

CS322:Big Data

Final Class Project Report

**Project (FPL Analytics / YACS coding): YACS Coding Date: December 1,2020**

|  |  |  |  |
| --- | --- | --- | --- |
| SNo | Name | SRN | Class/Section |
| 1 | Shaashwat Jain | PES1201802346 | 5-I |
| 2 | Srishti Sachan | PES1201802126 | 5-B |
| 3 | Rajdeep Sengupta | PES1201800144 | 5-C |
| 4 | Raghav Aggarwal | PES1201800312 | 5-C |

## Introduction

The core concept of Big data is to process a large amount of data over a distributed file system. Our project “centralized scheduling framework” uses a scheduling framework which manages jobs between the master and multiple workers. These jobs are too big to execute on a single node so the master partitions these jobs into several map-reduce tasks which can run on multiple machines/nodes simultaneously. Each task is assigned to a worker on the basis of a scheduling algorithm. Presented in this report is a description of our project implementation.

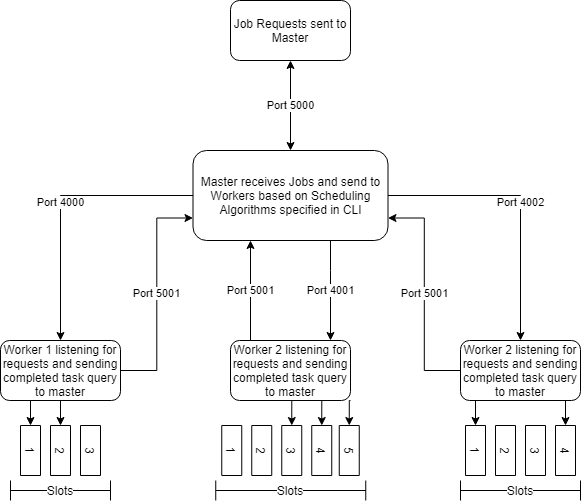
## Related work

1. **Socket programming** is used for all sorts of communication between the master, worker and requests to schedule the jobs and for checking the availability of the slots.
2. **Semaphore programming** is implemented to avoid deadlocks and race conditions for shared resources.
3. **Threads** were used to receive/send tasks and simultaneously decide the scheduling algorithms for workers. Also, threads were used in workers to simulate each worker slot.
4. **Regex** is used for log processing to find out the metrics used for comparison of algorithms.

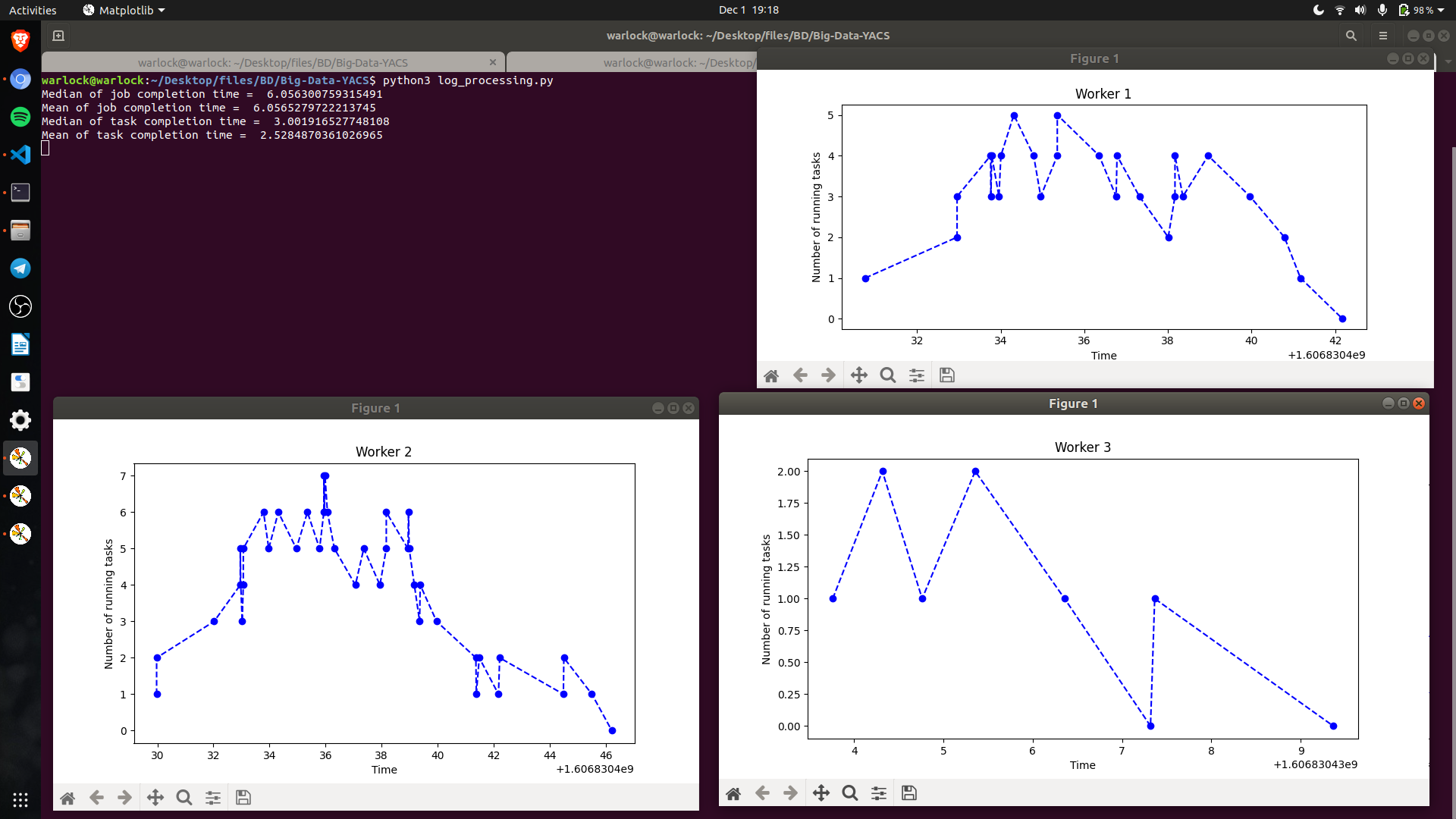
## Design

For scheduling algorithm we implemented 3 algorithms as specified in the task:-

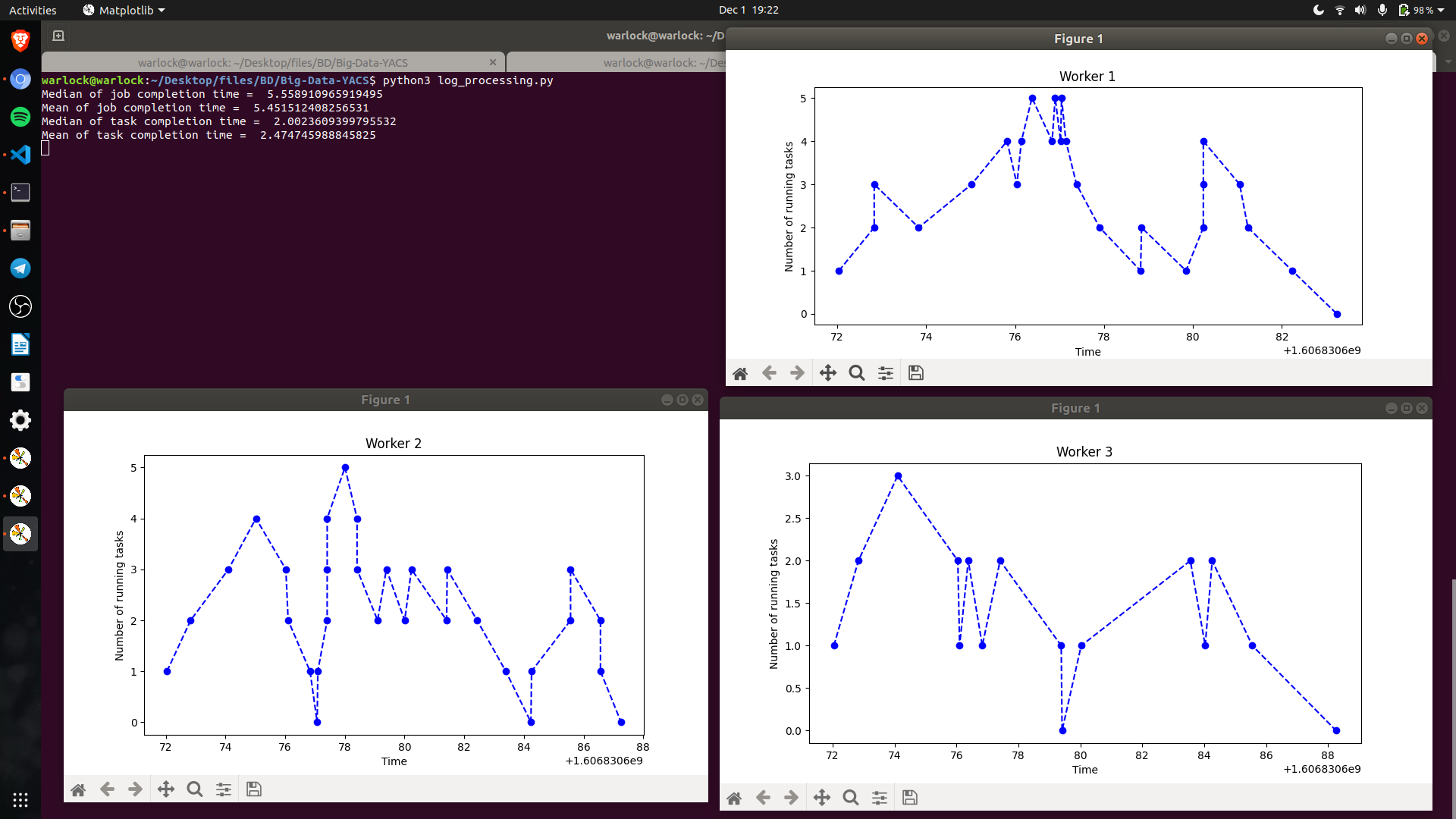
1. **Round Robin**:- Allocating the tasks to each worker iteratively. Whenever slots on a particular worker is not found, we skip to the next worker.
2. **Random scheduling**:- Randomly allocating tasks for each job to any worker.
3. **Least loaded**:- Allocating tasks to the worker having most available slots at that time.



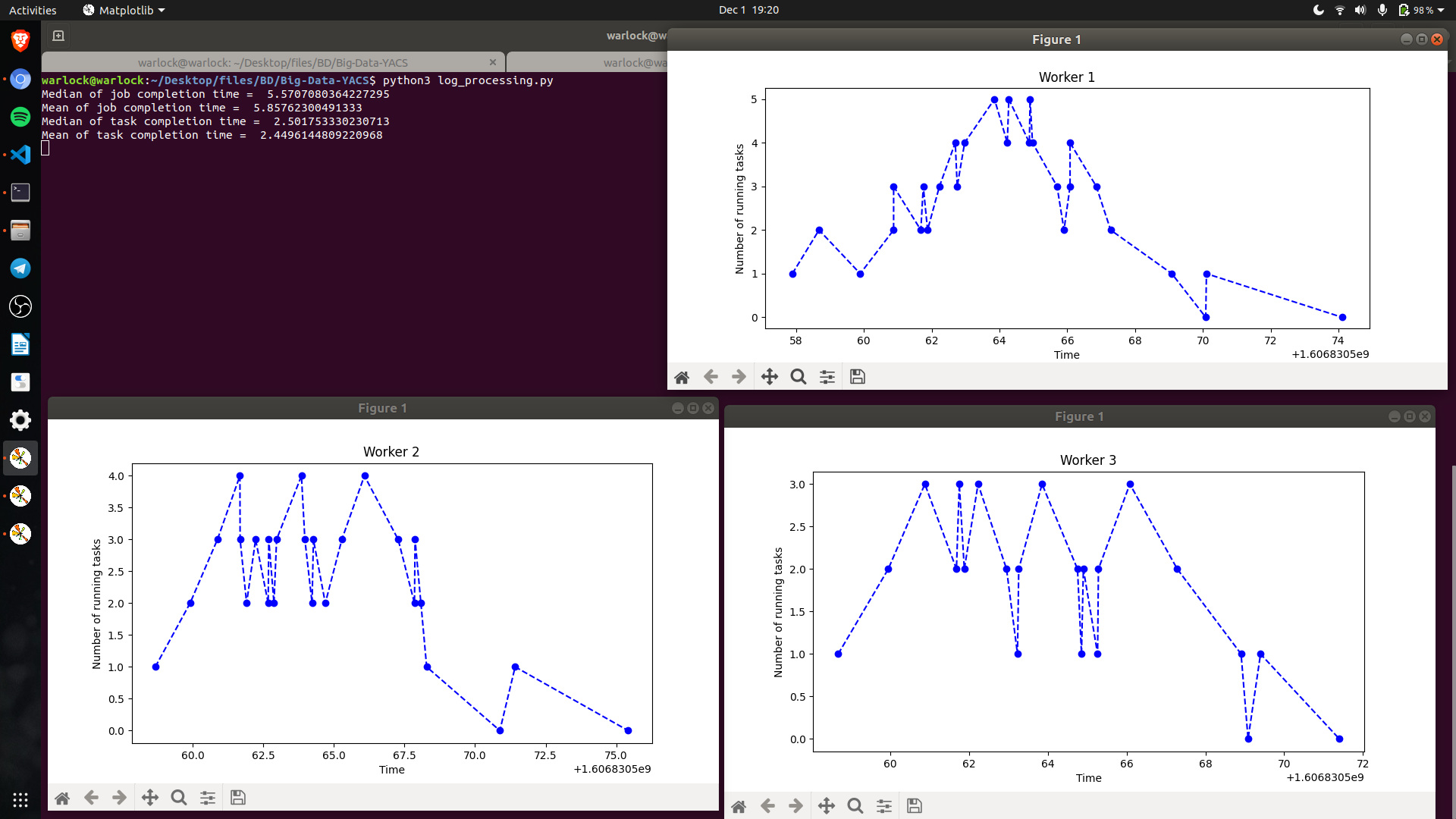
## Results



*Fig:- Least Loaded scheduling algorithm*



*Fig:- Random scheduling algorithm*



*Fig:- Round Robin scheduling algorithm*

**Note:- We have seeded our randomness to compare the given scheduling algorithm.**

Inferences drawn from the results:-

1. Average job completion time was least for Random scheduling and it is comparable with round robin.
2. Average task completion time was least for round robin scheduling.

## Problems

1. The first problem we faced was regarding threading. We started analysing the result of our code and saw the master was not scheduling more than three tasks on a worker. It was happening because our implementation was not supporting multiple threading. Finally we were able to debug our program and get the desired result.
2. One more problem was using the logging library which was completely new to us. We did some search on it and were able to implement the task.
3. We faced problems in socket programming as we were not able to configure the closed ports initially.

## Conclusion

We learnt a lot from this project as it required a clear understanding of all fields including Big Data, Computer Networking, Operating System and different python libraries.

1. We got to learn about threading and how to run multiple threads at the same time.
2. Semaphores were also used for processes and resources synchronization.
3. Logging library in python was used, which can be very helpful in tracking the communication between master and worker in our code.

## EVALUATIONS:

|  |  |  |  |
| --- | --- | --- | --- |
| SNo | Name | SRN | Contribution (Individual) |
| 1 | Shaashwat Jain | PES1201802346 | Master & Worker |
| 2 | Srishti Sachan | PES1201802126 | Log Processing |
| 3 | Rajdeep Sengupta | PES1201800144 | Scheduling Algorithms |
| 4 | Raghav Aggarwal | PES1201800312 | Master & Worker |

## (Leave this for the faculty)

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Evaluator | Comments | Score |
|  |  |  |  |

## CHECKLIST:

|  |  |  |
| --- | --- | --- |
| SNo | Item | Status |
| 1. | Source code documented |  |
| 2. | Source code uploaded to GitHub – (access link for the same, to be added in status 🡪) |  |
| 3. | Instructions for building and running the code. Your code must be usable out of the box. |  |