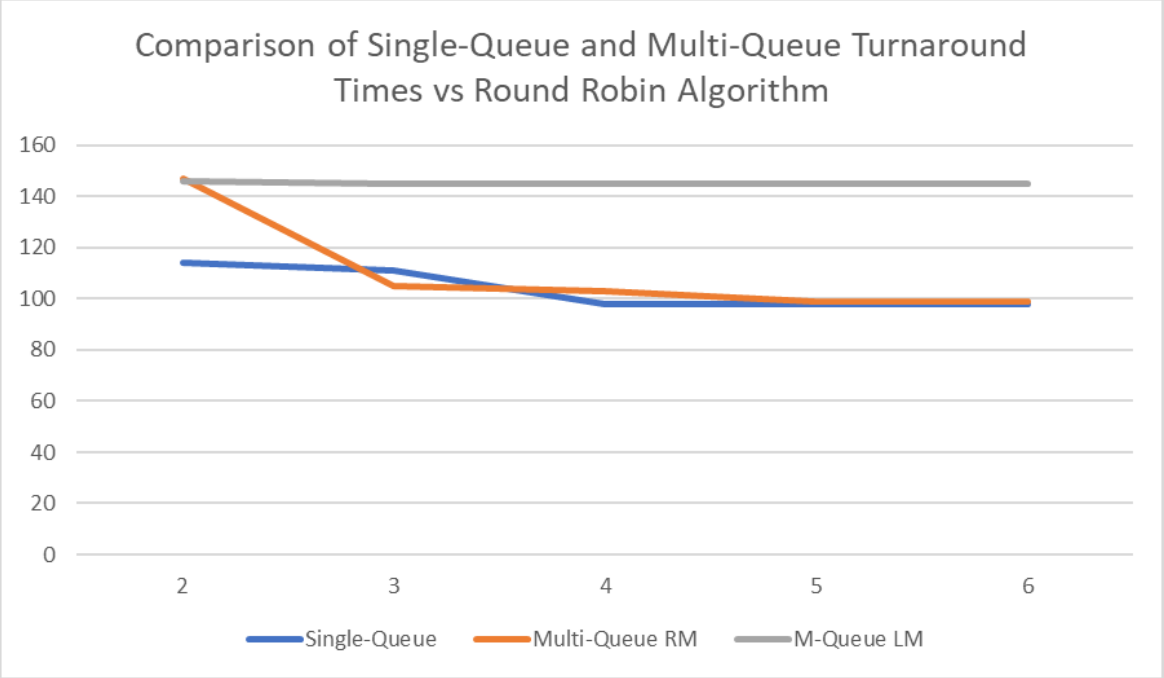
Comparison of Average Turnaround Times Of Single-Queue and Multi-Queue

The random method of the program is used to conduct the experiments on this part with the input of 150 10 300 100 10 250 30.

The bursts are created in the range of [10, 300] with mean of 150 ms and interarrival times are created in the range of [10, 250] with mean of 100 ms. 30 bursts are created with these values for the experiment.

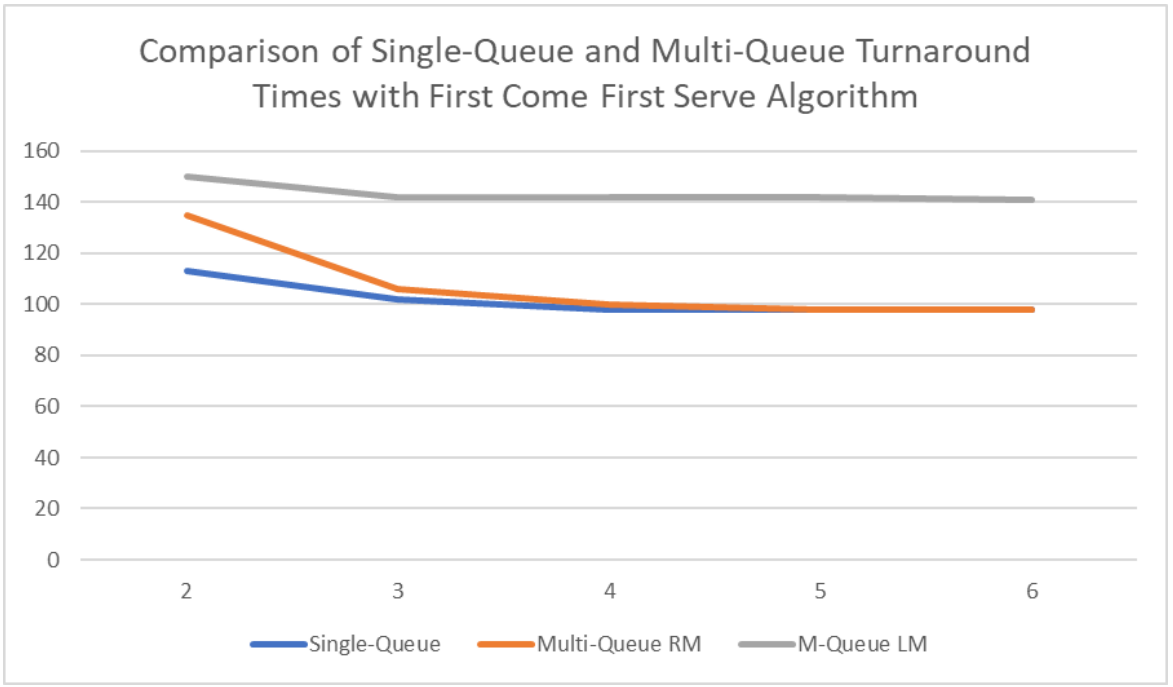
All experiments are repeated 5 times and the average of the results is used in order to have objective and unbiased results.

Single-Queue vs. Multi-Queue Using RR Q = 20



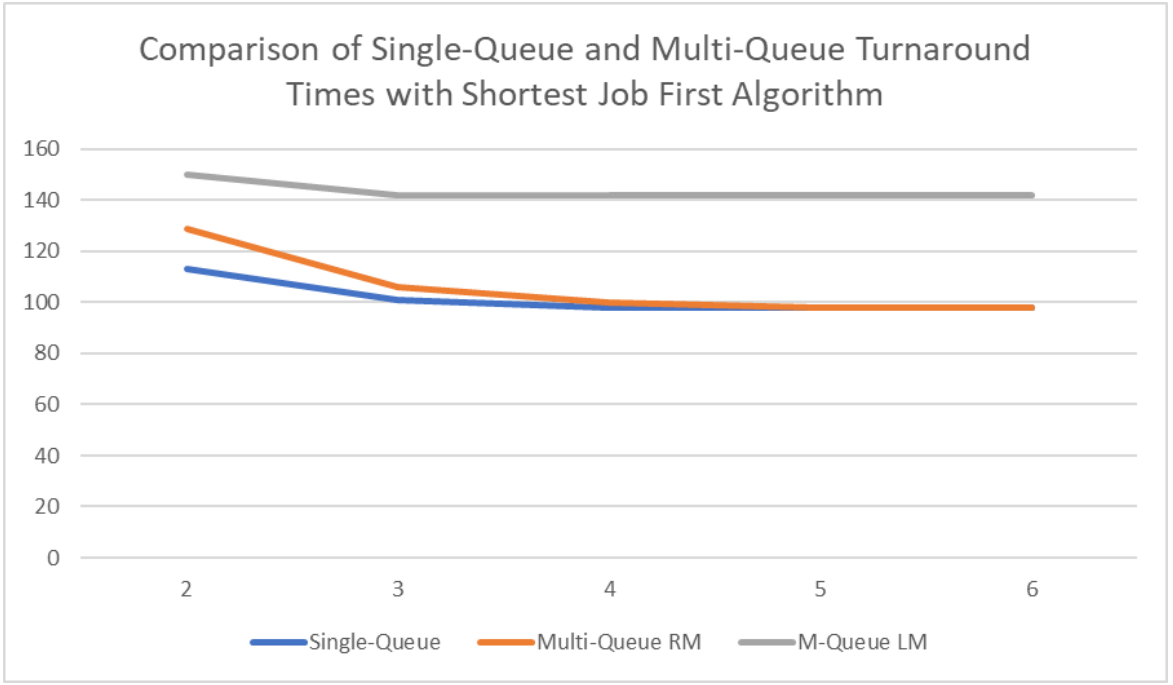
During this experiment 30 bursts are randomly created and added to a single queue or multiple queues (with RM or LM). Processors select and process the bursts using the Round Robin algorithm with 20 ms as Quantum. Looking at this graph we can observe that Single-Queue approach results in smaller turnaround times compared to RM multiqueue approach when the number of processors is smaller than 3. If there are 3 or more processors, RM multiqueue approach and single queue approach perform very similarly. However, we can clearly see that LM mutiqueue approach results in much higher average turnaround times compared to RM multiqueue approach when there are more than 2 processors.

Single-Queue vs. Multi-Queue Using FCFS



According to the graph, when the number of processors are 4 or smaller, the single queue approach performs significantly better in terms of average turnaround time, compared to LM multiqueue and RM multiqueue approaches. When there are more than 4 processors RM multiqueue approach and single queue approach perform very similarly. Also, even though the algorithm has changed, we can see that the LM multiqueue approach still performs worse compared to other approaches just as it did with the Round Robin Algorithm.

Single-Queue vs. Multi-Queue Using SJF



Similarly with RR and FCFS algorithms we can see that the single queue approach performs better with a smaller number of processors compared to multiqueue. If there are 4 or more processors, RM multiqueue approach and single queue approach perform almost the same. LM multi queue method still results in the highest turnaround times.

With all three algorithms we can see that a single queue results in lower average turnaround times when the number of processors is smaller than 3. This could be a result of the inaccuracies in mutiqueue selection methods. Also, having a single queue for a lower number of processors makes sure that no processor is empty at any time when there is more than one processor another processor needs to work on. Context switches in single queue locking mechanisms do not take as much time as the waiting time in multiqueue approaches.

However, if there are a bigger number of processors, we see that RM Multi-Queue approach performs similarly with single queue approach. This could be a result of less processes needing to wait in queues while some processors are empty, since the processor count is bigger. Because more processors ensure that there will be more idle processors that are not doing work at any moment.

The difference in average turnaround times between LM and RM Multi-Queue approaches could be explained as inaccuracy of the algorithm that chooses the queue with the least amount of load at the moment. The algorithm cannot consider any processes being processed at that moment as they are retrieved from the queue, and therefore can be quite inaccurate while finding the queue with the least load. For example, if there are two processors and the first has retrieved a burst of 100 ms from its own queue while the other one is idle, the algorithm will add the upcoming bursts into the first processes queue as both queues are empty and the id of the first processor is smaller.

Considering all these observations we can conclude that LM Multi-Queue is not efficient for this program as the implementation of the algorithm is not accurate enough. Also, if the number of processors is smaller than 3, Single-Queue is the option with the smallest average time. If there are more than 3 processors, Single-Queue and RM Multi-Queue approaches perform similarly.

Part B)

Comparison of Average Turnaround Times of Algorithms

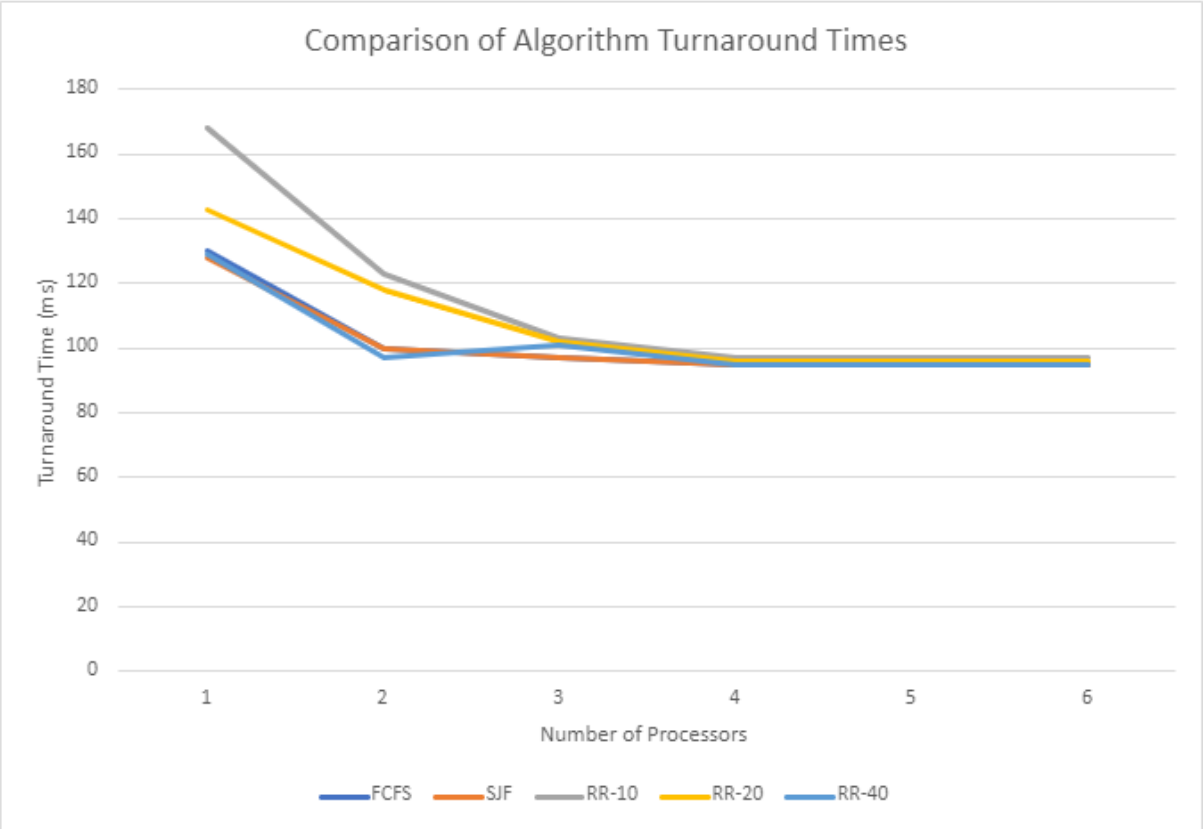
For this part, bursts and interarrival times are created randomly with the default values for ranges and 20 bursts are used for each experiment. Also, multi queue approach is used with round robin method as the scheduling approach.

Therefore, burst are created in the range [10,1000] with mean of 200 ms and inter arrival times are created in range [10,500] with mean of 100 ms.

In order to make a fair comparison, each experiment is repeated 5 times and the average of these values are used in the table below.

Most of the values are left as default so, an example of the program execution command is:
./mps -n 1 -r 200 10 1000 100 10 500 20

	n = 1	n = 2	n = 3	n = 4	n = 5	n = 6
FCFS	130	100	97	95	95	95
SJF	128	100	97	95	95	95
RR-10	168	123	103	97	97	97
RR-20	143	118	102	96	96	96
RR-40	129	97	101	95	95	95



As seen from the graph, the round robin algorithm with small quantum value performs worse compared to others when the number of processors is low. As the quantum value of the round robin increases it starts to act more like the other algorithms like FCFS and SJF. The reason for this is that the bursts are completed in less iterations and the start and end times get closer.

It is expected that the turnaround times for FCFS and SJF would be the same because they both are not preemptive and all bursts have turnaround times equal to their burst lengths. This behavior can be observed in the graph. The lines indicating the SJF and FCFS are almost identical.

All algorithms converge to a similar turnaround time after the number of processors exceeds 4. The reason for this is that the program reaches a point where queues do not overload by the bursts and therefore round robin acts the same as other algorithms. With different burst lengths and inter arrival times, this point could have been different.