19th May 2021

EOPSY LAB 3

Krzysztof Piotrowski

Introduction

The main goal of this laboratory is to get familiar with the scheduling algorithm using MOSS Scheduling Simulator. As default program use First come, First Serve algorithm which automatically executes queied request and processes them by order of arrival[. We run 2/5/10 processes each with average execution of 2000 ms and standard deviation of zero. Each is blocked for input/output every 500 ms. Overall time of simulation is set to 10000 ms.

First example: 2 Processes

For 2 Processes I received following results:

```
Scheduling Type: Batch (Nonpreemptive)
Scheduling Name: First-Come First-Served
Simulation Run Time: 4000
Mean: 2000
Standard Deviation: 0
Process #       CPU Time        IO Blocking     CPU Completed       CPU Blocked
0               2000 (ms)       500 (ms)        2000 (ms)           3 times
1               2000 (ms)       500 (ms)        2000 (ms)           3 times
```

```
Process: 0 registered... (2000 500 0)
Process: 0 I/O blocked... (2000 500 500)
Process: 1 registered... (2000 500 0)
Process: 1 I/O blocked... (2000 500 500)
Process: 0 registered... (2000 500 500)
Process: 0 I/O blocked... (2000 500 1000)
Process: 1 registered... (2000 500 500)
Process: 1 I/O blocked... (2000 500 1000)
Process: 0 registered... (2000 500 1000)
Process: 0 I/O blocked... (2000 500 1500)
Process: 1 registered... (2000 500 1000)
Process: 1 I/O blocked... (2000 500 1500)
Process: 0 registered... (2000 500 1500)
Process: 0 completed... (2000 500 2000)
Process: 1 registered... (2000 500 1500)
Process: 1 completed... (2000 500 2000)
```

Conclusions:

The simulation lasted for 4000 ms – 2000 ms for each process. Each process has been executed successfully.

Pattern for single process is as follow: when process is registered it is running for 500 ms (due to our assumptions). After that time its I/O are blocked and we are able to register another process. This pattern is repeating for all processes until they reach their time of execution (2000 ms). After that time I/O are not blocked and processes are terminated.

Both processes ran in four 500 ms time intervals and each was blocked 3 times because last time interval did not block but terminated process.

Second example: 5 Processes

For 5 Processes I received following results:

```
Scheduling Type: Batch (Nonpreemptive)
Scheduling Name: First-Come First-Served
Simulation Run Time: 10000
Mean: 2000
Standard Deviation: 0
Process #     CPU Time      IO Blocking   CPU Completed      CPU Blocked
0             2000 (ms)     500 (ms)      2000 (ms)          3 times
1             2000 (ms)     500 (ms)      2000 (ms)          3 times
2             2000 (ms)     500 (ms)      2000 (ms)          3 times
3             2000 (ms)     500 (ms)      2000 (ms)          3 times
4             2000 (ms)     500 (ms)      2000 (ms)          3 times
```

```
Process: 0 registered... (2000 500 0 0)
Process: 0 I/O blocked... (2000 500 500 500)
Process: 1 registered... (2000 500 0 0)
Process: 1 I/O blocked... (2000 500 500 500)
Process: 0 registered... (2000 500 500 500)
Process: 0 I/O blocked... (2000 500 1000 1000)
Process: 1 registered... (2000 500 500 500)
Process: 1 I/O blocked... (2000 500 1000 1000)
Process: 0 registered... (2000 500 1000 1000)
Process: 0 I/O blocked... (2000 500 1500 1500)
Process: 1 registered... (2000 500 1000 1000)
Process: 1 I/O blocked... (2000 500 1500 1500)
Process: 0 registered... (2000 500 1500 1500)
Process: 0 completed... (2000 500 2000 2000)
Process: 1 registered... (2000 500 1500 1500)
Process: 1 completed... (2000 500 2000 2000)
Process: 2 registered... (2000 500 0 0)
Process: 2 I/O blocked... (2000 500 500 500)
Process: 3 registered... (2000 500 0 0)
```

```
Process: 3 I/O blocked... (2000 500 500)
Process: 2 registered... (2000 500 500)
Process: 2 I/O blocked... (2000 500 1000)
Process: 3 registered... (2000 500 500)
Process: 3 I/O blocked... (2000 500 1000)
Process: 2 registered... (2000 500 1000)
Process: 2 I/O blocked... (2000 500 1500)
Process: 3 registered... (2000 500 1000)
Process: 3 I/O blocked... (2000 500 1500)
Process: 2 registered... (2000 500 1500)
Process: 2 completed... (2000 500 2000)
Process: 3 registered... (2000 500 1500)
Process: 3 completed... (2000 500 2000)
Process: 4 registered... (2000 500 0)
Process: 4 I/O blocked... (2000 500 500)
Process: 4 registered... (2000 500 500)
Process: 4 I/O blocked... (2000 500 1000)
Process: 4 registered... (2000 500 1000)
Process: 4 I/O blocked... (2000 500 1500)
Process: 4 registered... (2000 500 1500)
```

Conclusions:

This time the simulation used full given time of 10000 ms – 2000 ms for each process.

What is worth mentioning in this case that after blocking process 1 we did not call process 2 but process 0. It is due to the fact that process 0 is before process 2 in queue. So we will interchange them until process 1 and 2 will be completed. After that process 2 and process 3 occured and when they had finished last process (4) was called.

What is also worth mentioning process 4 did not completed, it was blocked 3 times but we did not get last time interval that finish it.

Last example: 10 Processes

For 10 Processes I received following results:

```
Scheduling Type: Batch (Nonpreemptive)
Scheduling Name: First-Come First-Served
Simulation Run Time: 10000
Mean: 2000
Standard Deviation: 0
Process #     CPU Time      IO Blocking   CPU Completed      CPU Blocked
0             2000 (ms)     500 (ms)      2000 (ms)          3 times
1             2000 (ms)     500 (ms)      2000 (ms)          3 times
2             2000 (ms)     500 (ms)      2000 (ms)          3 times
3             2000 (ms)     500 (ms)      2000 (ms)          3 times
4             2000 (ms)     500 (ms)      1000 (ms)          2 times
5             2000 (ms)     500 (ms)      1000 (ms)          1 times
6             2000 (ms)     500 (ms)      0 (ms)             0 times
7             2000 (ms)     500 (ms)      0 (ms)             0 times
8             2000 (ms)     500 (ms)      0 (ms)             0 times
9             2000 (ms)     500 (ms)      0 (ms)             0 times
```

```
Process: 0 registered... (2000 500 0 0)
Process: 0 I/O blocked... (2000 500 500 500)
Process: 1 registered... (2000 500 0 0)
Process: 1 I/O blocked... (2000 500 500 500)
Process: 0 registered... (2000 500 500 500)
Process: 0 I/O blocked... (2000 500 1000 1000)
Process: 1 registered... (2000 500 500 500)
Process: 1 I/O blocked... (2000 500 1000 1000)
Process: 0 registered... (2000 500 1000 1000)
Process: 0 I/O blocked... (2000 500 1500 1500)
Process: 1 registered... (2000 500 1000 1000)
Process: 1 I/O blocked... (2000 500 1500 1500)
Process: 0 registered... (2000 500 1500 1500)
Process: 0 completed... (2000 500 2000 2000)
Process: 1 registered... (2000 500 1500 1500)
Process: 1 completed... (2000 500 2000 2000)
Process: 2 registered... (2000 500 0 0)
Process: 2 I/O blocked... (2000 500 500 500)
Process: 3 registered... (2000 500 0 0)
Process: 3 I/O blocked... (2000 500 500 500)
Process: 2 registered... (2000 500 500 500)
Process: 2 I/O blocked... (2000 500 1000 1000)
Process: 3 registered... (2000 500 500 500)
Process: 3 I/O blocked... (2000 500 1000 1000)
Process: 2 registered... (2000 500 1000 1000)
Process: 2 I/O blocked... (2000 500 1500 1500)
Process: 3 registered... (2000 500 1000 1000)
Process: 3 I/O blocked... (2000 500 1500 1500)
Process: 2 registered... (2000 500 1500 1500)
Process: 2 completed... (2000 500 2000 2000)
Process: 3 registered... (2000 500 1500 1500)
Process: 3 completed... (2000 500 2000 2000)
```

```
Process: 4 registered... (2000 500 0 0)
Process: 4 I/O blocked... (2000 500 500 500)
Process: 5 registered... (2000 500 0 0)
Process: 5 I/O blocked... (2000 500 500 500)
Process: 4 registered... (2000 500 500 500)
Process: 4 I/O blocked... (2000 500 1000 1000)
Process: 5 registered... (2000 500 500 500)
```

Conclusions:

In this example not all processes were executed. The reason for that is we planned to run each for 2000 ms and we have simulation time set only for 10000ms. Only 4 of them executed and finished properly.

Also, processes 4 and 5 started but they were unable to finish due to lack of time. Process was blocked twice which is one less that in previous example. It is beacause last time process 4 was executing alone while this time we had additional registration and blockade of process 5.

Process 6 and higher was not even called single time.