Ayden Burgoyne

Assignment 1 Brief:

COMP 2240

6/09/2019

Assignment 1 Brief:

Performance of FB

My program produced the same results as the supplied results for both datafiles. It makes sense that FB had the best performance in the second datafile and tied second best performance for the first. The reason for this is it deprioritises processes that go for a long period of time, meaning that every process gets their first run-through faster. This in turn reduces the average waiting time as the short processes get completed on their first run while longer processes are delayed. This however means that longer processes have their turnaround and waiting time significantly increased compared to first come first serve.

It's interesting to note that in the normal implementation of Feedback the algorithm is meant to pre-empt the currently running process if it is of a lower priority and a new process joins a higher priority ready queue this however isn't followed in the assignment. If it had been implemented it would've resulted in more time used up from the dispatcher and more waiting for processes with longer execute times.

Performance of NRR

It's interesting to note that NRR has the best performance on datafile 1 while on datafile 2 it has the worst. This is probably because not all the process arrives at once in the second datafile meaning that all the shorter process had queue behind each other at the end meaning P1 had to keep getting swapped out instead of running for an extended time slice with not processes left like happens in the first datafile. Essentially this also adds more dispatcher calls as P1's time slices get shorter due to being in the CPU for shorter periods of time.

Performance of FCFS

For first come first serve it's interesting to note that in the first datafile it has the worst performance while in the second it improves. This is probably because process one's executing didn't affect the results of the rest so much as they hadn't arrived when it was executing. However, in the first datafile they had all arrived at the same time, meaning that they were waiting in the ready queue for longer.

Performance of RR

Round Robin performance is interesting in that it achieves the same results as Feedback when the arrival times are the same however performs worse when the processes are split up. This is because when the arrival times are split up, the priority of the bigger execution time process is already lower than the newly arrived short execution time process. Meaning it gets to skip in front essentially. However, when they arrive at the same time in datafile 1 the smaller execution time processes still must wait for p1 to execute. (P1 executing first is because of the dispatch rules for breaking ties).