Bilkent University

Computer Engineering

CS342

Project - 02

Mohammed S. Yaseen
21801331

28/03/2020

Introduction:

The experiment summarizes the main differences between multiple thread scheduling algorithms with multiple parameters, including the number of threads, the number of bursts per thread, the inter-arrival time of bursts, as well as the execution length of a burst.

Comparing multiple scheduling algorithms:

Here multiple scheduling algorithms will be compared while using the same or varying number of threads as well as the same number of bursts per thread, interarrival times, and burst lengths.

VRUNTIME Algorithm

In this case the total execution time will be fixed for all threads to 1500ms, and the interarrival time will be 200 ms for the first burst with 2 ms increment for the first arrival for each thread, so that i'th thread will start at 200+2*i ms. For each consecutive burst the interarrival time will be 200/i. Two trials will be done. For the first trial the number of bursts per thread will be the same for all threads; however for the second trial the number of bursts will be varying as in Table 1. The results are graphed in Figure 1:

Thread No.	Burst count	Burst length	Total execution length
1	5	300 ms	1500 ms
2	10	150 ms	1500 ms
3	15	100 ms	1500 ms
4	20	75 ms	1500 ms
5	25	60 ms	1500 ms

Table 1: The number of bursts as well as their lengths per thread.

Figure 1: Thread waiting time per trial for VRUNTIME algorithm.

It can be observed from the figure that the total waiting time per thread is very similar in both trials. On the other hand, we notice that the time taken by each thread is increasing from the 1st thread to the 5th one. These are due to the nature of the algorithm, where it reduces the chance of having a thread execute when its total execution time increases.

FCFS Algorithm

The same setup was used for the FCFS algorithm but the results were different. The results are illustrated in figure 2.

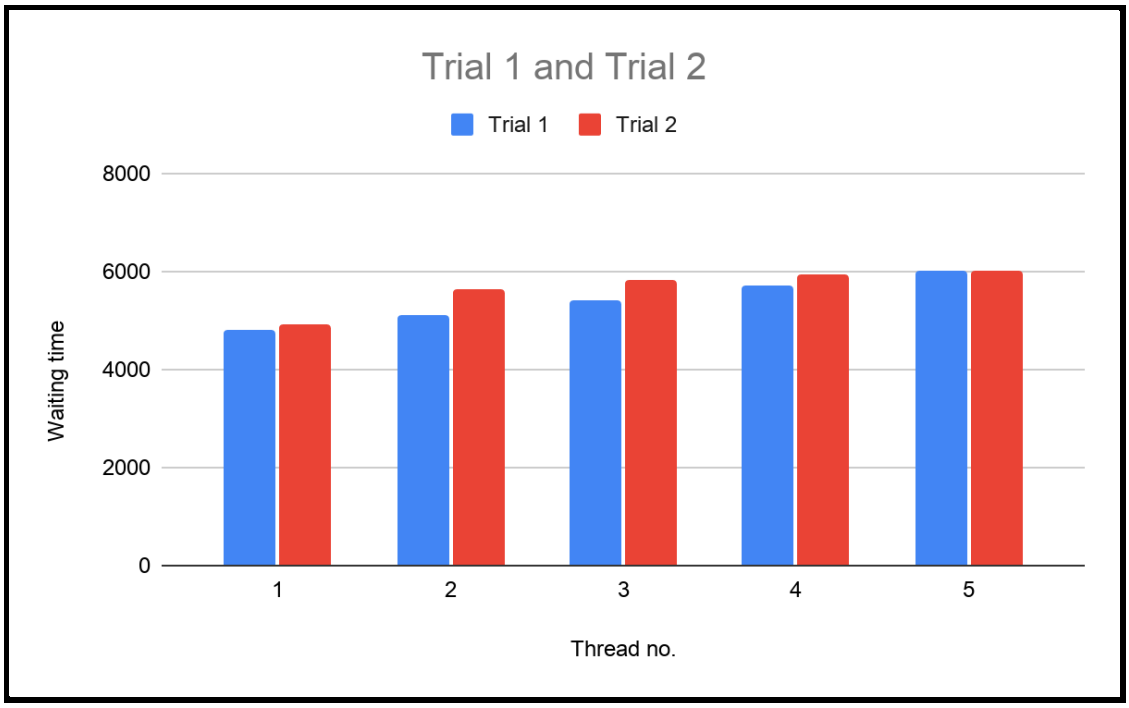

Figure 2: Thread waiting time per trial for FCFS algorithm.

Again, just like for VRUNTIME, there is no significant difference between the execution time of the threads.

PRIO Algorithm

Once more the same setup as for VRUNTIME is used with inversing the starting time of each thread that is a thread would start at 200+2*(THREAD_COUNT - i). The results are graphed in figure 3.

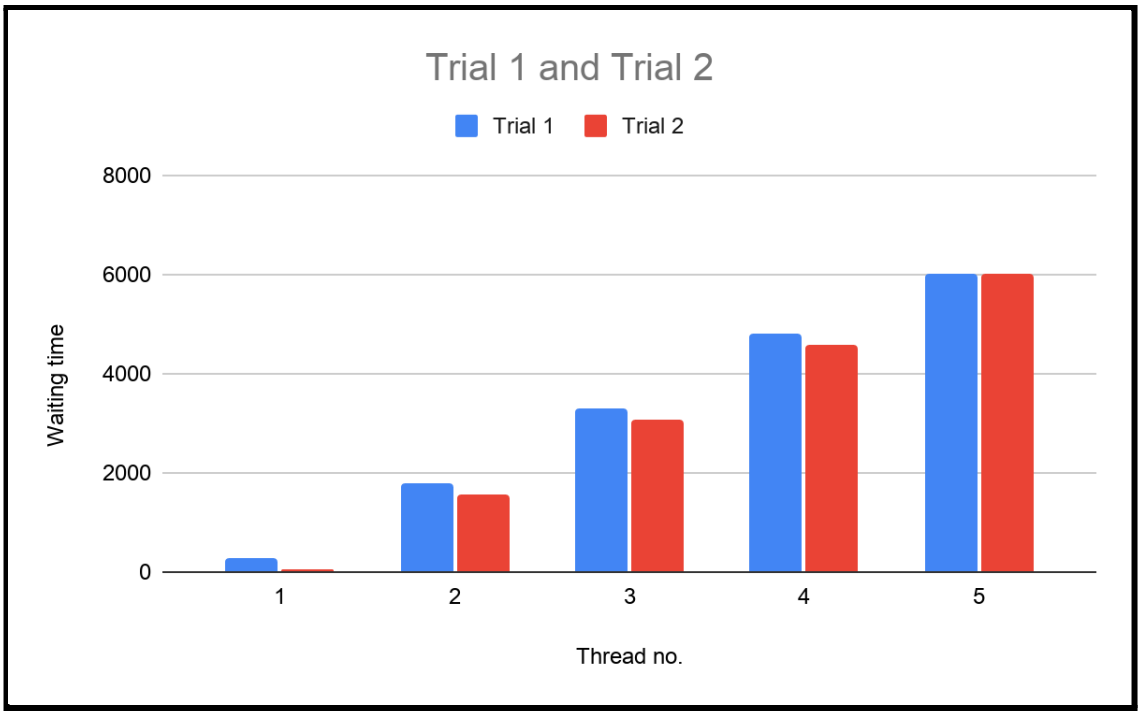

Figure 3: Thread waiting time per trial for PRIO algorithm.

Here, although the starting time of the first algorithm is the latest, it still finishes execution with a very small amount of waiting time. The waiting time increases as the thread number increases. This is due to how the algorithm works, where the thread with the smaller number has a higher priority so that once it is in the queue it runs directly.

SJF Algorithm

For this algorithm the setup for the first trial is the same as the VRUNTIME algorithm; however, for the second trial another setup is to be used. Here the burst length of the algorithms will vary. The burst lengths per thread will be as table 2. The results are graphed in figure 4.

Thread No.	Burst count	Burst lengths	Total execution length
1	5	700 ms	1500 ms
2	5	600 ms	1500 ms
3	5	500 ms	1500 ms
4	5	400 ms	1500 ms
5	5	300 ms	1500 ms

Table 1: The number of bursts as well as their lengths per thread.

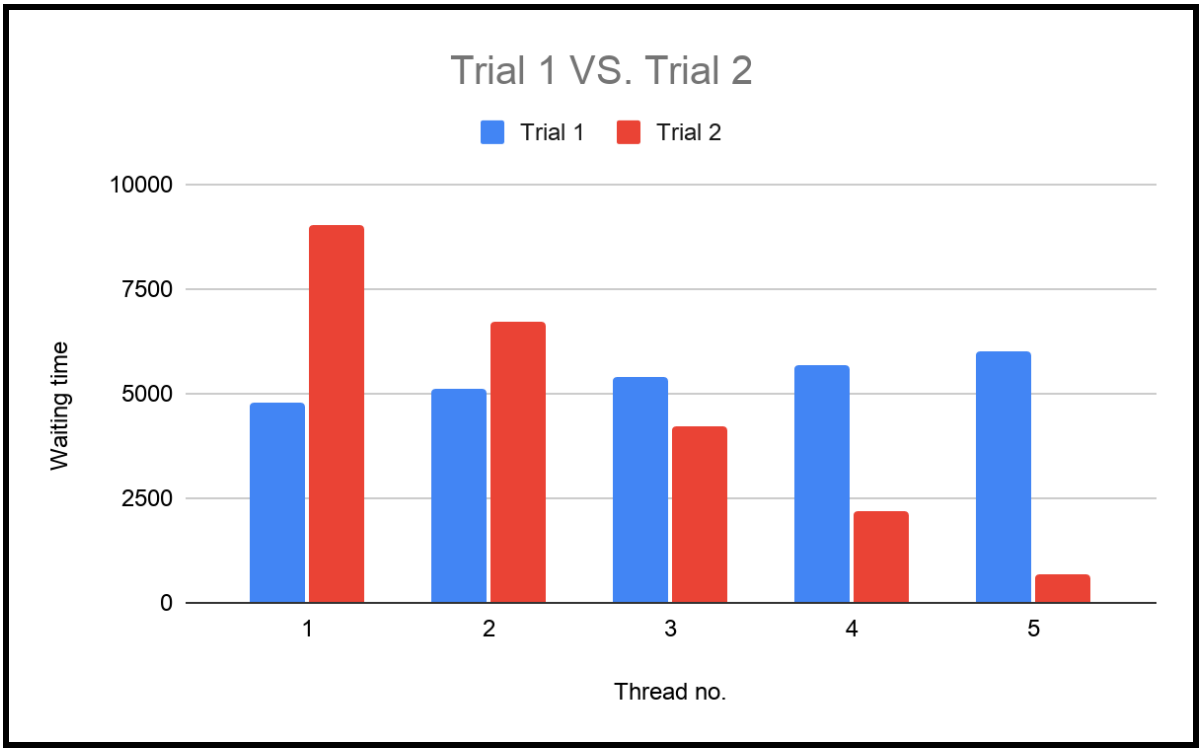

Figure 4: Thread waiting time per trial for PRIO algorithm.

Since we set up the second trial to have smaller bursts for higher thread numbers, we notice that in the waiting time decreases as the number of threads increases (the burst length decreases).

Conclusion

It can be noticed that each scheduling algorithm has a different behavior. Comparing FCFS with VRUNTIME we notice that although they used the exact same data for thread interarrival time and burst lengths, the VRUNTIME allowed the thread that was scheduled earlier the first time to wait less in the run queue. On the other hand, looking at the PRIO algorithm, we notice that no matter what order the threads arrive with, the thread with higher priority will always be scheduled earlier. Finally, the SJF algorithm allows threads to execute earlier if they require less time to execute. This, in fact, can cause processes with long bursts to wait much longer.