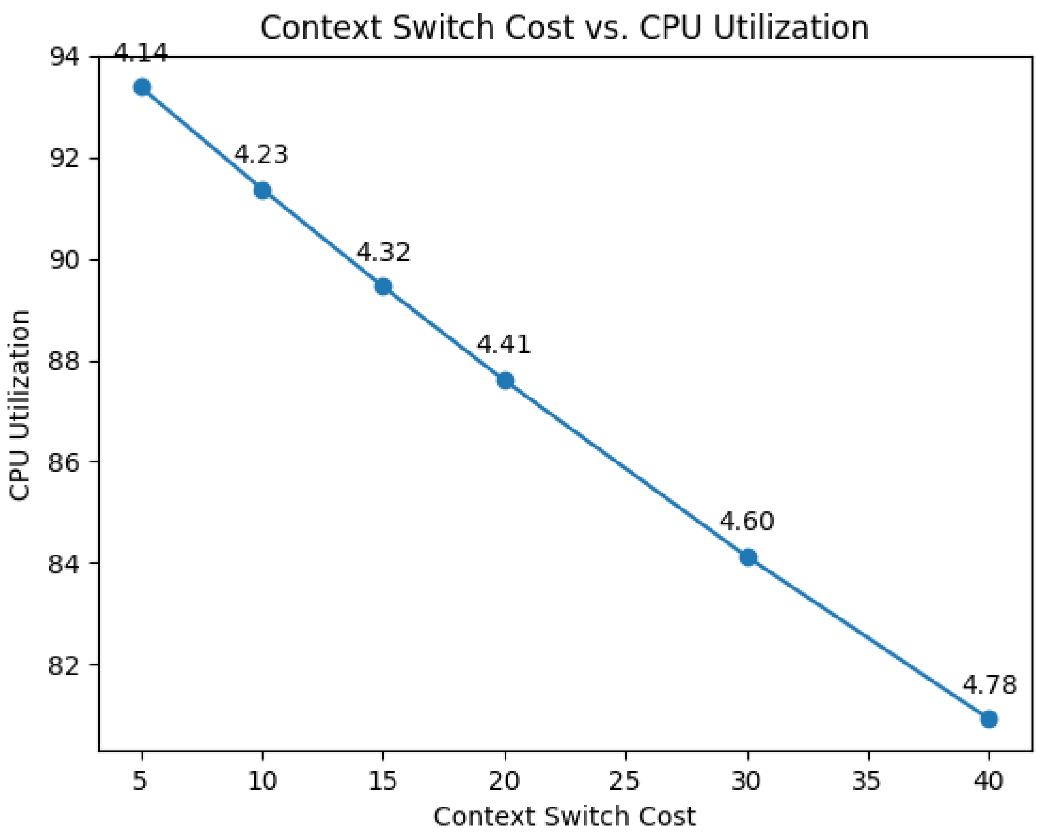
# Dean Kopping OS assignment report

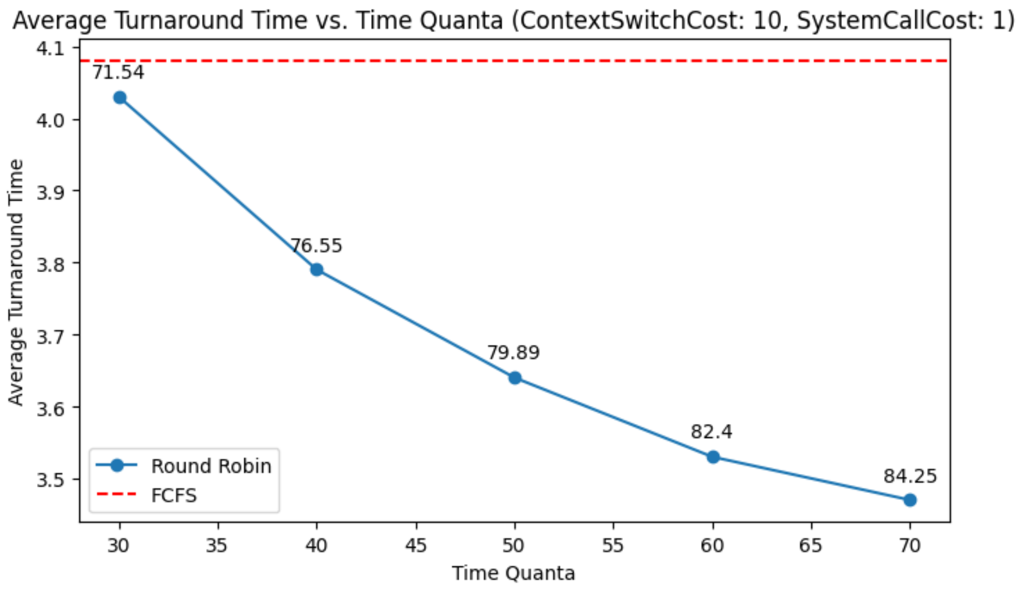


Graph 1

The above graph depicts the relationship between the context switching cost and CPU utilization for the FCFS algorithm. The graph also includes the average turnaround time for these processes at each given context switch cost.

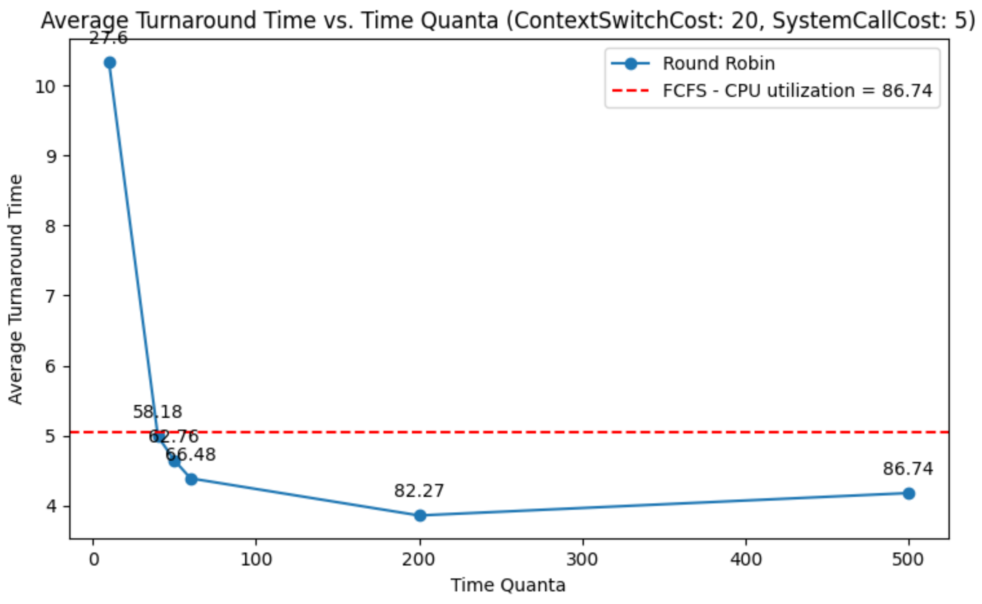
The cost of the context switch correlates directly with CPU utilization. The greater the context switch cost is, the lower the CPU utilization will be. In addition to this, the average turnaround time increases as the context switch cost increases.

These results are all expected and unsurprising because if the context switch takes more time, the CPU will spend more time idle, and the overall turnaround time has to increase.



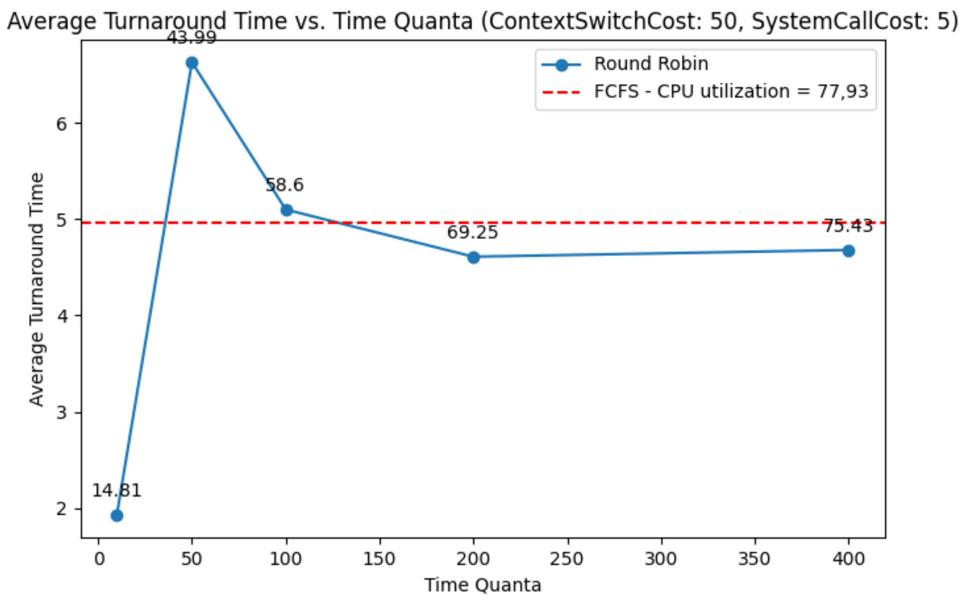
Graph 2

The above graph shows that for a context switch cost of 10 and a system call cost of 1, the round-robin scheduling algorithm performs better than FCFS for every time quanta. It shows that a larger time quantum will result in a higher CPU utilization as well as a lower average turnaround time.



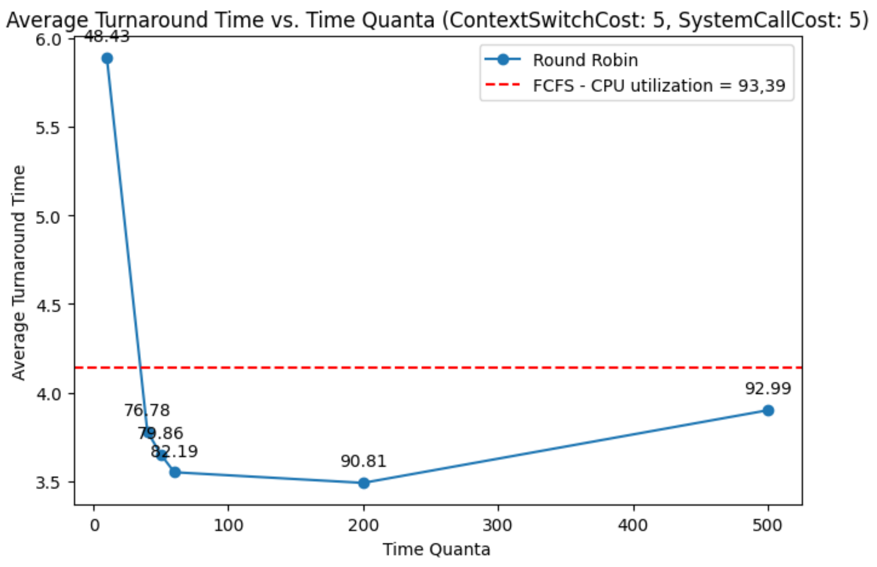
Graph 3

In this graph, we notice that when the context switch cost is increased to 20 and the system call cost is increased to 5, the round-robin algorithm is not always better than FCFS. With these costs, the FCFS algorithm performs well with a CPU utilization of 86,74 and an average turnaround time of just over 5. By contrast, the round-robin algorithm performs very poorly when the time quantum is very low (10). The round robin algorithm only begins to outperform FCFS when the time quantum is larger than 50 with regards to average turnaround time. However, the CPU utilization in the round-robin algorithm only begins to match the FCFS when the time quantum is around 500.



Graph 4

When given a very high context switch cost of 20, the round robin algorithm produces an extremely low average turnaround time, however, this is coupled with an extremely low CPU utilization and is therefore not ideal. We can notice that the round robin algorithm maintains a lower CPU utilization than FCFS across the whole range of time quanta but has a consistently lower average turnaround time when time quanta is lower than 50 and higher then around 140.



Graph 5

The results of Graph 5 are very similar to those of Graph 3. The difference in context switch cost between these 2 graphs is 15. This proves to not be a great enough difference to have a meaningful effect on the results.

Conclusion:

CPU utilization:

By referring to the data, CPU utilization is highest when context switch cost as well as system call cost is at its lowest. In this situation, the FCFS algorithm has a consistently high CPU utilization. The round robin CPU utilization increases as the time quantum increases.

Average turnaround time:

Graphs 3, 4, and 5 all indicate that the average turnaround time is optimized when using the round robin algorithm with a time quantum close to 200. It appears that after this point the average turnaround time begins to increase again, and the line tends towards the FCFS constant line.

When does RR become FCFS:

It is apparent that the round-robin algorithm will become the FCFS algorithm when the time quantum gets sufficiently large. The produced graphs show that the average turnaround time as well as the CPU utilization for the round robin algorithm both get closer to that of the FCFS algorithm when the quantum gets larger than 500.

Based on this, one can hypothesize that once the time quantum is roughly 800-1000, the round-robin algorithm will perform as FCFS.

Overall, we can conclude that the round-robin algorithm outperforms FCFS when the time quantum is optimal. If too small, processes will not be completed as regularly, and the average turnaround time will be high. If the time quantum is too large, then round robin will act as FCFS.