khazanah informatika

Jurnal Ilmu Komputer dan Informatika

Job Scheduling on Grid Computing Using First Fit, Best Fit, and Worst Fit

Ardi Pujiyanta*, Fiftin Novianto
Informatics Study Program
Universitas Ahmad Dahlan, Yogyakarta
*ardipujiyanta@tif.uad.ac.id

Abstract-Grid computing can be considered as a large-scale distributed cluster computing and parallel distributed network processing. The two most important issues in managing user works are resource allocation and scheduling of required resources. When user jobs are submitted, they are managed by resource intermediaries which find and allocate the right resources. After the resource allocation stage, work is scheduled on the existing resources according to the user's required resources. In most grid systems with traditional scheduling, jobs are submitted and placed in waiting room queues to wait for the required resources to become available. Each grid system can use a different scheduling algorithm to execute jobs based on other parameters, such as resources, delivery time, and execution duration. There is no guarantee that these traditional scheduling algorithms will get the job done. The First Come First Serve Left Right Hole Scheduling (FCFS-LRH) reservation strategy improves resource utilization in a grid system by using a local scheduler, compared to traditional strategies. There are two objectives of this research. First, comparing the first fit, best fit, and worst fit algorithms to find empty timeslots and place them in a virtual view. Second, reducing the idle time value. The results showed that the FCFS-LRH method could reduce the idle time value of the FCFS-EDF and FCFS methods. The overall execution time of the first fit with the FCFS-LRH strategy is better than the FCFS-EDF.

Key Word: Grid computing, Scheduling, FCFS-LRH, FCFS-EDF

Article info: *submitted January 8, 2022, revised May 17, 2022, accepted May 27, 2022*

1. Introduction

In general, a grid computing system is used to increase the utilization of homogeneous or heterogeneous resources so that workload management will be optimal [1][2][3][4]. Computing resources facilitate organization formation such as servers, network nodes, storage elements [5]. Resources clustered together will result in a robust computing environment. Grid computing allows independent users and organizations to utilize untapped CPU cycles, such as databases, scientific tools, and storage elements. Millions of computer systems will be interconnected, placed on a global network with minimal access costs[6]. Grid computing is similar to Power Gridlines, as in power company operations. The grid system model provides the sharing of data and computing resources regardless of the location and origin of the resources. Grid users will submit their work to the Grid operating system via an interface. Then, the Grid system decides and finds computing resources that can serve the user's needs.[7]. Complete research on Grid done in the reference [8][9][10][11].

Grid computing is a promising next-generation science, engineering, and research problem-solving technology. Grid computing differs from conventional distributed computing in that it focuses on large-scale resources, sharing innovative applications. Grid computing is a problem-solving environment that leverages unused resources and maximizes resource capability. Grid computing uses an innovative approach in leveraging existing information technology infrastructure to optimize computing resources in managing data and computing workloads [12][13]. The grid computing platform enables the sharing, selection, and combination of geographically distributed heterogeneous resources (data sources and computers), belonging to different managerial organizations (virtual organizations) to answer large-scale engineering, commerce, and science problems.[14][15][16]. The primary purpose of parallel computers is to overcome the single processor speed blockage [17]. There

are three approaches to creating parallel applications. The first approach is based on automatic parallelization, with this approach, the programmer does not have to worry about parallelizing jobs. The second approach is based on the use of parallel libraries. This approach has the same parallel code for multiple applications placed in the parallel library. The third approach is re-coding or writing code from scratch in making parallel applications. Programmers are free to choose the language and programming model used to create similar applications[18].

Jobs from users are submitted and managed by a resource broker who must find and allocate the right resources for the job. After the resource allocation stage, the work must be scheduled on the existing resources according to the user's required resources, in most of the grid systems with traditional scheduling, the work is submitted and placed in a waiting room queue to wait for the required resources to become available. Each grid system can use a different scheduling algorithm to execute jobs based on different parameters, such as the number of resources, delivery time, and execution duration. With this traditional scheduling algorithm(FCFS), there is no guarantee the job will be executed. FCFS-EDS is proposed to provide guaranteed jobs executed on grid computing[19]. First fit algorithm is used by FCFS-EDS to place jobs in empty spaces in virtual views. Once the job is placed in the virtual view, the user will be notified that the job has been accepted. The job to be executed will be mapped to the physical view. The weakness of FCFS-EDS is that user-submitted jobs are not placed on the left side of the virtual view used. By not placing a job on the left side of the virtual view, it is suspected that it can cause a high delay. The reservation strategy First Come First Serve Left Right Hole Scheduling (FCFS-LRH)[20] is proposed to improve resource utilization in the grid system. Job requests are sent based on number of jobs, initial start time, execution time. Incoming user requests will be sorted by priority of execution start time, execution time, and required amount of resources. The accepted job will be placed in the virtual view and sent to the physical view when it is executed. The purpose of this study: first, first fit, best fit, and worst fit algorithms will be used on FCFS-LRH then compared with FCFS-EDS first fit. Which algorithm has the best timing when the job is placed in the virtual view. Second, comparing the FCFS-LRH method with FCFS-EDS and FCFS, can the FCFS-LRH method reduce idle time?.

2. Methods

The steps in this research are as follows: first, determine the tools used in the study. Second, determine the amount of data to be used obtained from randomly generated data. Generate data using usability factors 2 and 3 and flexibility values from 25% to 100%. The three data generated results will use as input to the first fit, best fit, and worst fit algorithms to get the execution time value for the virtual node.

a. Tools and materials

1) Tools

Hardware and software requirements needed to run the simulation and test the proposed reservation scheduling algorithm in this study:

Hardware

a. Prosesor : Amd A 10-5750 M APU 2.50 GHz
b. *Ram* : 16 GB.
c. *Disk drive* : 320 GB.
d. *Display* : 12" *Wide-screen*.

Software

a. Operating Systems windows 8 64 bit.
b. *Eclipse Kepler Build id*:20130614-0229AppServ v2.5.8 : *Web Server*.

2) Materials

The data collection method used is a literature study method that refers to research data [19][21][22][23]. Figure 1 shows the use of a workload generator. The user submits a job description (1) based on the user's job description and grid description information, which will be used as input to the workload generator (2). The output of the workload generator is then submitted or sent back to the grid (3). The network environment is responsible for carrying out the work, returning the user output (4), and generating a detailed work report. The user processes all the results in a post-production step (5).

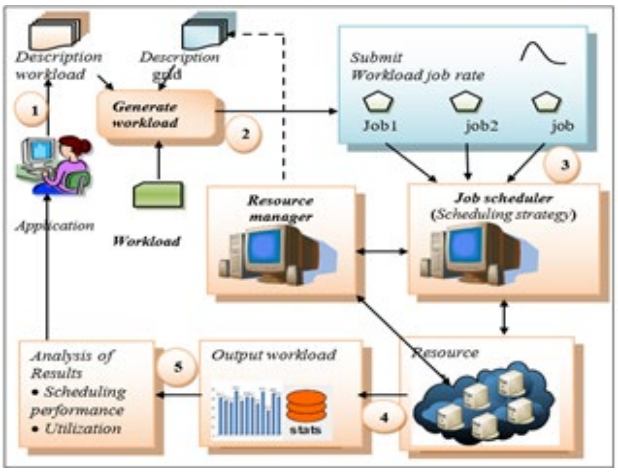

Figure 1. Generate Workload Process Model on Grid Computing [24]

b. Workload Generator

The scheduling performance proposed in this study checked using data generated from the workload generator. The workload generator output used as input to the proposed reservation scheduling. Characteristics of the workload generator in this study[19][21][22][23]:

1) The arrival rate of incoming jobs follows the Poisson distribution [21].
2) The execution period of each reservation request is uniformly distributed.

3) The earliest start time of each reservation is uniformly distributed.
4) The flexible reservation percentage is randomly selected.
5) Relax time range for each flexible reservation is uniformly distributed.
6) The required amount of resources is uniformly distributed.
7) The width of the timeslot in this study is 5 minutes [2].
8) The number of jobs generated is 800.

c. Method FCFS-LRH

An empty timeslot will be found in the virtual view when a user submits a job to the grid. If an empty timeslot is found, the job will be allocated to the virtual view. The user will be notified that the job is accepted. If no empty timeslot is found the job will be rejected. Contains a description of how to carry out research. Figure 2. below shows the parameters used by the FCFS-LRH Method. User will submit (jobId, $tesr$, $tlsr$, te, $numCN$). The function of each parameter can be explained as follows:

jobId : job number.
tes : earliest start time the job can executed.
tls : the last start time the job can executed
te : Job execution time
$numCN$: The number of resources needed by the job.
tf : Total time flexibility
$tr1, tr2$: left and right side flexibility time.

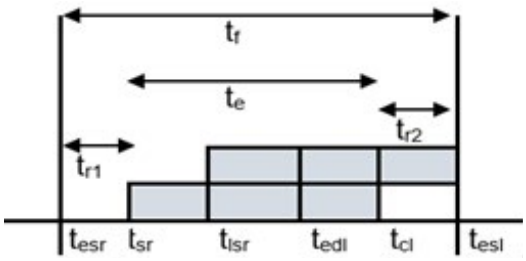

Figure 2. Job scheduling allocation.

d. Virtual view and Physical View

All jobs sent to the grid will first find a place in the virtual view, whether there is an empty slot or not[25]. If an empty slot found, the job will be placed in the virtual view. The user will be notified that the job will executed. Figure 2 an example of randomly placing 10 jobs placed on 6 virtual view resources. Figure 3 shows job placement in physical view after recombination.

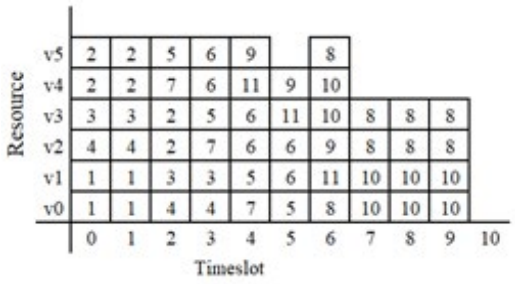

Figure 3 job scheduling in virtual view

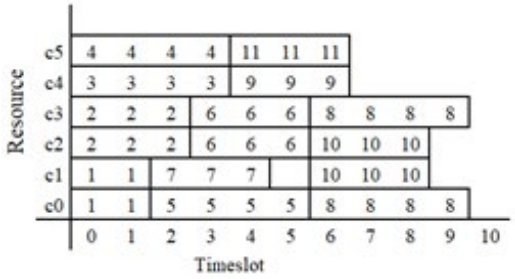

Figure 4 job scheduling in physical view

e. Performance Metrics FCFS-LRH

The resource may be idle despite a reservation request. This occurs when the idle time does not match the allocation policy. RIT is calculated by applying the formula below.

RIT = Finishprevious – startcurrent

When there is a reservation request with a conflict. The following equation calculates the total idle time of the resource.

$$\text{Total RIT}(T_{RIT}) = \sum_{i=1}^{size} RIT$$

f. Algoritme FCFS-LRH

Input: Job (jobId, $tesr$,lsr,te, $numCN$)
Output: IdleTime

1. For j=0:numSlot
2. sort arrival jobs based on priority $tesr$, te, $numCN$
3. Endfor
4. For i=0:numSlot // *numSlot is the amount of job/ timeslot*
5. calculate the value of d2=$tesr$+te-1
6. Search timeslot free with First fit,Best fit, Worst fit strategy
7. IF (timeslot==0) then insert Jobid value
8. IF (timeslot!=0) then execution procedure moveSlot().
9. Endfor
10. Procedure moveSlot();
11. Initialization; finish=0,suc=false, start=$tesr$, finish=$tesr$+te-1.
12. relax=start–$tesr$, tr=$tlsr$–$tesr$,CNs=0.
13. while (!suc and relax <=tr)
14. For cek=start:finish
15. set the variable CNs=0

16. For s=0:atrans.size()
17. IF atrans.get(s,cek)!=0 then
18. variable CNs increases by 1
19. Endif
20. Endfor
21. calculate the variable sel=maxC-CNs // *maxC is the number of physical nodes*
22. IF (sel>=CN) then
23. calculate the variable t=start, suc=true
24. Else
25. calculate the variable t=cek, finish=start+te-1, relax=start-$tesr$, suc=true
26. IF (start>=$tlsr$) then continuous to line 4
27. Endif
28. Endfor
29. Endwhile
30. IF (suc==true) then
31. calculate the variable start=t+1, finish=start+te-1, relax=start-$tesr$
32. insert JobID with the first fit, best fit, worst fit strategy
33. calculate IdleTime
34. Endif

The explanation of the FCFS-LRH algorithm is as follows: user submits Job (jobId, ,lsr,te, N). Lines 1-3 show the sorting of jobs by priority. Lines 5-6 look for empty timeslots in the virtual view using First fit,Best fit, Worst fit. If there is an empty timeslot do line 7. If there is no empty timeslot move the job, shown in line 8 and call the moveslot procedure. The function of the moveslot procedure on lines 11-34 is to shift the job if there is an empty timeslot. If the job can be shifted then allocate the job to the timeslot and calculate the idle time.

3. Result

Experimental stages in this study: (1) Setting parameters whose values fixed and changes shown in table 1; (2) Setting the flexibility parameters and usability factors, are shown in table 2; (3) Generating jobs randomly with usability factors 2 and 3, and determining the percentage of flexibility from 25% to 100%, is shown in table 3. The results of the job generation in table 3 used as input to the FCFS-LRH, FCFS-EDS method. (4) The results of data input processing were tested using the FCFS-LRH, FCFS-EDS methods with the first fit, best fit, and worst fit strategies. (5) The first fit, best fit and worst fit strategy with the best value will be used to find the idle time value in the FCFS, FCFS-EDF and FCFS-LRH methods.

The user sends his work to the resource in the form of JobId, execution start time, execution time, execution end time and the number of resource nodes needed. The FCFS-LRH strategy will respond by finding an empty slot in the virtual view. If an empty slot is found, the job will be allocated to the virtual view, and the user will be notified that the job has been accepted. If no vacant slot found, the job will be rejected. Table 4, Figure 5, shows the search time and job allocation using FCFS-LRH and FCFS-EDS with the first fit, best fit, and worst fit strategies. Table 4, figure 5 uses the flexibility of 25% to 100%; Utilization factor =2 and =3 ; the number of jobs is between 300 and 795. The average result of the search time and job placement in the virtual view for the FCFS-LRH method with the first fit algorithm is 146.61; the best fit of 153.21; the worst fit is 150.66. The search and job allocation results using FCFS-EDS with a first fit obtained 181.30. These results show that the average job search and placement time in virtual view first fit is faster than best fit and worst fit. Figure 5 shows that using =2 and =3 the average search time and job allocation in the FCFS-LRH virtual view with first fit is faster than FCFS-EDS with first fit. These results indicate that FCFS-LRH notifications to users are better than FCFS-EDS.

Figure 6 compares idle time between FCFS-LRH with FCFS and FCFS-EDF. FCFS-LRH average idle time is lower than FCFS and FCFS-EDF.

Table 5 shows that with the utilization factor of 2, the idle time value of FCFS-LRH is lower than FCFS and FCFS-EDF. Likewise, for the utilization factor 3, the idle time value of FCFS-LRH is lower than FCFS and FCFS-EDF.

Table 1 Jobs Experiment Parameters

Parameter name	Nilai parameter
Job execution time	constant
Amount of resources required	constant
Flexibility time	Changed
Execution start time	Changed
Execution end time	Changed

Table 2 Parameters of Utilization Factors and Percent Flexibility

Load	μ	Percent flexibility (%)
Small	μ=2	25, 50, 75, 100
Moderate	μ=3	25, 50, 75, 100

Table 3 Generate jobs

Factor utilization(μ)	Flexibility(β) (%)	Number of jobs
2	25	383
2	50	421
2	75	459
2	100	553
3	25	627
3	50	711
3	75	764
3	100	795

Table 4. Comparison of the execution time of first fit, best fit and worst fit

Number of jobs	μ	β(%)	FCFS-LRH first fit	FCFS-LRH best fit	FCFS-LRH worst fit	FCFS-EDS first fit
383	2	25	99.73	102.58	98.95	130.19
421	2	50	116.90	128.21	123.13	151.66
459	2	75	128.08	132.49	132.02	156.65
553	2	100	142.47	148.18	145.94	174.32
627	3	25	152.66	157.3	158.53	187.23
711	3	50	169.38	175.83	174.57	207.79
764	3	75	177.56	186.71	181.49	214.19
795	3	100	186.11	194.36	190.61	228.35
Average			146,61	153,21	150,66	181.30

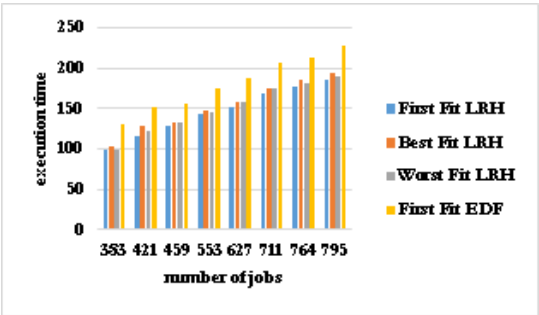

Figure 5 Comparison of Execution Time of First Fit, Best Fit, Worst Fit Based on Number of Jobs

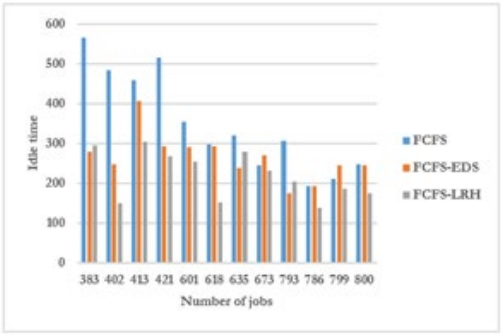

Figure 6 FCFS-LRH idle time comparison with FCFS-EDS and FCFS

Table 5. average idle time based on utilization factor

Method	μ=2	μ=3
FCFS	505,25	303,8
FCFS-EDS	305,75	272,3
FCFS-LRH	254,25	228,8

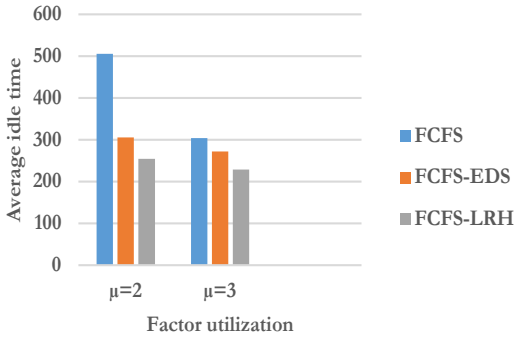

Figure 7. **FCFS-LRH average idle time comparison with FCFS-EDS and FCFS**

4. Discussion

Reservation of resources in advance ensures the availability of resources when needed, increases the efficient utilization of resources, and reduces the execution time of a process. There are various approaches. There is no guarantee that most of the conventional methods will execute the work because the result is placed in a waiting room. e.g. the FCFS approach. The FCFS-LRH approach proposes that users can be sure that their work will be executed and reduce idle time.

Based on the research results, the use of first fit is better than best fit and worst fit when used in the FCFS-LRH method. The FCFS-LRH method using first fit is faster than FCFS-EDF, which results in faster notifications to users. The experimental results on the FCFS-LRH method using a usability factor of 2 and a flexibility of 25% to 100% resulted in a reduction in the idle time value of FCFS-LRH compared to FCFS of 49.68%. Meanwhile, when compared with FCFS-EDF, the idle time reduction of FCFS-LRH is 16.84%. If you use a benefit factor of 3 and flexibility of 25% to 100%, the result is a reduction in the idle time value of FCFS-LRH compared to FCFS of 24.69%.

Meanwhile, when compared with FCFS-EDF, the idle time reduction of FCFS-LRH is 15.98%. The average

idle time reduction of FCFS-LRH compared to FCFS is 40.3%. The average idle time reduction of FCFS-LRH compared to FCFS-EDF is 16.44%. The FCFS-LRH method can reduce the idle time value due to the job scheduling policy by sorting incoming jobs by priority. As well as allocating incoming jobs starting from the left side of the timeslot.

5. Conclusion

From the study results, it can be concluded that the average idle time of FCFS-LRH is lower than FCFS by 24.39% and FCFS-EDF by 16.89%. The FCFS-LRH idle time value is lower because the FCFS-LRH scheduling policy is carried out by sorting incoming jobs by priority. As well as placing jobs starting from the far left of the timeslot. This is not done in the FCFS and FCFS-EDF methods.

Reference

[1] S. Kumari and G. Kumar, "Survey on Job Scheduling Algorithms in Grid Computing," *Int. J. Comput. Appl.*, vol. 115, no. 15, pp. 17–20, Apr. 2015, doi: 10.5120/20227-2511.

[2] A. Sulistio, K. H. Kim, and R. Buyya, "On incorporating an on-line strip packing algorithm into elastic grid reservation-based systems," *Proc. Int. Conf. Parallel Distrib. Syst. - ICPADS*, vol. 1, 2007, doi: 10.1109/ICPADS.2007.4447738.

[3] A. Sulistio, U. Cibej, S. K. Prasad, and R. Buyya, "GarQ: An efficient scheduling data structure for advance reservations of grid resources," *Int. J. Parallel, Emergent Distrib. Syst.*, vol. 24, no. 1, pp. 1–19, 2009, doi: 10.1080/17445760801988979.

[4] A. B.Patel, "Modeling and Simulation of Grid Resource Brokering Algorithms," *Int. J. Comput. Appl.*, vol. 42, no. 8, pp. 31–36, 2012, doi: 10.5120/5715-7774.

[5] H. B. Prajapati and V. A. Shah, "Scheduling in Grid Computing Environment," in *2014 Fourth International Conference on Advanced Computing & Communication Technologies*, Feb. 2014, pp. 315–324, doi: https://doi.org/10.1109/ACCT.2014.32.

[6] A. Sulis, "GRID computing approach for multireservoir operating rules with uncertainty," *Environ. Model. Softw.*, vol. 24, no. 7, pp. 859–864, Jul. 2009, doi: https://doi.org/10.1016/j.envsoft.2008.11.003.

[7] C. Castillo, G. N. Rouskas, and K. Harfoush, "On the design of online scheduling algorithms for advance reservations and QoS in grids," *Proc. - 21st Int. Parallel Distrib. Process. Symp. IPDPS 2007; Abstr. CD-ROM*, 2007, doi: 10.1109/IPDPS.2007.370226.

[8] I. Foster and C. Kesselman, "The history of the grid," *Adv. Parallel Comput.*, vol. 20, pp. 3–30, 2011, doi: 10.3233/978-1-60750-803-8-3.

[9] I. Foster, C. Kesselman, and S. Tuecke, "The Anatomy of the Grid," *Grid Comput.*, pp. 169–197, 2003, doi: 10.1002/0470867167.ch6.

[10] A. Chervenak, I. Foster, C. Kesselman, C. Salisbury, and S. Tuecke, "The data grid: Towards an architecture for the distributed management and analysis of large scientific datasets," *J. Netw. Comput. Appl.*, vol. 23, no. 3, pp. 187–200, 2000, doi: 10.1006/jnca.2000.0110.

[11] R. Buyya, D. Abramson, and J. Giddy, "Nimrod/G: An architecture for a resource management and scheduling system in a global computational grid," *Proc. - 4th Int. Conf. High Perform. Comput. Asia-Pacific Reg. HPC-Asia 2000*, vol. 1, pp. 283–289, 2000, doi: 10.1109/HPC.2000.846563.

[12] C. T. Yang, W. C. Shih, and C. H. Hsu, "On utilization of the grid computing technology for video conversion and 3D rendering," *Comput. Stand. Interfaces*, vol. 32, no. 1–2, pp. 29–37, 2010, doi: 10.1016/j.csi.2009.06.003.

[13] K. Al Tabash, A. Barradah, and R. Al Shaikh, "Empirical Utilization Analysis for High Performance and Grid Computing," in *2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation*, Mar. 2014, pp. 392–398, doi: https://doi.org/10.1109/UKSim.2014.47.

[14] S. Sahhaf, M. Barshan, W. Tavernier, H. Moens, D. Colle, and M. Pickavet, "Resilient algorithms for advance bandwidth reservation in media production networks," *Proc. 2016 12th Int. Conf. Des. Reliab. Commun. Networks, DRCN 2016*, no. Drcn, pp. 130–137, 2016, doi: 10.1109/DRCN.2016.7470847.

[15] A. A. Haruna, B. Z. Nordin, and H. Narleeni, "Grid Resource Allocation: A Review," *Res. J. Inf. Technol.*, vol. 4, no. 2, pp. 38–55, 2012, [Online]. Available: http://www.maxwellsci.com/print/rjit/v4-38-55.pdf.

[16] M. B. Qureshi, M. A. Alqahtani, and N. Min-Allah, "Grid resource allocation for real-time data-intensive tasks," *IEEE Access*, vol. 5, pp. 22724–22734, 2017, doi: 10.1109/ACCESS.2017.2760801.

[17] Kai Hwang and Zhiwei Xu, "Scalable parallel computers for real-time signal processing," *IEEE Signal Process. Mag.*, vol. 13, no. 4, pp. 50–66, Jul. 1996, doi: https://doi.org/10.1109/79.526898.

[18] M. P. Tiemeyer and J. S. K. Wong, "A task migration algorithm for heterogeneous distributed computing systems," *J. Syst. Softw.*, vol. 41, no. 3, pp. 175–188, Jun. 1998, doi: https://doi.

org/10.1016/S0164-1212(97)10018-8.

[19] R. Umar, A. Agarwal, and C. R. Rao, "Advance Planning and Reservation in a Grid System," *Commun. Comput. Inf. Sci.*, vol. 293 PART 1, pp. 161–173, 2012, doi: 10.1007/978-3-642-30507-8_15.

[20] A. Pujiyanta, L. E. Nugroho, and Widyawan, "Resource allocation model for grid computing environment," *Int. J. Adv. Intell. Informatics*, vol. 6, no. 2, pp. 185–196, 2020, doi: https://doi.org/10.26555/ijain.v6i2.496.

[21] A. Iosup, D. H. J. Epema, J. Maassen, and R. Van Nieuwpoort, "Synthetic grid workloads with Ibis, KOALA, and GRENCHMARK," in *Integrated Research in GRID Computing - CoreGRID Integration Workshop 2005, Selected Papers*, 2007, pp. 271–283, doi: 10.1007/978-0-387-47658-2_20.

[22] A. Sulistio, K. H. Kim, and R. Buyya, "Using revenue management to determine pricing of reservations," in *Third IEEE International Conference on e-Science and Grid Computing (e-Science 2007)*, 2007, pp. 396–404, doi: 10.1109/E-SCIENCE.2007.83.

[23] M. Carvalho and F. Brasileiro, "A user-based model of grid computing workloads," in *2012 ACM/IEEE 13th International Conference on Grid Computing*, 2012, pp. 40–48, doi: 10.1109/Grid.2012.13.

[24] A. Pujiyanta, L. E. Nugroho, and Widyawan, "Planning and Scheduling Jobs on Grid Computing," *Proceeding - 2018 Int. Symp. Adv. Intell. Informatics Revolutionize Intell. Informatics Spectr. Humanit. SAIN 2018*, pp. 162–166, 2019, doi: https://doi.org/10.1109/SAIN.2018.8673372.

[25] A. Pujiyanta, L. E. Nugroho, and Widyawan, "Advance Reservation for Parametric Job on Grid Computing," *Proc. 2019 4th Int. Conf. Informatics Comput. ICIC 2019*, pp. 0–4, 2019, doi: https://doi.org/10.1109/ICIC47613.2019.8985978.