Comparative Study on Job Scheduling Using Priority Rule and Machine Learning

Conference Paper · December 2021

DOI: 10.1109/ETCCE54784.2021.9689812

4 authors, including:

Saydul Akbar Murad
Universiti Malaysia Pahang
18 PUBLICATIONS **50** CITATIONS
SEE PROFILE

Zafril Rizal M. Azmi
Universiti Malaysia Pahang
21 PUBLICATIONS **49** CITATIONS
SEE PROFILE

Abu Jafar Md Muzahid
Universiti Malaysia Pahang
20 PUBLICATIONS **78** CITATIONS
SEE PROFILE

2021 International Conference on Emerging Technology in Computing, Communication and Electronics (ETCCE)

2021 Emerging Technology in Computing, Communication and Electronics (ETCCE) | 978-1-6654-8364-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/ETCCE54784.2021.9689812

Comparative Study on Job Scheduling Using Priority Rule and Machine Learning

Saydul Akbar Murad
Faculty of Computing
Universiti Malaysia Pahang
26600, Pekan, Pahang, Malaysia
saydulakbarmurad@gmail.com

Zafril Rizal M Azmi*
Faculty of Computing
Universiti Malaysia Pahang
26600, Pekan, Pahang, Malaysia
zafril@ump.edu.my

Abu Jafar Md Muzahid
Faculty of Computing
Universiti Malaysia Pahang
26600, Pekan, Pahang, Malaysia
mrumi98@gmail.com

Md. Al-Imran
Information & Communication Engineering
Noakhali Science & Technology University
Noakhali, Bangladesh
mdalimran143@gmail.com

***Abstract*—Cloud computing is a potential technique for running resource-intensive applications on a wide scale. Implementation of a suitable scheduling algorithm is critical in order to properly use cloud resources. Shortest Job First (SJF) and Longest Job First (LJF) are two well-known corporate schedulers that are now used to manage Cloud tasks. Although such algorithms are basic and straightforward to develop, they are limited in their ability to deal with the dynamic nature of the Cloud. In our research, we have demonstrated a comparison in our investigations between the priority algorithm performance matrices and the machine learning algorithm. In cloudsim and Google Colab, we finished our experiment. CPU time, turnaround time, wall clock time, waiting time, and execution start time are all included in this research. For time and space sharing mode, the cloudlet is assigned to the CPU. VM is allocated in space-sharing mode all the time. We've achieved better for SJF and a decent machine learning algorithm outcome as well.**

***Keywords*—** Cloud computing, Priority rule, Time share, Space share, Adaboost Classifier.

I. Introduction

A distributed computer system that provides software, CPU, memory, storage, and other computing resources is known as cloud computing. It offers on-demand services through the internet as a pay-per-use service [1]. Virtualizing technologies utilize cloud computing to build and run their cloud computing environments. Service providers [2], [3] and internet providers benefit from cloud environments. The software, network, and facilities utilized to produce the service are the responsibility of the service provider. At the same time, internet service providers are referred to as Cloud clients or consumers. The task algorithm distributes user tasks to the cloud to optimize [4] use rates, decrease the time to wait, and balance the cloud infrastructure in order to avoid overburdening activities [1]. The scheduling might be static or dynamic. The Scheduler determines resource and task specifics in the static planning process. While the tasks and resources are planned dynamically, they are not determined from the outset. To conduct the experiment, we used the shortest job first (SJF), Longest work first (LJF), and Machine learning algorithms. CPU time, Turnaround time, Wall Clock time, waiting time, and Execution start time are all taken into account. We compared the performance matrices [5] in both time and space sharing mode. In addition, to obtain those performance matrices, we employed Adaboost Classifier method. For most of the performance matrices, SJF came out on top. The remainder of our paper is organized in the following manner. Related Work in Section II. Methodology is covered in Section III. The Simulation Results on Cloudsim are shown in Section IV, the Results and Discussion are presented in Section V, and the conclusion and future work are covered in Section VI.

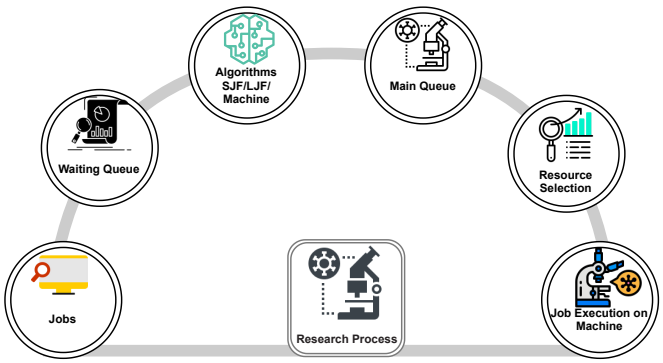


Figure 1: Research design and procedure

978-1-6654-8364-3/21/$31.00 ©2021 IEEE

II. Related Works

In [6] this paper they introduced the idea of load distribution and afterwards analyze several strategies for task planning including First Come First Serve, Shortest Job First, and Round Robin on Cloudsim. M. Ibrahim et al. Study some of the latest cloud scheduling techniques in terms of makespan (the time difference between beginning and finishing of the sequence of jobs/tasks) and the performance (number of tasks completed successfully per unit time (Makespan) and compare them experimentally) [7]. S. Santra and K. Mali offer a circular approach to the Round Robin method, and they aim to clarify during its execution the load balance scenario of a cloud server [8]. The following article [9] contains a performance analysis of the Cloud environment methods for STFQ (Start Time Fair Queuing), WRR (Weighted Round Robin) and BVT (Borrowed Virtual Time).

III. Methodology

The strategies and actions of this research were described in this chapter. In order to fulfill the study objectives, the approach outlined below is a compelling step-by-step process. This chapter's explanations will be based on the research and method design and performance evaluation. The material provided here, however, is just basic descriptions of the method.

A. Research design and procedure

The strategies and actions of this research study are outlined in this section. The strategy outlined here is designed to ensure that the study objectives are achieved through a compelling step by step method. This chapter's explanations will be focused on the working environment. To complete our experiment, we used two different types of algorithms. The first is the priority rule, and the second is machine learning. We used SJF and LJF as priority rules, and calculated our task for VM timeshare and spaceshare.The steps involved in completing the research are described in *Figure: 1*. Each step is detailed in the following sub-sections.

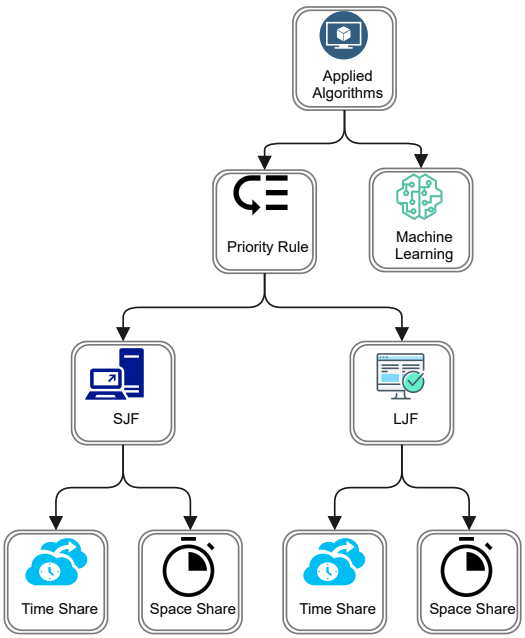


Figure 2: Procedure of applied algorithm

1) Job: In Cloudsim, a job is defined as a cloudlet. During the cloud simulation engine simulation session, Cloudlet identified the work to be performed [10]. Cloudlet is a model class in a package called 'org.cloudbus.cloudsim.' Cloudlet is one of the major models that specify the mobility needs of a cloud-based simulation engine for a cloud-based application. We put our defined cloudlet here, which indicates that the length of the cloudlet is determined by us. To set the cloudlet length, we first used the 'Random' method to produce the cloudlet length, and then we saved the entire cloudlet length. Finally, we employed all of the cloudlets for the experiment.

2) Waiting Queue: With regard to cloud services, there are many cloud users that are keen to utilize the Cloud Service Providers offerings. The requests were made to the provider and were all addressed by the cloud service provider's scheduler. Efficient use of servers may maximize system and computer share, reduce cost complexity and minimize wait times. In a cloud scenario, the service sought by the cloud user and the service provider are both random variables [11]. When the user request is received by the servers or the number of servers is restricted, then users need to wait. The notion of queuing is utilized in this respect to decrease the waiting time. Cloud clients come to the cloud provider, and they need to wait in line since they do not have servers or restricted resources (in a queue).

3) Algorithm: A specialized method for resolving a well-defined computer problem is an algorithm. In all fields of computer science, the creation and study of algorithms is crucial [12]. For this experiment we utilized a host, 100 cloudlets and 5 virtual machines. After establishing the environment, we followed two methods. First, priority rule and secondly machine learning. *Figure: 2* illustrates what was done in the next step of our experiment.

3.1) Priority Rule: Priority scheduling is a priority-based scheduling approach. The scheduler decides to operate according to the priority tasks in this approach which is distinct from other scheduling kinds, such as SJF and LJF [13]. Priorities might be dynamic or static. Static priorities are set during the formation process, whereas dynamic priorities are assigned based on the behavior of the processes in the system.

3.1.1) Shortest job first: Shortest Job First is an algorithm for the next execution of the process with the lowest runtime [14]. The manner of programming might be preemptive or non- preemptive. The average wait time for other procedures to be executed is substantially reduced. We have utilized non- preemptive in our trial.

Algorithm 1: Shortest Job First

```
for Cloudlet cloudlet : getCloudletList() do
    tempList.add(cloudlet);
end
int totalcloudlet = tempList.size();
for int i = 0; i<totalcloudlet; i++ do
    Cloudlet smallestcloudlet = tempList.get(0);
    for Cloudlet checkcloudlet: tempList do
        if smallestCloudlet.getcloudletLength() >
         checkcloudlet.getcloudletLength() then
            smallestCloudlet = checkCloudlet;
        end
    end
    sortList.add(smallestCloudlet);
    tempList.remove(smallestCloudlet);
end
```

3.1.2) Longest Job First: The algorithm LJP (Longest Job First) is a non-preemptive scheduling method. This method is based on the processes' burst time. The processes are sorted into the ready queue

by their burst timings, which are shown in descending order [15]. This method is based on the fact that the process with the longest burst duration is handled first, as the name implies.

Algorithm 2: Longest Job First

```
for Cloudlet cloudlet : getCloudletList() do
    tempList.add(cloudlet);
end
int totalcloudlet = tempList.size();
for int i = 0; i<totalcloudlet; i++ do
    Cloudlet longestCloudlet = tempList.get(0);
    for Cloudlet checkcloudlet: tempList do
        if longestCloudlet.getcloudletLength() <
         checkcloudlet.getcloudletLength() then
            longestCloudlet = checkCloudlet;
        end
    end
    longList.add(longestCloudlet);
    tempList.remove(longestCloudlet);
end
```

3.1.(1.2).1) Time Share for Cloudlet: *Figure: 3* depicts a virtual machine space-sharing strategy and a task time-sharing policy. Each virtual machine will reserve one (1) of the host's two (2) processing units, and although each task requires one (1) processing unit to complete, the policy algorithm will allocate a slice of the processing unit time to each task until both tasks are completed [16].

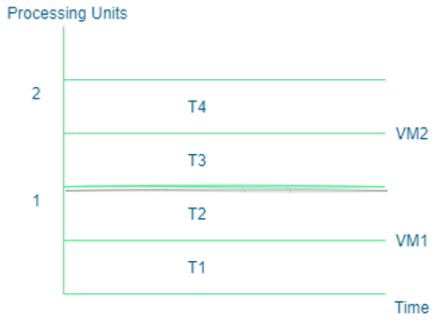


Figure 3: Space share for VM and Time share for Cloudlet.

3.1.(1.2).2) Space Share for Cloudlet: The space-shared strategy for both virtual machines and tasks is shown in *Figure: 4*. While each VM requires one (1) processing unit, each VM reserves one (1) of the host's two (2) processing units, only a task may be performed at a certain point, and the second will wait until the first job is completed [16].

3.2) Machine Learning: Machine learning is a type of data analysis that automates the process of creating analytical models. It's an artificial intelligence area based on the idea that machines can learn from data [17], [18], recognize patterns [19], [20], and make decisions with little or no human intervention [21]. We utilized eight features to train data in our study article. Cloudlet length, pesNumber, VM RAM, MIPS, VM bandwidth, Host RAM, Host storage, and Host bandwidth are among the features. Using those features, we were able to achieve our goal. We utilized Adaboost classifier for the utilization of data [22]. An AdaBoost Classifier is a meta-evaluator that starts with the fitting of a classifier to the original data set and then fits into a dataset additional copy of the classifier but adjusts weights of erroneously classified instances to further focus the classifier on

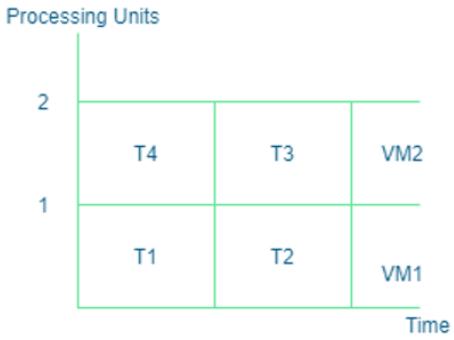


Figure 4: VMs and Tasks Space Shared Policy.

tough situations [23]. *Figure: 5* depicts the procedure we followed throughout our ML investigation through Adaboost Classifier.

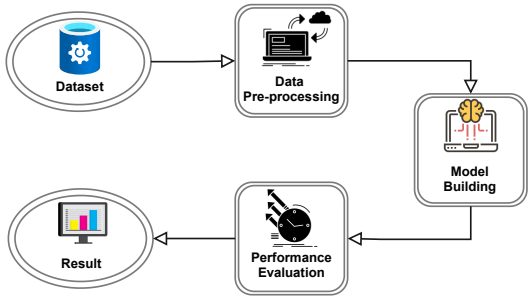


Figure 5: Experiment Workflow in Machine learning using Adaboost Classifier.

4) Main Queue: A queue, like a stack, is a popular data structure that arranges things in a sequential order. All cloudlets are placed in a queue once the priority algorithm and machine learning have been applied. Based on the algorithm, all of the cloudlets are synchronous. All of the algorithms in the main queue are ready to run.

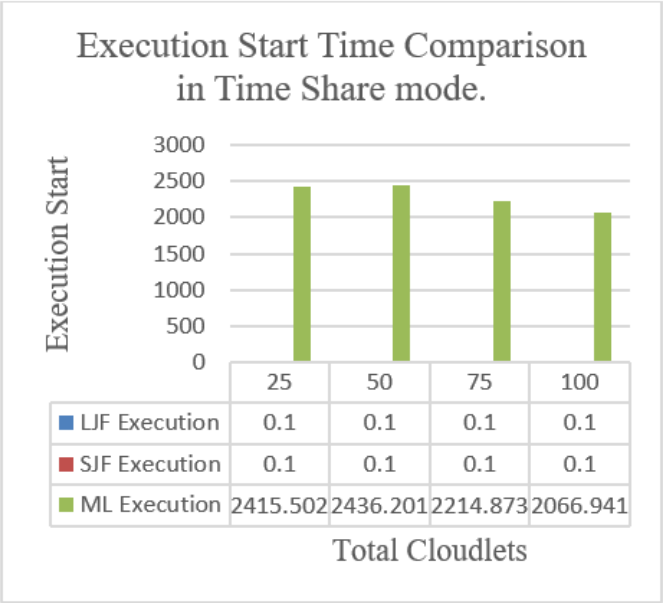
5) Resource Selection: The resource selection method in the cloud is scalable, so even if there are a huge number of resources involved, the desired information is obtained fast. Cloud resources are accessible for service-based demand. However, selecting a specific resource is quite challenging. The capacity to scale the quantity of resources is a key element of resource selection services. The purpose of resource selection is to provide a list of resources that are available to end users [24].

6) Job Execution on Machine: Jobs are ready for execution after resource selection. Job execution denotes jobs that are prepared to complete the task in the specified environment. It displays the intended outcome after completing the job. The execution time is determined by the running device's settings. A high-configuration device can complete a task in a reasonable amount of time, whereas a low-configuration device takes longer.

B. Performance Evolution

This study compared many current algorithms and machine-learning algorithms on five performance metrics: CPU time, execution start time, Turnaround times, waiting time and wall clock times in order to evaluate the performance of the suggested scheduling machines. The following sub sections describe these measures in depth.

1) CPU Time: CPU time (or process time) refers to the amount of time a central processing unit (CPU) was utilized to process instructions from a computer program or operating system, as opposed



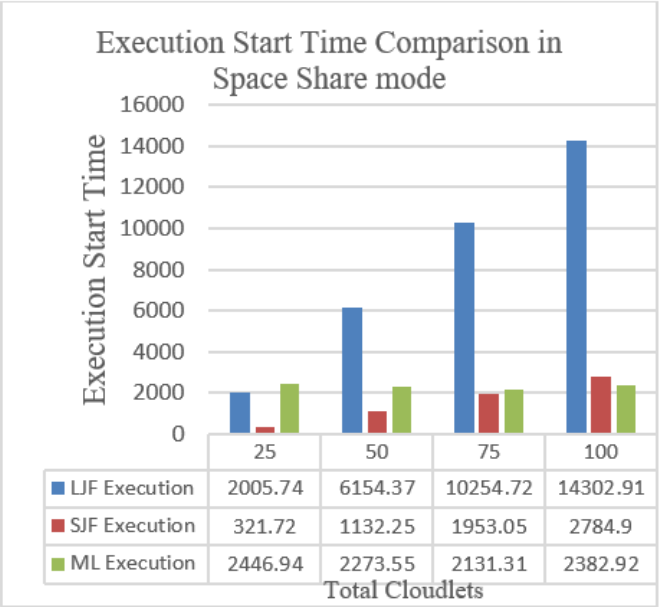
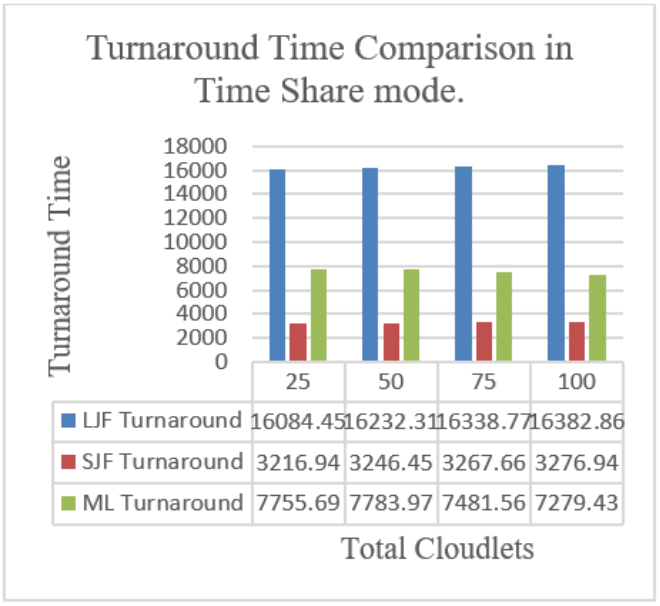


Figure 6: Execution start Time in Cloudlet time share and space share mode.



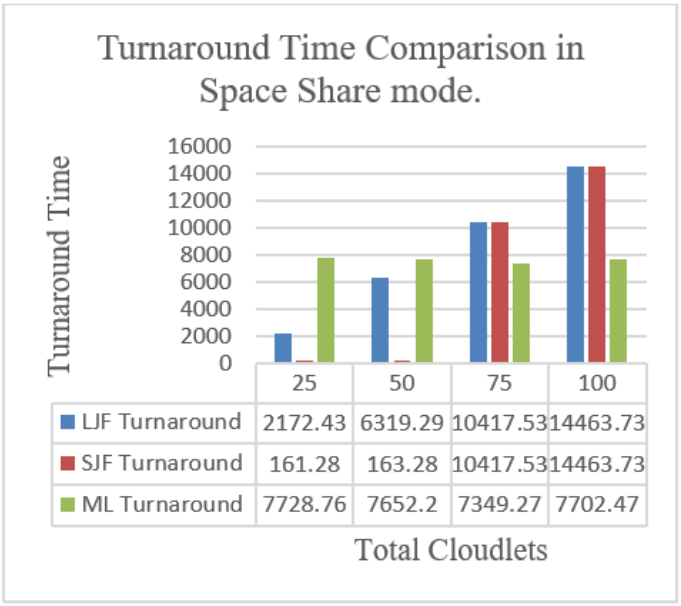


Figure 7: Turnaround Time in Cloudlet space share and Time share mode.

to elapsed time, which includes things like waiting for I/O operations or going into low-power (idle) mode. Clock ticks or seconds are used to quantify CPU time. CPU time is frequently measured as a percentage of the CPU's capacity, which is referred to as CPU use. CPU time is calculated by *equation: 1*.

$$\text{CPU Time} = \text{Process Uptime} \times \text{CPU Utilization of Process} \quad (1)$$

2) Execution Start time: The execution start time is the first time the process receives the CPU. When a large number of cloudlets arrive at the CPU, one or two cloudlets are given access to finish the task based on CPU availability. Another cloudlet must wait in the queue before being free of CPU. Cloudlet receives CPU access based on its priority. Execution start time can represent by *equation: 2*.

$$\text{Execution start time} = \text{Response time} - \text{Arrival time} \quad (2)$$

3) Turnaround Time: The total amount of time spent by a process from the time it arrives in a ready state to the time it is completed is known as turnaround time. For the same set of tasks, different CPU scheduling methods generate varying turnaround times. This is due to the fact that when the CPU scheduling method is changed, the waiting time of processes varies. *Equation: 3* represent the turnaround time.

$$\text{Turnaround time} = \text{Exit time} - \text{Arrival time} \quad (3)$$

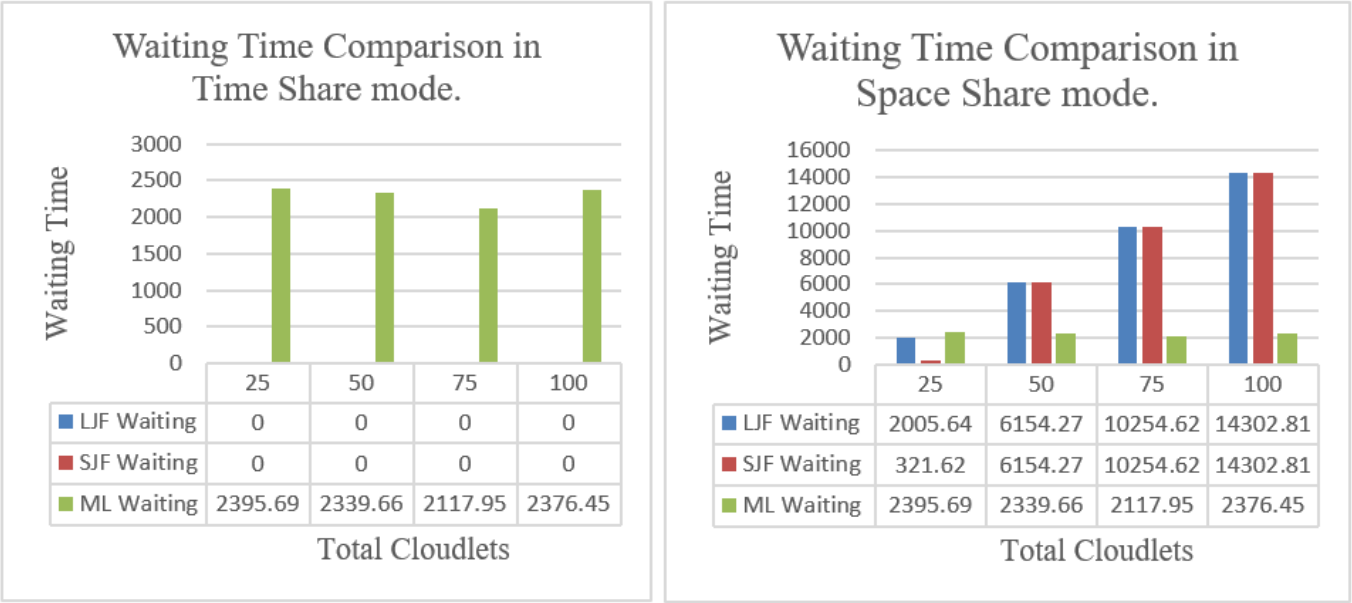


Figure 8: Waiting Time in Cloudlet space share and Time share mode.

Table I: Configuration of Host.

Cloud Hosts	1
Processing element (Pe)	2
MIPS	1000
RAM	20000
Storage	1000000
Bandwidths	1000
CPU/Host	6

4) Waiting Time: The entire time spent by the process in the ready state waiting for the CPU is known as waiting time. All cloudlets must wait in the main queue before receiving the CPU. When one work in the CPU is completed, another job is done in this CPU. The waiting time is calculated by *equation: 4*.

$$\text{Waiting time} = \text{Turn Around time} - \text{Burst time} \quad (4)$$

Table II: Configuration of Virtual Machine.

Number of VM	5
VM size	10000
MIPS	250
RAM	512
Bandwidths	1000
Pes number	1

5) Wall Clock Time: In practical computing, wall clock time is the amount of time it takes a program to run or complete its given duties, generally measured in seconds. The wall clock time for each program is set independently while the computer is multitasking and relies on how the CPU allocates resources among the applications.

IV. Simulation Results on Cloudsim

Cloudsim design has four fundamental components which are quite helpful in establishing the basic cloud computing environment. The efficiency of task algorithms is measured in this setting. Datacenter, Datacenter Broker, virtual machine and cloudlet are all components. The datacenter is responsible for providing hardware level services to cloud consumers. Datacenter brokers build and destroy virtual machines (VMs) in accordance with job requirements. It also shields the user from VM administration [25]. The tasks are processed by the virtual machine according to the policies set by the cloudlet scheduler. Cloudlet is a virtual machine job that is currently executing.

A. Simulation Parameters (Cloudsim)

Cloudsim is a simulator used to run the cloud experiment. Details regarding the host, virtual machine, Cloudsim system settings, Cloudlet, and datacenter could be found in this section. The experiment is run on cloudsim. *Table: I* shows that this experiment is carried out using only two processing element and one cloud host with a processing speed of 1200MIPS each. This host has a RAM size of 20000 MB, a memory of 1000000 MB, and a bandwidth of 10000. *Table: II* explain about Virtual Machine during the time of experiment in Cloudsim. The 5 VM machine with a storage size of 10000 has been utilized. Each VM has a ram size of 512 and the capacity to process instructions is 1000 in every second. The bandwidth capability is 1000.

Table III: Configuration of `Cloudsim` System.

Architecture	X86
Operating System	Linux
Time Zone	10
Processing Cost	3.0
Storage Cost	0.1
Memory Cost	0.05
Bandwidth Cost	0.1

Table IV: Details about `cloudlet`.

`cloudlet`t number	100
Length	Heterogeneous
File Size	300
Output Size	300
Pes number	1

Table: III shows that the experiment is being carried out on a Linux operating system with a resource location time zone of 10.0. The cost

of processing is 3.0, while the cost of memory is 0.05. The cost of storage and bandwidth are both 0.1. We came up with an excellent idea for cloudlet from *Table: IV*, which is applied in this experiment. We utilized 100 cloudlets in this case, and the input length was set by us. The length of each cloudlet varies. The input and output file sizes are both 300 KB. We came up with an excellent idea for cloudlet,

Table V: Datacenter paraments in `Cloudsim`.

Number of Datacenter	1
Number of Virtual Machine	5

which is applied in this experiment. We utilized 100 cloudlets in this case, and the input length was set by us. The length of each cloudlet varies. The input and output file sizes are both 300 KB.

V. Result and Discussion

Cloudsim and Google colab were used to analyze the impact of each Priority algorithm and Machine learning. The settings, datasets, and performance indicators discussed in the Chapter Methodology session are used in this evaluation.

Figure: 9 demonstrates that the LJF method takes the most CPU time in time sharing mode, whereas the trend is reversed in space share mode. In comparison to CPU time in time share mode, LJF CPU time is relatively low in space share mode. The ML method consumes the most CPU time in Space sharing mode.

Figure: 6 shows that under Time sharing mode, the Machine learning algorithm takes longer to start the execution than the other two algorithms. In space sharing mode, however, LJF takes longer than SJF and ML algorithms. SJF takes the least amount of time in terms of both space and time sharing.

In time sharing mode, LJF has the fastest turnaround time and the slowest SJF, as seen in *Figure: 7*. The value for ML is in the intermediate phase. We detect up and down position at various points for priority and ML algorithms in space share mode. SJF and LJF have the same value in cloudlet 75 and 100, while ML has a low value. In cloudlet 25 and 50, the ML algorithm has a longer turnaround time.

Figure: 8 demonstrates that in time sharing mode, the waiting time for both SJF and LJF algorithms is zero, but the waiting time for the ML method is excessively high. In space sharing mode, the waiting times for SJF and LJF are almost same, which is excessive when compared to the ML method. The ML algorithm has a waiting time of between 2100 and 2400 MS. This number is about 15000 for both SJF and LJF.

Figure: 10 shows that in time share mode, the LJF wall clock time is greater and the SJF time is lower. In this case, the values for the SJF, LJF, and ML algorithms are almost identical in every way. In space share mode, for LJF wall clock time is higher in cloudlet 75 and 100, where in cloudlet 25 and 50 wall clock time is higher for ML algorithm. In every way, SJF is in a lower position.

VI. Conclusion and Future Work

The experiment in this study was carried out using the CLoudsim simulator. We used two priority algorithms and a machine learning method to finish the experiment. This thesis looked at PR and ML to see what their strengths and weaknesses are in terms of CPU time, execution start time, turnaround time, waiting time, and wall clock time performance metrics. The comparison of such performance matrices in the time share and space share modes was shown in the results session. In future study, we will compare the effects of current and dynamic heuristic and meta heuristic policies in a cloud context by changing the time quantum for task execution and VM schedular. Apart from that, we will make extensive use of machine learning.

VII. Acknowledgment

The authors would like to thank the Ministry of Higher Education for providing financial support under Fundamental Research Grant Scheme (FRGS) No. FRGS/1/2019/ICT03/UMP/02/2 (University reference RDU1901194).

References

[1] I. M. Ibrahim *et al.*, "Task scheduling algorithms in cloud computing: A review," *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, vol. 12, no. 4, pp. 1041–1053, 2021.

[2] M. A. Rahim, M. Rahman, M. A. Rahman, A. J. M. Muzahid, and S. F. Kamarulzaman, "A framework of iot-enabled vehicular noise intensity monitoring system for smart city," *Advances in Robotics, Automation and Data Analytics: Selected Papers from ICITES 2020*, vol. 1350, p. 194, 2021.

[3] L. C. Kiew, A. J. M. Muzahid, and S. F. Kamarulzaman, "Vehicle route tracking system based on vehicle registration number recognition using template matching algorithm," in *2021 International Conference on Software Engineering Computer Systems and 4th International Conference on Computational Science and Information Management (ICSECS-ICOCSIM)*, 2021, pp. 249–254.

[4] A. Karim, M. A. Islam, P. Mishra, A. J. M. Muzahid, A. Yousuf, M. M. R. Khan, and C. K. M. Faizal, "Yeast and bacteria co-culture-based lipid production through bioremediation of palm oil mill effluent: a statistical optimization," *Biomass Conversion and Biorefinery*, pp. 1–12, 2021.

[5] A. J. M. Muzahid, S. F. Kamarulzaman, and M. A. Rahman, "Comparison of ppo and sac algorithms towards decision making strategies for collision avoidance among multiple autonomous vehicles," in *2021 International Conference on Software Engineering Computer Systems and 4th International Conference on Computational Science and Information Management (ICSECS-ICOCSIM)*, 2021, pp. 200–205.

[6] R. Pratap and T. Zaidi, "Comparative study of task scheduling algorithms through cloudsim," in *2018 7th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions)(ICRITO)*. IEEE, 2018, pp. 397–400.

[7] M. Ibrahim, S. Nabi, R. Hussain, M. S. Raza, M. Imran, S. A. Kazmi, A. Oracevic, and F. Hussain, "A comparative analysis of task scheduling approaches in cloud computing," in *2020 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID)*. IEEE, 2020, pp. 681–684.

[8] S. Santra and K. Mali, "A new approach to survey on load balancing in vm in cloud computing: Using cloudsim," in *2015 International Conference on Computer, Communication and Control (IC4)*. IEEE, 2015, pp. 1–5.

[9] M. V. Jambigi, V. Desai, and S. Athanikar, "Comparative analysis of different algorithms for scheduling of tasks in cloud environments," in *2018 International Conference on Computational Techniques, Electronics and Mechanical Systems (CTEMS)*. IEEE, 2018, pp. 359–361.

[10] "Cloudlet in Cloudsim Simulation - Cloudsim Tutorials." [Online]. Available: https://www.cloudsimtutorials.online/cloudlet-in-cloudsim-simulation/

[11] S. Mohanty, P. K. Pattnaik, and G. B. Mund, "A comparative approach to reduce the waiting time using queuing theory in cloud computing environment," *International Journal of Information and Computation Technology. ISSN*, pp. 0974–2239, 2014.

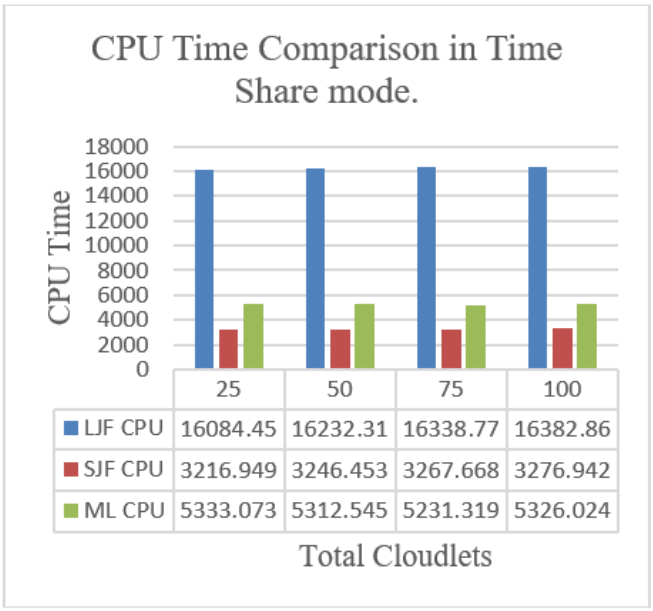
[12] "Computer science - Algorithms and complexity | Britannica." [Online]. Available: https://www.britannica.com/science/computer-science/Algorithms-and-complexity

[13] S. Chand, Q. Huynh, H. Singh, T. Ray, and M. Wagner, "On the use of genetic programming to evolve priority rules for resource constrained project scheduling problems," *Information Sciences*, vol. 432, pp. 146–163, 2018.

[14] A. P. U. Siahaan, "Comparison analysis of cpu scheduling: Fcfs, sjf and round robin," *International Journal of Engineering Development and Research*, vol. 4, no. 3, pp. 124–132, 2016.

[15] M. S. Abd Latiff, M. Abdullahi *et al.*, "Job scheduling technique for infrastructure as a service cloud using an improved league championship algorithm," in *Proceedings of the International Conference on Data Engineering 2015 (DaEng-2015)*. Springer, 2019, pp. 479–488.

[16] G. T. Hicham and E. A. Chaker, "Cloud computing cpu allocation and scheduling algorithms using cloudsim simulator." *International Journal of Electrical & Computer Engineering (2088-8708)*, vol. 6, no. 4, 2016.



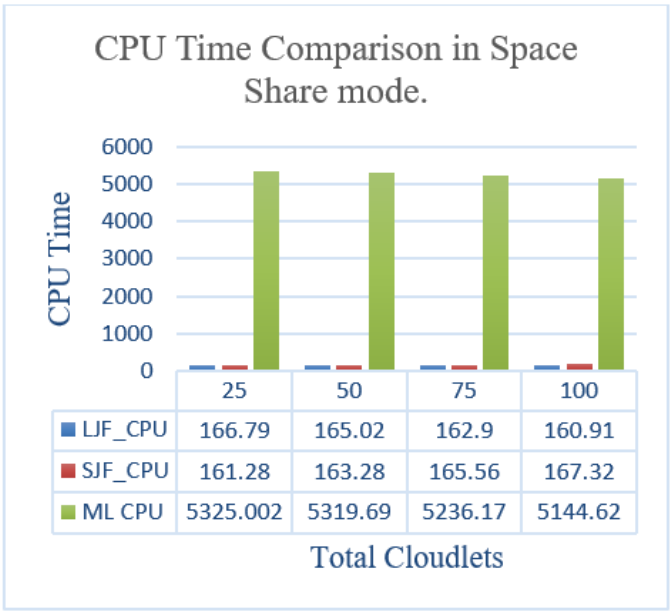
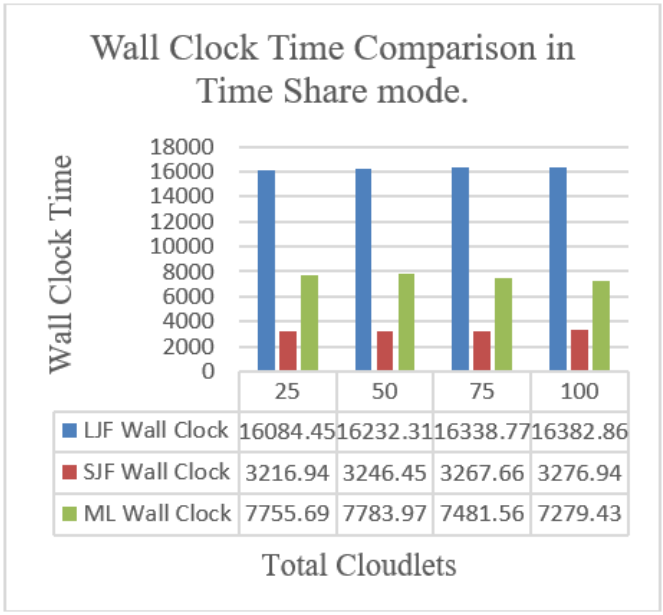


Figure 9: CPU Time in Cloudlet Space Share and Time Share mode.



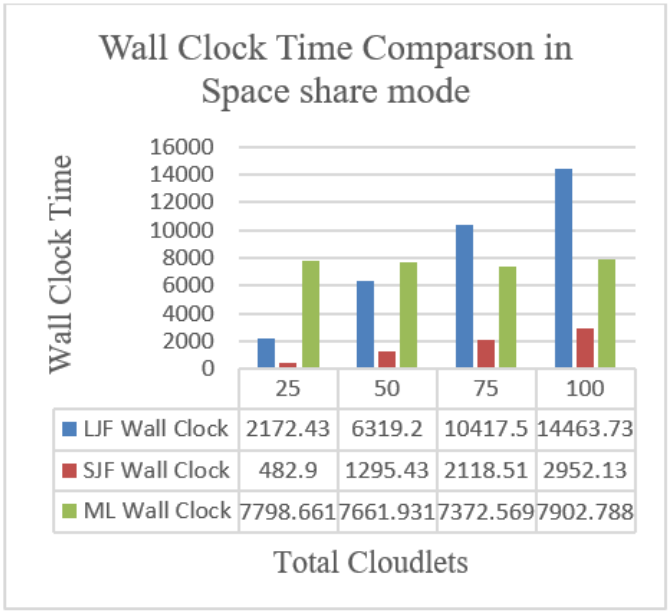


Figure 10: Wall Clock Time in cloudlet space share and Time share mode.

[17] M. Kowsher, A. Tahabilder, and S. A. Murad, "Impact-learning: a robust machine learning algorithm," in *Proceedings of the 8th international conference on computer and communications management*, 2020, pp. 9–13.

[18] S. A. Murad, Z. R. M. Azmi, Z. H. Hakami, N. J. Prottasha, and M. Kowsher, "Computer-aided system for extending the performance of diabetes analysis and prediction," in *2021 International Conference on Software Engineering & Computer Systems and 4th International Conference on Computational Science and Information Management (ICSECS-ICOCSIM)*. IEEE, 2021, pp. 465–470.

[19] A. J. M. Muzahid, S. F. Kamarulzaman, and M. A. Rahim, "Learning-based conceptual framework for threat assessment of multiple vehicle collision in autonomous driving," in *2020 Emerging Technology in Computing, Communication and Electronics (ETCCE)*. IEEE, 2020, pp. 1–6.

[20] M. M. Hasan, M. S. Islam, and S. Abdullah, "Robust pose-based human fall detection using recurrent neural network," in *2019 IEEE International Conference on Robotics, Automation, Artificial-intelligence and Internet-of-Things (RAAICON)*. IEEE, 2019, pp. 48–51.

[21] Z. R. M. Azmi, K. A. Bakar, A. H. Abdullah, M. S. Shamsir, and W. N. W. Manan, "Performance comparison of priority rule scheduling algorithms using different inter arrival time jobs in grid environment," *International Journal of Grid and Distributed Computing*, vol. 4, no. 3, pp. 61–70, 2011.

[22] M. A. Rahman, N. Zaman, A. T. Asyhari, S. N. Sadat, P. Pillai, and R. A. Arshah, "Spy-bot: Machine learning-enabled post filtering for social

network-integrated industrial internet of things," *Ad Hoc Networks*, vol. 121, p. 102588, 2021.

[23] "sklearn.ensemble.AdaBoostClassifier â€" scikit-learn 0.24.2 documentation." [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html

[24] C. Banerjee, A. Kundu, S. Bhaumik, R. S. Babu, and R. Dattagupta, "Framework on service based resource selection in cloud computing," *International Journal of Information Processing and Management (IJIPM)*, vol. 3, no. 1, pp. 17–25, 2012.

[25] R. Alturki, H. J. Alyamani, M. A. Ikram, M. A. Rahman, M. D. Alshehri, F. Khan, and M. Haleem, "Sensor-cloud architecture: A taxonomy of security issues in cloud-assisted sensor networks," *IEEE Access*, vol. 9, pp. 89 344–89 359, 2021.