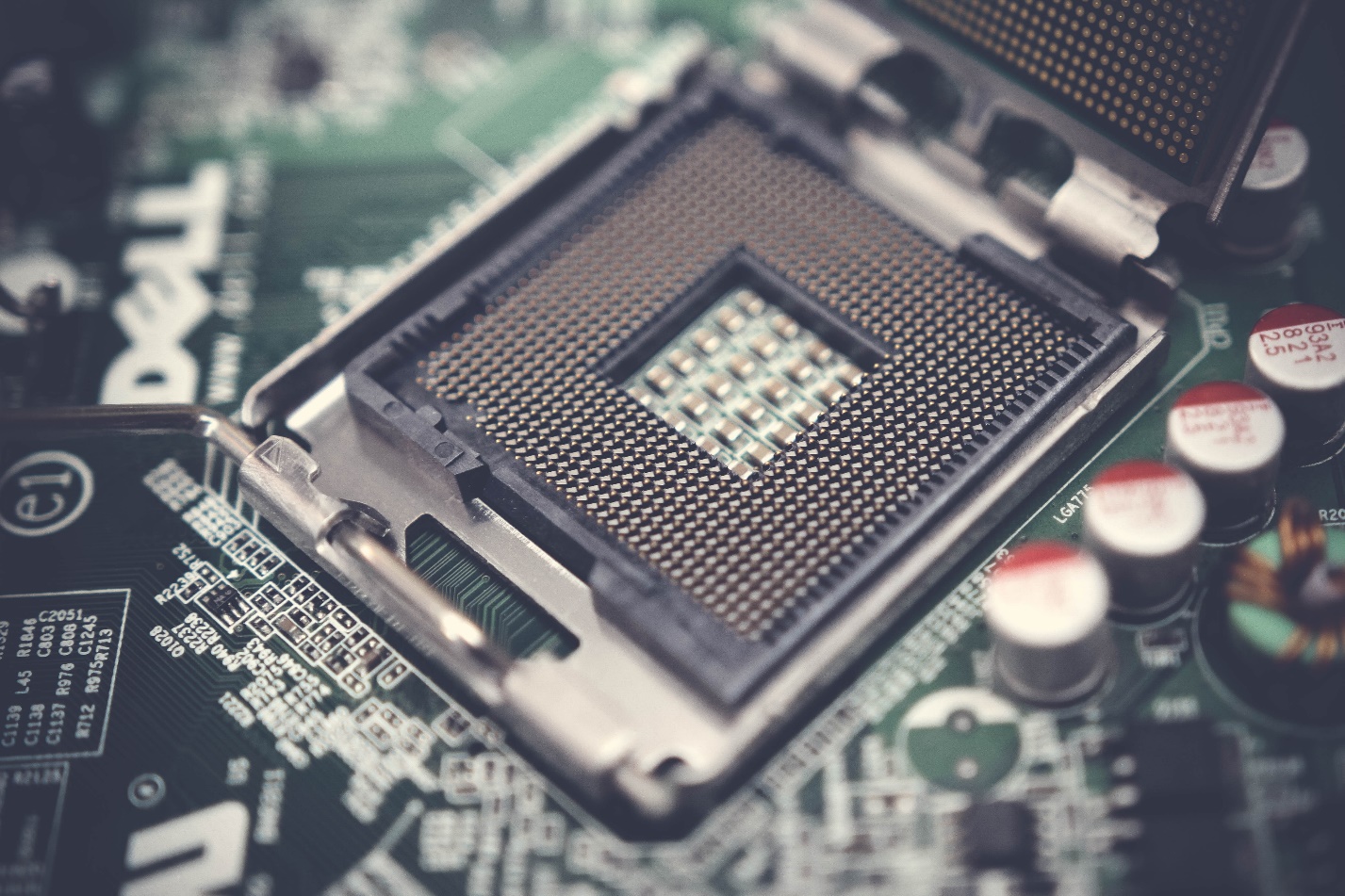
Fall 2018

Joshua Croft & Vildan Hakanaj

Trent University

Fall 2018

COIS 3320 Lab 2 Analysis Report



**Abstract:**

This report compares and contrasts the results and output of the program written in Lab #2 known as the ‘Process Simulator’. The Program was written to simulate various CPU scheduling algorithms to determine the relative merits of each.

**Introduction:**

In the discussion of CPU scheduling algorithms, it is sometimes unclear as to which algorithms are the most efficient. It is also unclear as to whether there is any one algorithm that can be considered the optimal choice in all situations and conditions. The aim of this report is; to simulate the following algorithms: First Come First Serve, Shortest Job First, Shortest Job First with Pre-emption and Round Robin (using two different time quantum values); to test each of the four outlined algorithms against certain criteria, and to determine which conditions favour which algorithm in regards to performance.

**Technologies used:**

* The program was written in the java programming language.
* Intellij IDE was used exclusively.
* Github was used for version control.

**Methods:**

The four algorithms are tested against three different types of “Job sets”. The job sets are defined as collections of jobs that are created under the same possible conditions, restrictions and scope. A Job is defined as a process to be run within the CPU and to be controlled and manipulated by the scheduling algorithm.

A Job contains an initial arrival time for entry into the waiting queue and a job length representing the amount of time remaining for processing. The properties of each job are generated using gaussian distributions. All three job sets assign a random arrival time upon creation of the job using a gaussian distribution with mean = 160 and standard deviation = 15. The job sets differ in their assignment of the job length property.

For the sake of simplicity, abstract integer “time units” will be used to represent time.

Job length distribution parameters for Job set #1

Mean = 150-time units

Standard Deviation = 20-time units

Job set #2 and Job set #3 is made up of two “types” of Jobs: large and small.

* Small Jobs
  + Mean = 50-time units
  + Standard Deviation = 5-time units
* Large Jobs
  + Mean = 250-time units
  + Standard Deviation = 15-time units

Job length distribution parameters for Job set #2

* 20% chance of a Job to considered “large Job”.
* 80% chance of a job to be considered “small Job”.

Job length distribution parameters for Job set #3

80% chance of a Job to considered “large Job”.

20% chance of a job to be considered “small Job”.

The five algorithms to run and a brief description of each are as follows:

First come first serve.

Jobs process on a first-in first-out basis dependant upon arrival time into the waiting queue. Once a job begins processing, it will run to completion (i.e. no other Job may run until this Job finishes processing).

Shortest Job First.

Jobs process depending on whichever currently waiting Job has the shortest Job Length property (shortest time to process). This excludes Jobs that have not yet arrived. Once a job begins processing, it will run to completion (i.e. no other Job may run until this Job finishes processing).

Shortest Job First with Pre-emption

Jobs process depending on whichever currently waiting Job has the shortest Job Length property (shortest time to process). This excludes Jobs that have not yet arrived. Once a job begins processing, it will run for a set interval of 40 time units or until completion if less time is required. At this stage, the shortest Job in the waiting queue will be the next to process.

Round Robin with a time quantum = 50 time units

Jobs process initially on a first-in first-out basis dependant upon arrival time into the waiting queue. Once a job begins processing, it will run for an interval of 50 time units or until completion if less time is required. If a job has not completed at this stage, it will move to the back of the waiting queue, behind any Jobs which may have arrived while the current Job was processing.

Round Robin with a time quantum = 75 time units

Jobs process initially on a first-in first-out basis dependant upon arrival time into the waiting queue. Once a job begins processing, it will run for an interval of 75 time units or until completion if less time is required. If a job has not completed at this stage, it will move to the back of the waiting queue, behind any Jobs which may have arrived while the current Job was processing.

No two Jobs run simultaneously. That is, the scheduler can only run a maximum of 1 Job at any given instance regardless of whichever algorithm is being used.

Each algorithm will output three key pieces of information: average turn around time, average response time, and the number of context switches that occurred.

Turn around time

The time it takes a Job to finish processing after arriving in the waiting queue.

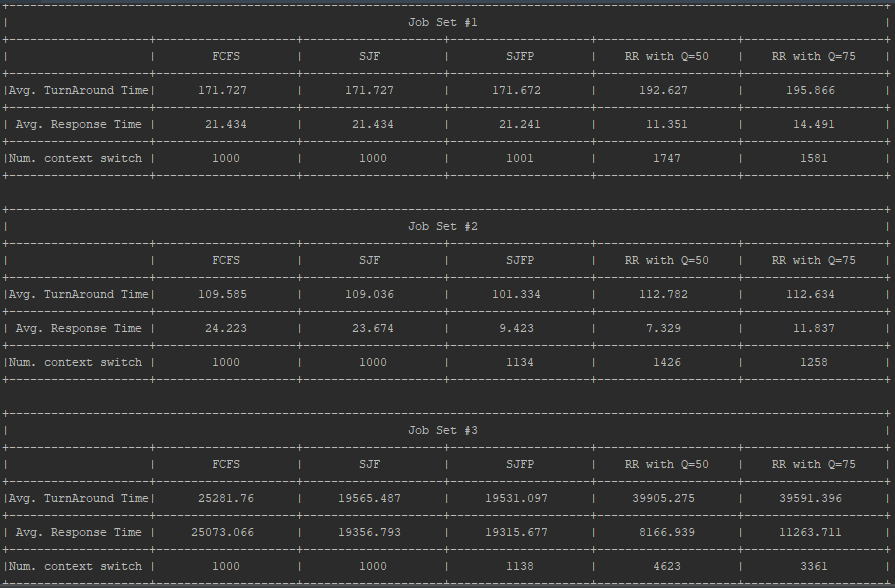
Response time

The time it takes a job to initially start processing after arriving in the waiting queue.

Context Switch

Occurs when a Job is loaded into the processor. (A new job begins processing)

**Results:**

The following output after running the application for 1000 trials is displayed in the following table

(Fig 1.1)

**Analysis:**

Job set #1

**FCFS and SJF**

The first thing that is evident when observing the first Job set is that both the First Come First Serve algorithm and the Shortest Job First algorithm share the exact same output. They both have the exact same number context switches as there are Jobs in the set. This is expected, both algorithms run each Job to completion once processing has begun (i.e. No context switching besides initially loading each Job).

**SJFP**

In the shortest job first with pre-emption we had two cases with the context switch.

1. When there is more context switches than there are jobs.
2. When the number of context switches is the same as the number of jobs.

The second case seemed to occur more often and the cause of that is the distribution on the arrival time of each job is to far apart, so we rarely have any context switches.

In the rare case that the distribution allows two jobs to be in the ready queue at the same time, the newest job would have to have a smaller job length than the job that was already running. This is unlikely to happened, not only because of the jobs being created under the same condition because the job that ran first has already had its time decreased due previous execution.

Regarding the turnaround time and response time we can see that is really similar to FCFS and SJF because of the context switching are similar and the jobs almost run SJF manner.

**Round Robin (RR)**

Round robin turnaround time in both cases with different quantum time we can see that its slightly higher than the previous algorithms. Its response time is reduced almost in half compared with the previous three algorithms.

The Round Robin with a smaller quantum time has a slightly better response time than the Round Robin with a bigger quantum time.

Job Set #2

**FCFS and SJF**

Similar to the results seen in the job set #1, we can observe that the output for these two algorithms are nearly identical. The difference between the Job set #2 and #1 is that the SJF has a slightly better response time and turnaround time than FCFS. This is due to the nature of job set #2 having a greater probability of having small jobs than large jobs. The reason of the results being better is that the jobs have a smaller chance to get stuck behind a large job.

**Conclusion:**