Implementing FCFS and SJF for finding the need of Reinforcement Learning in Cloud Environment

Prathamesh Lahande[1*], *Dr. Parag* Kaveri[2*]

[1] Symbiosis Institute of Computer Studies and Research, Pune, India
[2] Symbiosis Institute of Computer Studies and Research, Pune, India

Abstract. Cloud has grown significantly and has become a popular service-oriented paradigm offering users a variety of services. The end-user submits requests to the cloud in the form of tasks with the expectation that they will be executed at the best possible lowest time, cost and without any errors. On the other hand, the cloud executes these tasks on the Virtual Machines (VM) by using resource scheduling algorithms. The cloud performance is directly dependent on how the resources are managed and allocated for executing the tasks. The main aim of this research paper is to compare the behaviour of cloud resource scheduling algorithms: First Come First Serve (FCFS) and Shortest Job First (SJF) by processing high-sized tasks. This research paper is broadly divided into four phases: the first phase includes an experiment conducted by processing approximately 80 thousand tasks from the Alibaba task event dataset using the resource scheduling algorithms: FCFS and SJF on the cloud VMs under different circumstances; the second phase includes the experimental results; the third phase includes a empirical analysis of the behaviour of resource scheduling algorithms; the last phase includes the proposed need of Reinforcement Learning (RL) to improve cloud resource scheduling and its overall performance.

1 Introduction

Cloud computing has gained immense importance today by becoming a need-of-the-hour technology [17]. The end-users prefer to opt for cloud computing for executing their tasks rather than on their local servers [18]. Several requests in the form of high-sized heavy tasks are submitted to the cloud for execution which the cloud executes on the Virtual Machines (VM) using resource scheduling algorithms [1] [12]. The cloud has to ensure a smooth flow of task execution, irrespective of the variations in task loads and the circumstances occurring while the tasks are processed. The performance of the cloud is directly proportional to how the resources are handled and allocated [15]. If the resources are scheduled and managed efficiently, the cloud can execute a more significant number of tasks, thereby providing satisfactory results to the end users [4]. Hence, to provide the best results [19], resource scheduling is an important aspect of the cloud [18]. Therefore, it is essential to research and compare the resource scheduling algorithms under various different circumstances. The first

* Corresponding author: parag.kaveri@sicsr.ac.in

phase of this paper includes conducting an experiment on the WorkflowSim [16] environment in ten different circumstances where tasks are executed using resource scheduling algorithms: First Come First Service (FCFS) and Shortest Job First (SJF) in a total of ten scenarios. The major reason for having ten different scenarios is to perform a thorough evaluation of the behaviour of the resource scheduling algorithms under various circumstances. The performance parameters considered for this study are: Average Start Time (AST), Average Completion Time (ACT), Average Turn Around Time (ATAT), Average Waiting Time (AWT), and Average Cost (AC). The results obtained from the experiment conducted with respect to AST, ACT, ATAT, AWT, and AC are used for performing the empirical analysis using the Linear Regression Equation and R2 analysis model, which helps to understand the performance of an algorithm under various circumstances. The last phase includes the use of Reinforcement Learning (RL) [20] [21] [23] to improve the resource scheduling process and ultimately improve the overall cloud performance. The rest of the paper is organised as follows: Section 2 provides the related works. Section 3 provides the design of the experiment. Section 4 includes the experimental results. Section 5 includes the Empirical Analysis of Resource Scheduling Algorithms. Section 6 includes using Reinforcement Learning for improving Resource Scheduling followed by the conclusions in Section 7.

2 Related Works

The cloud is a complex environment where improving cloud performance through improved resource scheduling where several researchers have investigated and addressed these concerns. A survey conducted by International Data Corporation (IDC) found that the cloud has a number of issues with performance, availability, added processing expenses, etc. [18]. A dynamic priority task scheduling system has been proposed by the authors [1]. The results of the experiments demonstrate that this approach enhances load balancing and resource scheduling, especially under conditions of high work load. Researchers have created an Optimal User Scheduling for Multi-Cloud algorithm, which was proven through testing to be scalable, adaptable, and effective in a variety of circumstances [10]. A task-scheduling system based on dependability perception has been suggested [9]. The strategy suggested in this work results in a resource scheduling scheme that is more suitable, and the optimization effect becomes stronger as the number of jobs rises. The relevant characteristics of the cloud resource failure rule can also be highly predicted by it. In order to maximise usage, resource scheduling is essential, the authors have discussed the most effective scheduling methods to boost cloud performance. [11]. In order to improve cache resource scheduling, the study has developed an efficient resource consumption model [3]. The execution time, power consumption, and energy consumption of this effective resource usage are better to those of other resource techniques. Researchers have examined a number of scheduling strategies, including the greedy approach, the dynamic approach, the quality of service parameter-based approach, and others [5]. This study also looks at other heuristic scheduling methods. To model and simulate the cloud in terms of the cost of service delivered utilising resource scheduling, the WorkflowSim environment has been employed [8]. The heuristic methods have been compared by processing tasks in the cloud while taking resource scheduling into account [13]. A novel VM-assign load balancing technique that efficiently distributes requests among all accessible VMs has been presented in response to the load balancing issue [15]. Algorithms for scheduling cloud computing cluster resources have been reviewed in the literature [6]. Resource scheduling in IoT simulation frameworks is the study's main focus [2]. To decrease latency problems and queue processing time by machines and other intermediary processes, and hence improve cloud availability, a resource scheduling and queuing strategy has been developed [7].

3 Design of the Experiment

3.1 Experiment Configuration and Simulation Environment

The WorkflowSim [16] java-based cloud simulation framework is used to configure the cloud environment. For scheduling the tasks on the cloud VMs, the WorkflowSim environment incorporates the cloud resource scheduling algorithms FCFS and SJF [14] [17] [22]. Alibaba task event dataset generates 80,386 tasks which are submitted to the cloud for processing. A total of ten scenarios are considered for testing the effectiveness of FCFS and SJF under various circumstances. The first scenario involves executing all tasks on 5 VMs; the second scenario involves executing all tasks on 10 VMs; so on until the tenth scenario involves executing all tasks on 50 VMs. These tasks are scheduled and processed using the FCFS and SJF algorithms separately. Figure 1 depicts the entire flow of the experiment conducted.

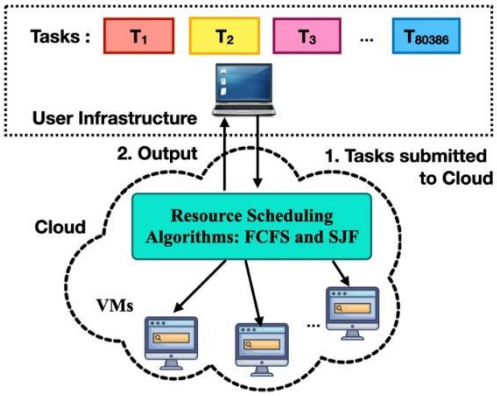

Figure 1. The flow of the Experiment

3.2 Dataset of the Experiment

The dataset used in this experiment consists of Alibaba task event dataset which generates 80,386 tasks. Each task is uniquely identified by its task id, the task creation time is denoted by its created timestamp and the amount of time it wants to use the cloud VM is denoted by the task's planned CPU.

Table 1. depicts the experiment dataset used.

Table 1. Planned CPU vs Task Size.

CPU	Tasks	CPU	Tasks	CPU	Tasks	CPU	Tasks
10	9017	60	77	100	17529	600	5
40	372	65	109	200	8	800	2
50	52791	70	76	300	2		
55	122	75	272	400	4		

4 Experiment Results

This section represents the experimental results represented in Table 2 for the resource scheduling algorithms FCFS and SJF with respect to performance parameters considered for the study which are: Average Start Time (AST), Average Completion Time (ACT), Average Turn Around Time (ATAT), Average Waiting Time (AWT) and Average Cost (AC).

Table 2. Performance Comparison of Resource Scheduling Algorithm with respect to AST, ACT, ATAT, AWT, and AC

	AST (in ms)		ACT (in ms)		ATAT (in ms)		AWT (in ms)		AC (in cost units)	
VMs	**FCFS**	**SJF**	**FCFS**	**SJF**	**FCFS**	**SJF**	**FCFS**	**SJF**	**FCFS**	**SJF**
5	47271.01	42280.96	47276.67	42286.62	13190.43	8200.38	13184.76	8194.71	1.69896	1.69903
10	35024.10	34598.04	35029.76	34603.70	943.51	517.45	937.85	511.79	3.03227	3.03349
15	34112.35	34102.76	34118.02	34108.42	31.77	22.18	26.11	16.51	4.1472	4.15305
20	34093.00	34091.08	34098.66	34096.74	12.41	10.49	6.75	4.83	5.05383	5.06433
25	34089.16	34088.51	34094.82	34094.17	8.58	7.92	2.91	2.26	5.78084	5.80541
30	34087.83	34087.53	34093.49	34093.20	7.24	6.95	1.58	1.29	6.38752	6.41645
35	34087.21	34087.06	34092.87	34092.72	6.63	6.47	0.96	0.81	6.86308	6.91813
40	34086.87	34086.79	34092.54	34092.45	6.29	6.21	0.63	0.54	7.27191	7.33259
45	34086.67	34086.62	34092.33	34092.29	6.09	6.04	0.43	0.38	7.59059	7.65555
50	34086.54	34086.51	34092.20	34092.18	5.96	5.93	0.3	0.27	7.85506	7.93602

From above Table 2, we can observe the performance of the SJF algorithm is better than the FCFS algorithm across all the scenarios and circumstances.

5 Empirical Analysis of Resource Scheduling Algorithms

This section includes the empirical analysis of resource scheduling algorithms FCFS and SJF using Linear Regression and R^2 analysis for all ten scenarios. Table 3 shows an empirical analysis of resource scheduling algorithms with respect to the AST and ACT.

Table 3. Empirical Analysis of Resource Scheduling Algorithms for AST, ACT, ATAT, AWT, AC.

	AST		**ACT**		**ATAT**		**AWT**		**AC**	
Criteria	**FCFS**	**SJF**	**FCFS**	**SJF**	**FCFS**	**SJF**	**FCFS**	**SJF**	**FCFS**	**SJF**
Linear Regression Equation	y = -759.81x + 39681	y = -469.23x + 37540	y = -759.81x + 39687	y = -469.23x + 37546	y = -759.81x + 5600.8	y = -469.23x + 3459.8	y = -759.81x + 5595.2	y = -469.23x + 3454.1	y = 0.6604x + 1.9358	y = 0.67x + 1.9162
Regression Line Slope	-759.81	-469.23	-759.81	-469.23	-759.81	-469.23	-759.81	-469.23	0.6604	0.67
Slope Sign	Negative	Negative	Negative	Negative	Negative	Negative	Negative	Negative	Positive	Positive
Line Y-Intercept	39681	37540	39687	37546	5600.8	3459.8	5595.2	3454.1	1.9358	1.9162
Relationship	Negative	Negative	Negative	Negative	Negative	Negative	Negative	Negative	Positive	Positive
R^2 value	0.3079	0.3038	0.3079	0.3038	0.3079	0.3038	0.3079	0.3038	0.9395	0.9421
VM analysis	↑ VM = ↓ AST	↑ VM = ↓ AST	↑ VM = ↓ ACT	↑ VM = ↓ ACT	↑ VM = ↓ ATAT	↑ VM = ↓ ATAT	↑ VM = ↓ AWT	↑ VM = ↓ AWT	↑ VM = ↑ Cost	↑ VM = ↑ Cost
Performance	SJF > FCFS		SJF > FCFS		SJF > FCFS		SJF > FCFS		SJF > FCFS	

From the above empirical analysis represented in Table 3, we can observe the following for all the scenarios:

- ↑ VM = ↓ AST: As the no. of VMs increases, the AST required gradually decreases.
- ↑ VM = ↓ ACT: As the no. of VMs increases, the AST required gradually decreases.
- ↑ VM = ↓ ATAT: As the no. of VMs increases, the AST required gradually decreases.
- ↑ VM = ↓ AWT: As the no. of VMs increases, the AST required gradually decreases.
- ↑ VM = ↑ Cost: As the no. of VMs increases, the AST required gradually increases.

6 Reinforcement Learning for improving Resource Scheduling

From the experiment conducted, results obtained and empirical analysis, we can state that the best-performing resource scheduling algorithm with respect to performance parameters such as AST, ACT, ATAT, AWT, and AC is the SJF algorithm. The main advantage of using SJF is that it favours the smaller tasks, thereby improving the overall time and cost of the cloud system. On the other hand, the FCFS algorithm is simple to implement, but does not provide similar results as SJF. In spite of the resource scheduling algorithm SJF outperforming FCFS, it cannot be implemented or applied to the cloud because the completion time and the planned CPU of a task must be predicted or known earlier, which is difficult with-out the use of a Machine Learning (ML) technique. Also, the SJF algorithm gives priority to tasks with low planned CPU, the tasks with longer planned CPU may starve and be denied service, violating the reliability feature of the cloud. The resource scheduling algorithm FCFS executes a certain task entirely and then schedules the next task from the task queue. This causes a higher AWT among the tasks, thereby decreasing the on-demand availability of the cloud. Also, since most of the cloud systems today are time-shared, FCFS is not suitable for providing optimal results. While processing the tasks on the cloud VMs, the resource scheduling algorithms on the cloud are statically fixed. A certain resource scheduling algorithm is implemented and used for all the different task loads across all different scenarios and circumstances. The behaviour of resource scheduling algorithms differs by task loads, scenarios, and circumstances, and fixing a certain resource scheduling algorithm for all the cases will output limited results. Also, the nature of FCFS algorithm is non-pre-emptive i.e. the on-going task will not be interrupted until its execution is finished. On the contrary, the SJF algorithm is pre-emptive in nature. Hence, in order to solve the above problems, there is a need to provide an external intelligence to the cloud system.

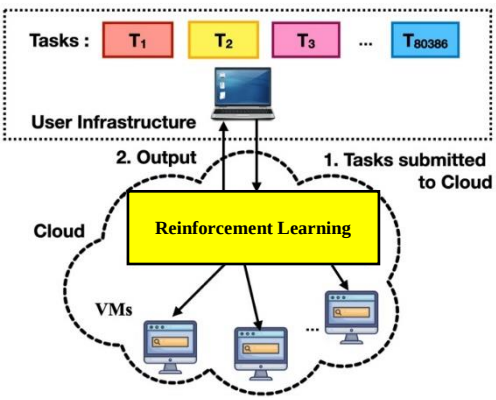

Figure 2. Proposed Reinforcement Learning to improve resource scheduling

Recently, a Machine Learning technique called the Reinforcement Learning mechanism has shown acceptable results when applied to any system, without having the need for any past data. Figure 2. depicts the proposed RL technique to improve the resource scheduling process. The entire mechanism of RL is reward and feedback based, where any system can learn and adapt to any environment without having any past data. This RL mechanism can be implemented to solve the above problems and make the resource scheduling process dynamic in nature. With RL, the cloud will initially go into a trial-and-error phase, and with a proper feedback mechanism, the resource scheduling can be made dynamic to enhance the cloud performance and achieve load balancing.

7 Conclusions

An experiment was conducted in this research paper using the WorkflowSim environment where heavy tasks were executed on the cloud VMs using various resource scheduling algorithms, and the behaviour of the resource scheduling algorithms was compared. From the experiment conducted, results obtained, and empirical analysis performed, it has been observed that the cloud resource scheduling algorithm used cannot be implemented on the real cloud since they suffer from serious pitfalls individually and can cause improper resource scheduling. The primary reason is that the resource scheduling algorithms are statically fixed and used to execute all the tasks on the cloud VMs. Hence, to provide a dynamic model, improve the resource scheduling process, and solve the individual issues of the algorithms, A Machine Learning (ML) technique known as the Reinforcement Learning (RL) mechanism can be used to solve these problems. The RL technique provides satisfactory better results when applied to any system. With RL, the cloud system will initially be in a learning phase. With trial-and-error learning and proper feedback mechanisms, the cloud will slowly and gradually adapt, and ultimately, it will learn over a period of time. The RL technique can also be used to solve the individual issues of the resource scheduling algorithms. By implementing the RL technique in the cloud and employing the proposed design, the cloud can be put through testing in any testing environment, such as the sandbox, where cloud performance can be improved. The cloud system can be handed over to the user for final use after it has been finalized and the majority of run-time errors have been removed. This cloud system will be more beneficial because the cloud can now handle the load with minimal errors. This comprehensive study will aid in the pre-deployment phase of the cloud and allow for the use of the pay-as-you-go model with fewer SLA violations.

References

1. Bo, Y., Feng, L., & Xiaoyu, Z., *Cloud computing task scheduling algorithm based on dynamic priority*, (2022).
2. Li, F., Liao, T. W., & Cai, W., *Research on the collaboration of service selection and resource scheduling for IoT simulation workflows*, (2022).
3. Sodinapalli, N. P., Kulkarni, S., Sharief, N. A., & Venkatareddy, P., *An efficient resource utilization technique for scheduling scientific workload in cloud computing environment*, (2022).
4. Gao, M., Li, Y., & Yu, J., *Workload Prediction of Cloud Workflow Based on Graph Neural Network*, (2021).
5. Kaur, G., & Bala, A., *Prediction based task scheduling approach for floodplain application in cloud environment*, (2021).

6. Khallouli, W., & Huang, J., *Cluster resource scheduling in cloud computing: literature review and research challenges*, (2021).
7. Patil, S. S., & Brahmananda, S. H., *Latency Aware Resource Scheduling and Queuing*, (2021).
8. Rajput, R. K. S., Hussain, R., & Goyal, D., *Modelling and Simulation of Cloud Service Cost Analysis using Resource Scheduling*, (2021).
9. Yuejuan, K., Zhuojun, L., & Weihao, O., *Task Scheduling Algorithm Based on Reliability Perception in Cloud Computing*, (2021).
10. Zhang, B., Zeng, Z., Shi, X., Yang, J., Veeravalli, B., & Li, K.: *A novel cooperative resource provisioning strategy for Multi-Cloud load balancing*, (2021).
11. Rupali, & Mangla, N., *Resource Scheduling on Basis of Cost-Effectiveness in Cloud Computing Environment*, (2020).
12. V, A., & Bhalaji, N., *Load balancing in cloud computing using water wave algorithm*, (2019).
13. Madni, S. H. H., Abd Latiff, M. S., Abdullahi, M., Abdulhamid, S. M., & Usman, M. J., *Performance comparison of heuristic algorithms for task scheduling in IaaS cloud computing environment*, (2017).
14. Muhammad Akhtar, Bushra Hamid, Inayat ur-Rehman, Mamoona Humayun, Maryam Hamayun, Hira Khurshid., *An Optimized Shortest job first Scheduling Algorithm for CPU Scheduling*, (2015).
15. Domanal, S. G., & Reddy, G. R. M., *Optimal load balancing in cloud computing by efficient utilization of virtual machines*, (2014).
16. Chen, W., & Deelman, E.: *WorkflowSim, A toolkit for simulating scientific workflows in distributed environments*, (2012).
17. Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., & Zaharia, M., *A view of cloud computing*, (2010).
18. Dillon, T., Wu, C., & Chang, E., *Cloud Computing: Issues and Challenges*, (2010).
19. Vengerov, D., *A reinforcement learning approach to dynamic resource allocation*, (2007).
20. Andrew, Alex., *Reinforcement Learning*, Kybernetes, (1998).
21. Richard S. Sutton, Andrew G. Barto., *Reinforcement Learning - An introduction*, (1998).
22. Uwe Schwiegelshohn, Ramin Yahyapour., *Analysis of First-Come-First-Serve Parallel Job Scheduling*, (1998).
23. Kaelbling L.P., Littman M.L., Moore A.W., *Reinforcement learning: A survey*, (1996).