## İhsan Doğramacı Bilkent University



CS342 - Operating Systems

**Project 2**

### Section 3

### Bedirhan Sakinoğlu - 21802576

### Gökhan Taş - 21802136

**Instructor:** Özcan Öztürk

**Table of Contents**

[Introduction 2](#_TOC_250003)

[Computer Specifications 2](#_TOC_250002)

[Graphs 2](#_TOC_250001)

[Conclusion 3](#_TOC_250000)

# 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 the difference in performance, we created 5 different thread counts and made 5 different runs for each case. During these test cases, we computed results using **time** command and final results can be seen below.

|  |  |  |  |
| --- | --- | --- | --- |
| ./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) |
| 20 | - | - | - |
| 50 | - | - | - |
| 100 | - | - | - |
| 150 | - | - | - |
| 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) |
| 20 | - | - | - |
| 50 | - | - | - |
| 100 | - | - | - |
| 150 | - | - | - |
| 200 | - | - | - |

Figure 2 - Test cases for exponential distribution

|  |  |  |  |
| --- | --- | --- | --- |
| ./systemsim **Y** **Z** 50 100 uniform 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) |
| 20 | - | - | - |
| 50 | - | - | - |
| 100 | - | - | - |
| 150 | - | - | - |
| 200 | - | - | - |

Figure 3 - Test cases for uniform distribution

# Conclusion

To sum up, both threads and processes used message queues to transmit the data, and threads have shorter execution time. It can be said that threads are much faster than processes.