HASIL CEK_IJAIN_Template

by Universitas Ahmad Dahlan Yogyakarta 35

Submission date: 20-Nov-2023 02:18PM (UTC+0700)
Submission ID: 2233939018
File name: IJAIN_Template.doc (617K)
Word count: 3901
Character count: 21192

Job Scheduling Reservations on Cloud Resources

Ardi Pujiyanta [a,1,*], Fiftin Noviyanto [b,2]

[a,b]Department of Informatics, Universitas Ahmad Dahlan, Yogyakarta, Indonesia
[1] ardipujiyanta@tif.uad.ac.id *; [2] fiftin.noviyanto@tif.uad.ac.id
* ardipujiyanta@tif.uad.ac.id

ARTICLE INFO

Article history
Received
Revised
Accepted

Keywords
Cloud computing
Reservation
Virtual view
Waiting time
makespan

ABSTRACT (10PT)

The current application of cloud computing focuses more on research problems. One of the main problems in the cloud is job allocation. Jobs are dynamically allocated to the server processor. Cloud provides virtualized Computing hardware. All cloud virtualized hardware is available to users on demand and can be dynamically upgraded. Excessive workload can be avoided by using several related algorithms. Resource scheduling is critical in research in the cloud, due to its large execution time and resource costs.

The differences in resource scheduling criteria and parameters used cause various categories of Resource Scheduling Algorithms. Resource scheduling has a goal, identifying the right resources to schedule workloads in a timely manner and improving the effectiveness of resource utilization. In other words, minimizing workload completion time. Mapping the right workloads to resources will result in good scheduling. Another goal of resource scheduling is to identify adequate and appropriate workloads. So it can support scheduling of multiple workloads, to meet various QoS requirements in cloud computing. The aim of this research is to determine the value of waiting time, idle time and makespan on cloud resources. The proposed method is to sort the arrival times of jobs with the least workload and place the jobs on a virtual view, before scheduling them on cloud resources. Experimental results show that the proposed method has smaller idle time, waiting time and makespan compared to the FCFS and Backfilling methods.

1. Introduction

Cloud computing uses virtualization technology, in offering services. The services provided can be in the form of storage computing via the internet network. Task allocation is one of the main problems in the cloud[1][2][3]. Task allocation can be done dynamically on the server processor. An unlimited collection of resources in the cloud is used for various computing needs [3]. Distributed platforms are utilized efficiently, to get the best resource management services from cloud systems [4]. There are many techniques for effective resource management in the cloud. Such as cloud job scheduling, resource migration, etc. Choosing the right technique will help save costs and good response time. So it benefits cloud users[5][6][7]. The cloud provides virtualized computing hardware similar to a public utility, so it is called Infrastructure as a Service (IaaS). Services are

available to users on demand and can be improved dynamical[18]The cloud computing service model refers to applications and software platforms, hence the name Software-as-a Service (SaaS)[8].

Scheduling is the distribution of certain work on resources to be completed efficiently. The main objectives are (i) reducing deadlines and maximizing resource utilization, (ii) optimizing the server in executing tasks, and (iii) working on higher priority jobs first and reducing completion time. Another advantage of scheduling is increasing system throughput and getting better performance[9][10]. The required level of service quality can be met by the minimum number of resources used and the workload can be maintained, or by minimizing the completion time of the workload (maximizing throughput)[11]. Mapping workloads to resources is necessary in scheduling. So identifying sufficient workloads will support scheduling of several workloads. QoS requirements such as CPU utilization, availability, reliability, security, etc., will be met[12].

In backfilling scheduling there are two things that are generally measured, the first is the accuracy of predictions and the second is the measurement of scheduling performance[13]. In dynamic cloud scheduling, a backfilling algorithm is used by dividing tasks into two queues[14]. The proposed method is the Simple Backfilling Algorithm (SBA) and DCBA in cloud computing. Both algorithms provide good performance for balanced or moderate workloads and also provide better performance when the workload becomes heavier. This method can also be implemented for all cloud tasks in future work[15][16]. The applied technique combines FCFS with a bacfilling algorithm. How it works is by scanning the queue in real-time. The proposed algorithm allows jobs at the back of the queue to be processed without delaying the head of the queue. Further experimental results show that the number of initial reservations accepted by a cluster must be below a threshold to maintain cluster performance[17].

In his research[18] used the M/G/1 queuing system. Strategic customers must decide whether to reserve a server first (and thus receive higher priority) or ignore the reservation. Server reservations in advance are subject to a fee. In this study, customer behavior strategies, equilibrium outcomes, and revenue maximization policies are characterized. Customers will be charged according to the amount of resources used. The main problem CPs face is choosing the right PM so that the new VM host still meets end user requirements. The distribution characteristics and scalability of cloud resources are taken into consideration[19]. In this Paper[20] proposed Static Independent Task Scheduling on Virtual Servers. Where tasks are allocated to VMs by measuring the availability of each resource. Processing power, cost, and amount of processing are used in grouping tasks.

From the literature review, we can observe that factors such as idle time, waiting time, resource availability and makespan, are considered for the decision to assign resources to a job. In the research that will be carried out, try to overcome the above. The factors mentioned above are the main focus in mapping jobs to virtual machines to get an optimal schedule. This research proposes the FCFS-slot free method which is used to identify idle resources, by utilizing user-submitted parameters to reduce resource execution delays, increase makespan values, and reduce job waiting times.

2. Method

2.1. System Architecture for Cloud Computing

Figure 1 shows our cloud service system, where the number of virtual machines (VMs) is equal to the number of machines in the logical view. Virtual machines (VMs) are a subset of cloud resources that can be allocated to cloud services. The proposed system consists of cloud system information (CSI), Logical view(LV), Local scheduler (LS).

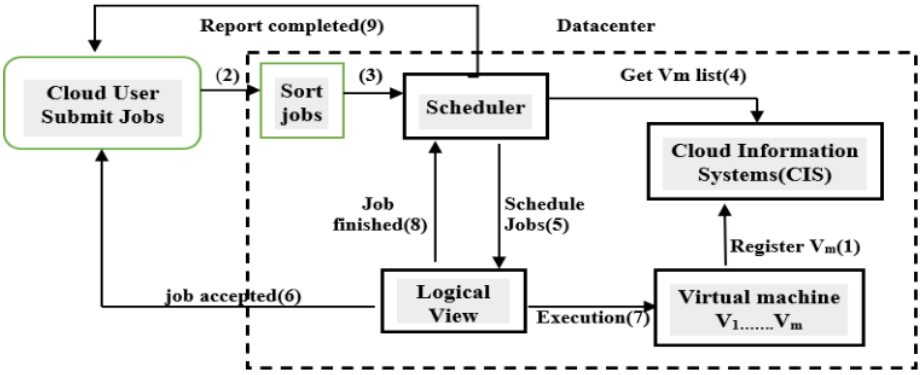

Fig 1 Proposed job allocation flow in the cloud

Figure 1 can be explained as follows, the number and status of Virtual Machines in a logical view are registered in the CIS, with the initial status of the Virtual Machine being free (List resource (1)). The user submits a reservation (2), then the work enters the task pool to be accommodated and sorted based on priority, then submitted to the scheduler (3). The next step, the scheduler will check the Virtual Machine status on CIS (4), whether there is a Virtual Machine status that can be used. If the Virtual Machine status is free then schedule job (5) in logical view. All jobs that have been scheduled in the logical view will be sent to the user that the job is accepted (job accepted (6), and executed (7) at a certain timeslot and a certain Virtual Machine number in the physical view. Virtual Machines that have finished executing in the logical view are also finished executed on the physical view. The status of the Virtual Machine on the CIS is changed by the logical view to free, and the scheduler gets a notification from the logical view that a job has finished executing (8). The scheduler then informs the user that the job has finished executing (9).

2.2. Proposed algorithm

In this section, we present a new scheduling algorithm that maximizes resource utilization, minimum makespan and minimizes delay time in the cloud.

Step 1: Create $V_M = V_{M1}, V_{M2}, V_{M3}, ...,V_j$ into a set of resources.
Step 2: Register the number of virtual machines on the Cloud Information System (CIS)
Step 3: Sort jobs $B = B_1, B_2, B_3, ..., B_i$ in ascending order
Step 4: Read the list of available virtual machines $V_M = V_{M1}, V_{M2}, V_{M3}, ..., V_j$
Step 5: Schedule ordered jobs $B = B_1, B_2, B_3, ..., B_i$, in the logical view (the number of machines in the logical view is equal to the number of virtual machines created).
Step 6: Inform the user that the job was accepted and will be executed.
Step 7: Schedule and execute job $B = B_1, B_2, B_3, ..., B_i$ on the available virtual machines.
Step 8: Mark or delete jobs that have finished executing on the virtual machine and tell the scheduler that the jobs have finished executing.
Step 9: Inform the user that the work has been completed.

Notation Explanation:
1. Time of the earliest start time of the job (t_{wet}): the fastest execution time of a job.
2. Start time to execute the job (t_{md}): the time a job starts to be executed.
3. Completion time to execute the job (t_{asa}): earliest execution time until the end of job execution
4. The end time to execute the job (t_{pa}): the latest execution start time.
5. Execution time of the job (t_e): execution time
6. Relaxed time (ts): the difference between the actual execution start time and the earliest execution start time.

Algorithm: Job scheduling algorithm
Input: Job (jobId, t_{asa}, t_{pa}, te, $numJob$)
Output: RIT, AWT, Makespan
1 Begin()
2 // Declaration and Initialization; Virtual logical (V_L)
3 update(Table(Vm))
4 CIS←Register Vm
5 B_i← Sort jobs
6 Read Vm
7 // Schedule ordered jobs B_i
8 Free←(t_{pa}- t_{asa})
9 Note ← false;
10 If (!Note) then
11 start ← t_{wet};
12 finish ← t_{wet} + t_e ;
13 flex ← start - t_{wet};
14 while(!Note AND (t_{pa} - t_{asa})<=Free)
10 min ← minR(start, finish);
11 If (min > 0) then
12 allocV_L (Id, jobId, t_{wet}, start, t_{pa}, t_e);
13 suk ← true;
14 else
15 start ← time + 1;
16 finish ← start + t_e - 1;
17 flex = start - t_{wet};
18 End
19 End while
20 End
21 RIT ←$Finish_{previous} - start_{current}$ // calculate RIT
22 TotalRIT← $\sum_{i-0}^{Size} RIT$ //calculate the total RIT
23 Makespan ← $max_{i \in job\ i}$ F_i
24 WT ← $Start_{reser} - Start_{new}$
25 //job accepted
26 V_M ← V_L // execute job V_L on the virtual machine

Fig. 2. Resource Allocation Cloud Algorithm

Lines 2 to 9 describe the declaration and initialization of the values used in the algorithm. Lines 10 to 13, calculation of the value of the job sent by the user. Lines 14 to 19, the loop is used to search for unused space or empty space in the virtual view. If there is free or unused space, the job will be allocated. Lines 21 to 25, the idle time, waiting time and makespan values will be calculated. Line 26 of the job will be executed on the virtual machine, according to the existing virtual view.

2.3 Performance Metrics

Scheduling is a list of tasks that determines how competing tasks access one or more reusable resources. These resources can be hardware, such as processors, communications lines, storage devices, or software. Task scheduling is the assignment of a group of tasks to certain resources by starting and ending task times with certain limits. Task scheduling is an integrated part of cloud computing. The purpose of task scheduling is to allocate resources for task implementation. Task scheduling guides resource allocation because there are many nodes on which tasks can run, the problem is how to assign tasks to those resources. This assignment is known as task allocation to the resources scheduled by the scheduler. Performance metrics are used to measure certain attributes in a proposed or used scheduling algorithm.

Resource Idle Time (RIT)

A resource may not be usable even if a reservation request is available[21][22][23]. Delay times occur because scheduling policies do not match the allocation of reservation requests. RIT is calculated by applying the formula below.

$RIT = Finish_{previous} - start_{current}$ (1)

when there is a reservation request with a conflict. The total resource idle time is calculated by the following equation:

$$Total\ RIT = \sum_{i-0}^{Size} RIT \quad (2)$$

Makespan

Makespan: Last job completion time. Users want to shorten their application completion time[24][25][26][27][28].

$Makespan = max_{i\ \epsilon\ job\ i}\ F_i$ (3)

where F_i indicates the completion time of job i:

Waiting Time

Sometimes a resource is not available at the time a reservation requires it, but the resource can be booked at a different time[29][30]. The difference between the expected start time and the actual start time is the waiting time as shown in equation 3.

$$Waiting\ Time(WT) = Start_{reser} - Start_{new} \quad (4)$$

Total Waiting Time (TWT) is the total waiting time in a timeslot, shown by equation 4.

$$Waiting\ Time(WT) = Start_{reser} - Start_{new} \quad (5)$$

The size value refers to the reservation length of a particular timeslot. So the Average Waiting Time is shown by equation 5.

$$Average\ Waiting\ Time(AWT) = \frac{TWT}{No\ reservasi} \quad (6)$$

2.4. Workload

Configure the entities used in the simulation environment. Randomly generated workloads with different job sizes from 100 to 800. The number of virtual machines used is 30. The number of data centers is 1, with the number of hosts being 30. The scheduler is space shared, which only allows one job to run at a certain time within the resource certain. Sets of executed jobs are independent of each other.

3. Results and Discussion

The device uses Java Developer, Windows 11 operating system, 11th Gen Intel(R) Core(TM) CPU i3-1115G4 @ 3.00GHz 3.00 GHz. The backfilling approach is proposed as a comparison, because this strategy shifts reservations early which will make room for new reservations to be allocated. Viewed from the other side, the next job must wait in the waiting room queue, until the previous job has finished executing, so there is no certainty about the time the job will be executed. Therefore, resource usage may be inefficient and jobs have to wait for quite a long period of time. The leading job queue will wait if the required time is greater than the required computing resource time. Backfilling allows jobs that have execution times smaller than the execution times of jobs at the front of the queue to move forward and execute on idle computing resources. The delay time, waiting time and job waiting time matrices are used as performance comparisons, so that resource use becomes more efficient.

Experiments are carried out to represent a realistic cloud scheduling environment, considering different computing scenarios, the parameters to be observed are resource utilization and job waiting time. The parameters used in the experiment are shown in Table 1. To measure delay time, job waiting time and makespan, the FCFS-Slotfree method will be compared with FCFS, backfilling.

Table 1 Job Experiment Parameters

Parameter name	Parameter value
Job execution time duration	Fixed
The number of resources the job requires	Fixed

Execution start time	Changed
Execution end time	Changed

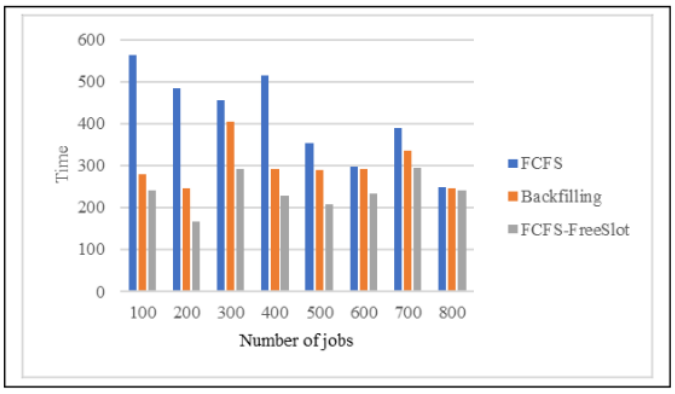

Fig. 3. Idle time results with different workloads.

Figure 3 shows the idle time generated by each algorithm for workloads of different sizes. This shows that the proposed algorithm shows significant improvement in idle time. If we look at the percentage, it can be seen that the proposed algorithm produces better idle time than other algorithms. We can observe that the average idle time for the proposed method is 25.3%, FCFS is 43.1% while for bacfilling it is 31.5%. In general, it can be observed that the proposed idle time is smaller than that of FCFS and backfilling. Implementing advance reservation in the proposed scheduling system increases resource utilization by 17.8% for FCFS and 6.27% for backfilling. This is caused by fragmentation or idle time. The proposed strategy uses FCFS-Slotfree to schedule common job deadlines that can minimize the initial idle period. Regardless of the size of the initial and final period of unemployment. The results show that FCFS-Slotfree provides the best system utilization compared to other strategies. FCFS-Slotfree provides a better allocation policy, according to reservation requests.

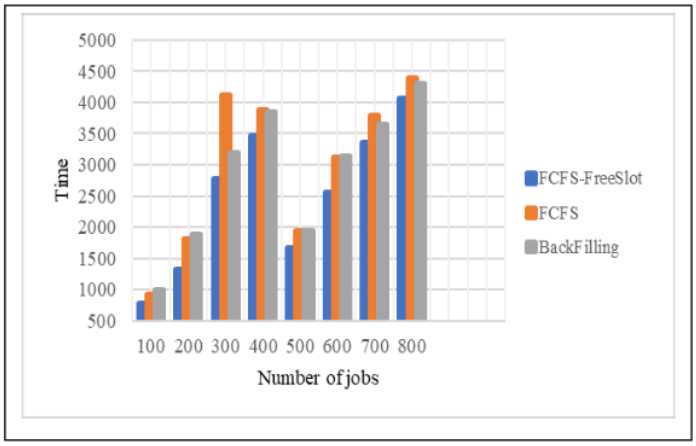

Fig. 4. Makespan results with different workload traces.

Figure 4 shows the makespan produced by each algorithm for workloads of different sizes. This shows that the proposed algorithm can reduce the average makespan value significantly. If we look at the percentages, it can be seen that the proposed algorithm produces better makespan than other algorithms. We can observe that the average makespan reduction for FCFS is 16.73%, while for bacfilling it is 12.87%. This is because the execution delay time value of the proposed method is

smaller compared to the FCFS and backfilling methods. The proposed method can place jobs early, when they are about to be executed.

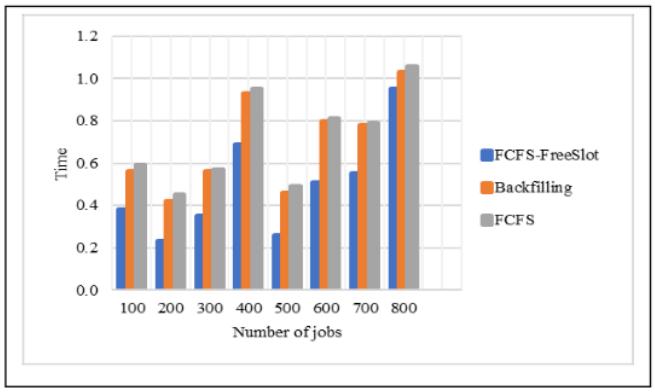

Fig. 5. Average waiting time results with different workloads.

Figure 3 shows the AWT generated by each algorithm for workloads of different sizes. This shows that the proposed algorithm shows significant improvement in AWT. If we look at the percentages, it can be seen that the proposed algorithm produces better AWT than other algorithms. We can observe that the average AWT reduction for FCFS is 13.3% while for bacfilling it is 12.03%. In general, it can be observed that there is a significant improvement in AWT in the proposed algorithm. The leading job queue will wait if the required time is greater than the required compute node time. Backfilling allows jobs that have an execution time smaller than the execution time of jobs in the front queue to move forward and execute on idle compute nodes. In the backfilling algorithm, the next job waits in the waiting room queue until the previous job has finished executing, so there is no certainty when the job will be executed.

4. Conclusion

Job scheduling algorithms have the aim of providing better quality of services such as delay time, wait time, cloud wait time, etc. Job scheduling simulation has been carried out using the proposed algorithm. Based on the simulation results, it is known that the proposed algorithm can work well. The proposed algorithm compared with well-known algorithms such as FCFS and backfilling, has better performance. The algorithms have been compared considering job sets of different sizes. After comparison, it can be seen that FCFS-Slotfree produces smaller delay, waiting time and makespan values than FCFS and backfilling. All the work is completed in a shorter duration. This shows that in cloud computing, the proposed algorithm shows a better scheduling policy.

Declarations

Author contribution. All authors contributed equally as the main contributor to this paper. All authors read and approved the final paper..

Funding statement. This research received a research grant from the Ministry of Research, Technology and Higher Education (Ristekdikti) of the Republic of Indonesia with contract number 001/PFR/LPPM UAD/VI/2023.

Conflict of interest. The authors declare no conflict of interest.

Additional information. No additional information is available for this paper.

References

[1] F. Alhaidari, T. Balharith, and E. Al-Yahyan, "Comparative analysis for task scheduling algorithms on cloud computing," *2019 Int. Conf. Comput. Inf. Sci. ICCIS 2019*, pp. 1–6, 2019, doi: 10.1109/ICCISci.2019.8716470.

[2] M. Ibrahim *et al.*, "A Comparative Analysis of Task Scheduling Approaches in Cloud Computing," *Proc. - 20th IEEE/ACM Int. Symp. Clust. Cloud Internet Comput. CCGRID 2020*, pp. 681–684,

2020, doi: 10.1109/CCGrid49817.2020.00-23.

[3] Y. Yao, J. Cao, S. Qian, and X. Wang, "Resource Scheduling for Real-Time Analytical Workflow Services in the Cloud," *IEEE Access*, vol. 6, pp. 57910–57922, 2018, doi: 10.1109/ACCESS.2018.2871827.

[4] K. Braiki and H. Youssef, "Resource management in cloud data centers: A survey," *2019 15th Int. Wirel. Commun. Mob. Comput. Conf. IWCMC 2019*, pp. 1007–1012, 2019, doi: 10.1109/IWCMC.2019.8766736.

[5] F. Nzanywayingoma and Y. Yang, "A Literature Survey on Resource Management Techniques, Issues and Challenges in Cloud Computing," *Telkomnika (Telecommunication Comput. Electron. Control.*, vol. 15, no. 4, pp. 1917–1933, 2017, doi: 10.12928/TELKOMNIKA.v15i4.6574.

[6] A. V. Karthick, E. Ramaraj, and R. Kannan, "An efficient tri queue job scheduling using dynamic quantum time for cloud environment," *Proc. 2013 Int. Conf. Green Comput. Commun. Conserv. Energy, ICGCE 2013*, pp. 871–876, 2013, doi: 10.1109/ICGCE.2013.6823557.

[7] A. V. Karthick, E. Ramaraj, and R. Kannan, "An efficient tri queue job scheduling using dynamic quantum time for cloud environment," *Proc. 2013 Int. Conf. Green Comput. Commun. Conserv. Energy, ICGCE 2013*, no. March, pp. 871–876, 2013, doi: 10.1109/ICGCE.2013.6823557.

[8] S. H. H. Madni, M. S. A. Latiff, Y. Coulibaly, and S. M. Abdulhamid, "Recent advancements in resource allocation techniques for cloud computing environment: a systematic review," *Cluster Comput.*, vol. 20, no. 3, pp. 2489–2533, 2017, doi: 10.1007/s10586-016-0684-4.

[9] N. Arora, "Review on Task Scheduling Algorithms in Cloud Computing Environment," *Int. J.*, vol. 8, no. 4, pp. 462–468, 2017.

[10] A. A. Nayak and S. Shetty, "A Systematic Analysis on Task Scheduling Algorithms for Resource Allocation of Virtual Machines on Cloud Computing Environments," *ICRTEC 2023 - Proc. IEEE Int. Conf. Recent Trends Electron. Commun. Upcom. Technol. Smart Syst.*, pp. 1–6, 2023, doi: 10.1109/ICRTEC56977.2023.10111894.

[11] K. Pradeep, N. Gobalakrishnan, N. Manikandan, L. J. Ali, K. Parkavi, and K. P. Vijayakumar, "A Review on Task Scheduling using Optimization Algorithm in Clouds," *Proc. 5th Int. Conf. Trends Electron. Informatics, ICOEI 2021*, pp. 935–938, 2021, doi: 10.1109/ICOEI51242.2021.9452837.

[12] S. Singh and I. Chana, "QoS-aware autonomic resource management in cloud computing: A systematic review," *ACM Comput. Surv.*, vol. 48, no. 3, 2015, doi: 10.1145/2843889.

[13] D. Tsafrir, Y. Etsion, and D. G. Feitelson, "Backfilling using system-generated predictions rather than user runtime estimates," *IEEE Trans. Parallel Distrib. Syst.*, vol. 18, no. 6, pp. 789–803, 2007, doi: 10.1109/TPDS.2007.70606.

[14] N. Jayapandian, "Parallel queue scheduling in Dynamic Cloud environment using Backfilling algorithm," *Int. J. Intell. Eng. Syst.*, vol. 11, no. 2, pp. 39–48, 2018, doi: 10.22266/ijies2018.0430.05.

[15] S. Singh and I. Chana, "Cloud resource provisioning: survey, status and future research directions," *Knowl. Inf. Syst.*, vol. 49, no. 3, pp. 1005–1069, 2016, doi: 10.1007/s10115-016-0922-3.

[16] S. Singh and I. Chana, "A Survey on Resource Scheduling in Cloud Computing: Issues and Challenges," *J. Grid Comput.*, vol. 14, no. 2, pp. 217–264, 2016, doi: 10.1007/s10723-015-9359-2.

[17] R. Istrate, A. Poenaru, and F. Pop, "Advance reservation system for datacenters," *Proc. - Int. Conf. Adv. Inf. Netw. Appl. AINA*, vol. 2016-May, pp. 637–644, 2016, doi: 10.1109/AINA.2016.106.

[18] J. Chamberlain, E. Simhon, and D. Starobinski, "Preemptible queues with advance reservations : Strategic behavior and revenue management," vol. 293, pp. 561–578, 2021.

[19] K. P. N. Jayasena and B. S. Thisarasinghe, "Optimized task scheduling on fog computing environment using meta heuristic algorithms," *Proc. - 4th IEEE Int. Conf. Smart Cloud, SmartCloud 2019 3rd Int. Symp. Reinf. Learn. ISRL 2019*, pp. 53–58, 2019, doi: 10.1109/SmartCloud.2019.00019.

[20] P. Mallik, A. K. Nayak, and R. Kumar Dalei, "Comparative Analysis of Various Task Scheduling Algorithms in Cloud Environment," *2021 19th OITS Int. Conf. Inf. Technol.*, pp. 37–41, 2022, doi:

10.1109/ocit53463.2021.00019.

[21] S. Shin, Y. Kim, and S. Lee, "Deadline-guaranteed scheduling algorithm with improved resource utilization for cloud computing," *2015 12th Annu. IEEE Consum. Commun. Netw. Conf. CCNC 2015*, pp. 814–819, 2015, doi: 10.1109/CCNC.2015.7158082.

[22] E. Hosseini, M. Nickray, and S. Ghanbari, "Optimized task scheduling for cost-latency trade-off in mobile fog computing using fuzzy analytical hierarchy process," *Comput. Networks*, vol. 206, no. October 2021, 2022, doi: 10.1016/j.comnet.2021.108752.

[23] X. Liu, C. Wang, X. Qiu, B. B. Zhou, B. Chen, and A. Y. Zomaya, "Backfilling under two-tier virtual machines," *Proc. - 2012 IEEE Int. Conf. Clust. Comput. Clust. 2012*, pp. 514–522, 2012, doi: 10.1109/CLUSTER.2012.36.

[24] N. Chitgar, H. Jazayeriy, and M. Rabiei, "DSCTS: Dynamic Stochastic Cloud Task Scheduling," *5th Iran. Conf. Signal Process. Intell. Syst. ICSPIS 2019*, no. December, pp. 18–19, 2019, doi: 10.1109/ICSPIS48872.2019.9066063.

[25] N. Sasikaladevi, "Minimum makespan task scheduling algorithm in cloud computing," *Int. J. Grid Distrib. Comput.*, vol. 9, no. 11, pp. 61–70, 2016, doi: 10.14257/ijgdc.2016.9.11.05.

[26] S. Mittal and A. Katal, "An Optimized Task Scheduling Algorithm in Cloud Computing," *Proc. - 6th Int. Adv. Comput. Conf. IACC 2016*, pp. 197–202, 2016, doi: 10.1109/IACC.2016.45.

[27] M. T. Alam Siddique, S. Sharmin, and T. Ahammad, "Performance Analysis and Comparison among Different Task Scheduling Algorithms in Cloud Computing," *2020 2nd Int. Conf. Sustain. Technol. Ind. 4.0, STI 2020*, vol. 0, pp. 19–20, 2020, doi: 10.1109/STI50764.2020.9350466.

[28] K. M. S. U. Bandaranayake, K. P. N. Jayasena, and B. T. G. S. Kumara, "An Efficient Task Scheduling Algorithm using Total Resource Execution Time Aware Algorithm in Cloud Computing," *Proc. - 2020 IEEE Int. Conf. Smart Cloud, SmartCloud 2020*, pp. 29–34, 2020, doi: 10.1109/SmartCloud49737.2020.00015.

[29] R. K. R. Indukuri, S. V. Penmasta, M. V. R. Sundari, and G. J. Moses, "Performance Evaluation of Deadline Aware Multi-stage Scheduling in Cloud Computing," *Proc. - 6th Int. Adv. Comput. Conf. IACC 2016*, pp. 229–234, 2016, doi: 10.1109/IACC.2016.51.

[30] P. Y. Zhang and M. C. Zhou, "Dynamic Cloud Task Scheduling Based on a Two-Stage Strategy," *IEEE Trans. Autom. Sci. Eng.*, vol. 15, no. 2, pp. 772–783, 2018, doi: 10.1109/TASE.2017.2693688.

HASIL CEK_IJAIN_Template

ORIGINALITY REPORT

8%	4%	7%	1%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Pallavi V K, R Trupthi, Varnitha G, Keerthan Kumar T G. "Improvised Threshold Based Task Scheduling", 2021 Second International Conference on Electronics and Sustainable Communication Systems (ICESC), 2021 Publication	1%
2	Sukhpal Singh, Inderveer Chana. "A Survey on Resource Scheduling in Cloud Computing: Issues and Challenges", Journal of Grid Computing, 2016 Publication	1%
3	pubs2.ascee.org Internet Source	1%
4	Submitted to Universitas Pendidikan Indonesia Student Paper	1%
5	www.researchgate.net Internet Source	1%
6	tudr.thapar.edu:8080 Internet Source	<1%

7	Yamin Thet Htar Hlaing, Tin Tin Yee. "Static Independent Task Scheduling on Virtualized Servers in Cloud Computing Environment", 2019 International Conference on Advanced Information Technologies (ICAIT), 2019 Publication	<1 %
8	Fahd Alhaidari, Taghreed Balharith, Eyman AL-Yahyan. "Comparative Analysis for Task Scheduling Algorithms on Cloud Computing", 2019 International Conference on Computer and Information Sciences (ICCIS), 2019 Publication	<1 %
9	Sukhpal Singh, Inderveer Chana. "Cloud resource provisioning: survey, status and future research directions", Knowledge and Information Systems, 2016 Publication	<1 %
10	drops.dagstuhl.de Internet Source	<1 %
11	Fereshteh Sheikholeslami, Nima Jafari Navimipour. "Auction-based resource allocation mechanisms in the cloud environments: A review of the literature and reflection on future challenges", Concurrency and Computation: Practice and Experience, 2018 Publication	<1 %

12	iaeme.com Internet Source	<1%
13	repo.library.stonybrook.edu Internet Source	<1%
14	www.hindawi.com Internet Source	<1%
15	Krishnanand Rai, Satish Vemireddy, Rashmi Ranjan Rout. "Fuzzy Logic based Task Scheduling Algorithm in Vehicular Fog Computing Framework", 2021 IEEE 18th India Council International Conference (INDICON), 2021 Publication	<1%
16	Lina Ni, Jinquan Zhang, Changjun Jiang, Chungang Yan, Kan Yu. "Resource Allocation Strategy in Fog Computing Based on Priced Timed Petri Nets", IEEE Internet of Things Journal, 2017 Publication	<1%
17	Swati Lipsa, Ranjan Kumar Dash, Nikola Ivković, Korhan Cengiz. "Task Scheduling in Cloud Computing: A Priority-Based Heuristic Approach", IEEE Access, 2023 Publication	<1%
18	businessdocbox.com Internet Source	<1%

19	dyuthi.cusat.ac.in Internet Source	<1%
20	www.slideshare.net Internet Source	<1%

Exclude quotes	On	Exclude matches	Off
Exclude bibliography	On		