İhsan Doğramacı Bilkent University

Department of Computer Engineering

CS342 - Operating Systems

Project 2

Section 3

- Bedirhan Sakinoğlu - 21802576
- Gökhan Taş - 21802136

Instructor: Özcan Öztürk

Table of Contents

Introduction

The fundamental goal of the project is to make a simulation where we create multiple threads and run them with different scheduling algorithms so that we can see difference between each algorithm. In order to achieve that, there should not be deadlock or race condition in our simulation. In this simulation we use FCFS, SJF, and RR algorithms with different burst decider algorithms that are fixed, exponential and uniform. Moreover, we simulate the stages of a thread which are Waiting, Ready and Running. In our simulation we have simple I/O structure to simulate waiting input situation and we have a ready queue so that we can compute threads one by one according to queue. The expected result was to assess difference between given scheduling algorithms.

Computer Specifications

- OS: Windows 10
- Processor: Intel® Core™ i7-9750H CPU @ 2.69GHz
- GPU: NVIDIA GTX 1660 Ti (Mobile)
- RAM: 16.0 GB

We used VirtualBox to use Ubuntu 20.04.3 with specifications as follows:

- RAM: 10 GB
- Core Count: 4
- VRAM: 128 MB

Graphs

In order to assess performance of scheduling, we calculated finish time, waiting time and turnaround time of first 30 threads that were created in the simulation. We used given terminal codes such as “`./systemsim` **`Y Z`** `50 100 fixed 20 10 100 0.1 0.6 0.7`

0.5 30 **X** 1” terminal code where Y decides scheduling algorithm, Z defines quantum time and X defines number of threads that are created. However, we calculated times in a wrong way and thus our table is not correct.

./systemsim **Y** **Z** 50 100 fixed 20 10 100 0.1 0.6 0.7 0.5 30 **X** 1			
Number of Extra Threads (X)	FCFS (Y) & (Z = INF)	SJF (Y) & (Z = INF)	RR (Y) & (Z = 10)
30	-	-	-
100	-	-	-
200	-	-	-

Figure 1 - Test cases for fixed distribution

./systemsim **Y** **Z** 50 100 exponential 20 10 100 0.1 0.6 0.7 0.5 30 **X** 1			
Number of Extra Threads (X)	FCFS (Y) & (Z = INF)	SJF (Y) & (Z = INF)	RR (Y) & (Z = 10)
30	-	-	-
100	-	-	-
200	-	-	-

Figure 2 - Test cases for exponential distribution

Conclusion

To sum up, since we sometimes have a race condition problem program will not proceed as it is wanted. However, most of time it finishes and all threads are scheduled as expected. On the other hand, since we measure the waiting time wrong, we get wrong finish time and turnaround time.