International Journal of Engineering & Technology, *7 (4) (2018) 3097-3102*

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET
doi: 10.14419/ijet.v7i4.1756
Research paper

Quantum time task scheduling technique in novel hybrid shortest job first and round robin

Mohd Noor Derahman [1] *, Ahmad Shakir Roslan , Fahrul Hakim Ayob

[1] *Faculty of Computer Science and Information Technology, University Putra Malaysia*
**Corresponding author E-mail: mnoord@upm.edu.my*

Abstract

In a cloud computing environment, there are huge number of tasks with different computing requirements need to be scheduled and provisioned to the various resources within different capabilities. Thus, the mapping between users and resources is crucial so that the performance could be improved. The hybrid algorithm Shortest-Job-First (SJF) and Round Robin (RR) are expected to address all the concerns in scheduling task namely response time, waiting time and turnaround time simultaneously. Existing schedulers has been focused on those parameters but starvation problems are mostly not their major concern. Therefore, this study attempts to produce a better performance of hybrid algorithm through the integration of two traditional algorithms namely SJF and RR with dynamic quantum (SRDQ). Our proposed SRDQ with the best quantum time approach apparently reduces the longer waiting time when involves with a large cloudlet. Thus, it is suitable for the cloud computing environments where the resource hunger applications are normally provisioned.

Keywords: *Dynamic Variable Quantum Time; Round Robin; Scheduling; Shortest Job First.*

1. Introduction

Cloud computing offers services over the Internet such as computational task, storage, video-audio streaming and database manage-ment. Cloud service user (CSU) is considered as a client that submit their tasks such as cloudlet/s to the cloud service provider (CSP) which offers the above services. Prior to the resource assignment, proper procedures such as resource discovery, resource scheduling and provisioning need to be performed. Thus, cloud services need a great amount of resource control in order to provide a good perfor-mance based on the priority service level agreement (SLA) between CSUs and CSPs; therefore, a good scheduling is required to manage jobs and tasks.
Scheduling is a method where tasks are assigned to resources in order to be completed. In cloud computing, a broker or a local scheduler will take this responsibility to schedule the tasks. For this effort, the scheduler keeps resources as busy as possible to gain optimum throughput that leads user-satisfaction guaranteed. Currently, there are many schedulers with different objective function. Some of the schedulers focus on minimizing the waiting time. While the other are trying to maximize the throughput. These objective functions are always conflicting with each other. Hence, to find the optimum scheduler which satisfies everyone is quite challenging.
There are many schedulers created from basic algorithms, or a com-bination of multiple basic algorithms into a common scheduler to meet their objective.
The advancement of scheduling algorithm has been a popular topic in cloud computing. Many researchers are trying to create a scheduling algorithm to extend the advantages and minimize the disadvantages of the primitive scheduling algorithm. An efficient task scheduler s highly demanded in accordance with the hasty evolution of con-temporary cloud computing which leads to the optimum level of performance. There have been numerous conventional task schedul-ing algorithms commonly used such as First-Come- First-Serve (FCFS) [1, 2], Round Robin[3], Shortest-Job- First[4], Max-min[5], and also Min-Min[6]. Those are the classical example of basic job tasks scheduling in cloud computing.
A considerable amount of waiting time which leads to starvation problem in cloud computing are worrying the cloud providers as this could drop the service performance. As mentioned before, researchers are combining scheduling algorithm to improve the per-for-mance of the cloud computing. There are few hybrids algorithm such as FCFS and SRTF [7], which try to solve the starvation problem but unfortunately the starvation problem is still persists. Therefore, this study is going to evaluate a new hybrid algorithm known as SJF and RR with Dynamic Quantum (SRDQ) which was recently proposed and claimed to be able to minimize the turnaround and waiting time and partially minimize the long task starvation.
Meanwhile,an innovative hybrid task scheduling algorithm, which combines Shortest-Job-First (SJF) and Round Robin (RR) schedulers, which consider a dynamic variable task quantum is proposed [8]. The hybrid of these two algorithms are called SJF and RR with Dynamic Quantum (SRDQ). In this study, there are two basic keys that serve as the reliable core to the proposed algorithms; (1) a balanced waiting time between short and long task achieved from a dynamic task, and (2) the ready queue which is split into two sub-queues. The short and long tasks were labeled as Q1 and Q2, respectively. Communally, two tasks from Q1 and one from Q2 were assigned to the resources. However, the experiment conducted by the researchers did not consider increasing the number of task in the performance evaluation. Therefore, it failed to efficiently prove the reliability of the algorithm especially when involve a large cloudlet. The rest of this paper is presented as follows. Related work is discussed in Section 2. Description and explanation of the existing and an extended algorithm are elabo-

rated in Section 3. Results and discussion are conveyed in Section 4. Finally, Section 5 is conclusion and future works.

2. Related work

Cloud computing has been receiving enormous attention and it is highly popular in the Information Technology (IT) sector [9]. This is due to the budget limitation which faced by companies as well as individuals to purchase the hardware and software. Cloud computing has influenced individuals in many aspects, such as the way computing services are conducted. As the resources are being shared, it is important to ensure that the scheduler is using the best algorithm that tallied with the requirements. Therefore, previous and current researchers have been studying hybrid algorithm that attempts to improve different types of algorithms.

The study conducted by [10], seeks to optimize scheduling using ABC-SA method. The paper focused on an optimization of hybrid algorithm through integration of simulated annealing (SA) into artificial bee colony (ABC) algorithm. The aim is to achieve an effectual scheduling based on the size of the task, request priority, as well as the closest distance between client nodes and the cloud server. The findings exemplified that the proposed algorithm is more competent and it was also found to be more apposite for cloud computing environment as it reduces makespan when the number of resources is increases. However, this study did not properly validate the waiting time, which leads to starvation.

Previous work such as [11] proposes a new algorithm to reduce the waiting time. The proposed Orthogonal Taguchi Based-Cat Swarm Optimization (OTB-CSO) is focused on how to reduce the makespan. Other algorithms such as Particle Swarm Optimization with Linear Descending Inertia Weight (PSO- LDIW), Hybrid Particle Swarm Optimization with Simulated Annealing (HPSO-SA) and Minimum and Maximum Job First (Min-Max) are used to compare with the proposed algorithm. This study proved that the makespan can be reduced. However, the experiment conducted by the researcher only consider up to 100 number of tasks and did not consider to increase the number of task in their performance evaluation. Therefore, it could not efficiently prove the reliability of the algorithm.

[12] Refers cloud computing as internet service, which comprises of a variety of domains such as research, media, bigdata analysis as well as business. Therefore, this type of environment requires an efficient task scheduling, which currently being a fundamental issue in cloud computing. According to [12], there were many metaheuristic algorithms and hard optimization problems proposed in the effort to solve task scheduling in cloud computing. Hence, it is crucial that scheduling strategy adaptation needs to be conducted by an efficient

Scheduler. This paper attempts to propose a combination of firefly algorithm (FA) and particle swarm optimization (PSO) heuristics for cloud scheduling. The integration of these algorithms was intended to ensure the entire system is able to achieve a balance load, while consequently decreases the makespan of a set of tasks. The study was simulated using CloudSim tool kit package. Researchers found that the proposed FA is performing better as compared to the PSO, min-min scheduling and the first come first serve algorithms. How-ever, this study did not clarify the waiting time. Therefore, starvation might occur. Another researcher [13], proposed a task scheduling technique based on the ant colony algorithm. The focus of the study is to minimize the makespan and cost of the task hence improving the load balancing. They propose Multi-Objective Optimization Algorithm for Cloud Computing Task Scheduling Based on Improved Ant Colony

Algorithm (MO-ACO). The researcher managed to achieve their tar- get and also refined the pheromone initialization and update method within the ant colony algorithm. By using Cloudsim, they conduct the experiment. The outcome of the experiment was compared with Min-Min algorithm and the Ant Colony Optimization (ACO) algorithm. The results have proved that the proposed algorithm is able to minimize the cost and makespan and improve the system load balancing. However, the simulation is limited to 10 virtual machines and does not test on smaller or larger number of virtual machine, which leads to uncertain consistency in the test results.

[14] Proposed Choco-Based algorithm (CB), improved FFD (IFFD) and improved BFD (IBFD) to optimize the resource allocation. They believes, other than CPU and memory, the bandwidth requirement is also important but always being neglected and was not taken into consideration. By using Java and Choco, they appraise the performance of their algorithm with different number of virtual machines. From the analysis, it is noticed that the proposed algorithms was able to allocate resources better than the existing algorithms. However, they did not clarify the waiting time that result to the starvation. To summarize, there are many factors highlighted and improved in cloud computing scheduling. Most studies focus on minimizing the makespan and cost but unfortunately it does not really focus on minimizing the waiting time that will lead to the starvation problem. The race of finding the most optimum scheduling technique to be used in the cloud computing environment are indeed very intriguing and challenging hence many researchers are keen to help improving this area of study. Therefore, this study is going to focus on minimizing the waiting time. Hence, the starvation problem will be reduced and none of the tasks will be neglected.

3. System model

This section presents the SJF and RR with dynamic quantum based on SRDQ [8]. The simulation is implemented in the CloudSim environment which involves the host level and VM level scheduling. The above paper had proposed the SJF and RR with Dynamic Quantum (SRDQ) algorithm to overcome the starvation problem in the cloud computing environment. SRDQ intends to minimize the weakness of SJF which does have a high waiting time for longer job which does lead to the starvation problem and maximize the strengths of both the SJF and the RR by combining the two algorithms. The reason of using the RR for the hybrid is due to the fact that starvation is never occur in RR. Therefore, the idea of combining the two algorithms is going to be a better algorithm is suggested. SRDQ does show a better performance in this study. The implementation of the scheduling is showed in Algorithm 1 while the simulation parameters are referred in [8]. Calculation of the quantum time for the cloudlets is based on the Eq. 1.

$$\mathrm{q}_{ij} = q^{\sim} + \frac{q^{\sim}}{\left(B_{ij}+q_{i(j-1)}\right)^2} \tag{1}$$

where q_{ij} is the quantum at round j, i: 1,2,...,n and, B_{ij} is burst time of task i at round j,qi(j−1) and α is a binary selector α = 0,1(refers to [8]). For the first round, j = 1, q $_{i(j-1)}$ is zero because there is no previous round. It will be set to either zero or one based on the source queue. Meanwhile, for the cloudlet 2, the calculation of quantum time is slightly different as in Eq. 2.

$$\mathrm{q}_{ij} = q^{\sim} + \frac{q^{\sim}}{\left(B_{ij}-q_{i(j-1)}\right)^2} \tag{2}$$

The median of $q^{\sim}$ will be updated for a new cloudlet as in Eq. 3.

$$q^{\sim} = q^{\sim} + \frac{q^{\sim}}{Bnew} \tag{3}$$

while for the finished cloudlet is Eq. 4

$$q^{\sim} = q^{\sim} + \frac{q^{\sim}}{Bterminated} \tag{4}$$

Algorithm 1: SRDQ with quantum time

```
Data: cloudlet ID, burst time, median/mean/best
Result: quantum time initialization;

Sort cloudlets ascending order based on the burst time; cloudlet;
calculate the median, q;
Separate the cloudlets into Q1 and Q2
Calculate quantum qi j current execution task for both Q1 and Q2
based on Eq. 1 in [8]
While cloudlet in Q1 or Q2 do
    Read current;
    If cloudlet in Q1 then
        Calculate quantum time for Q1
    Else
        Calculate quantum time for Q2
    End
    Assign [2] tasks from Q1 to a resource;
    Assign [1] task from Q2 to a resource;
End
Update median, q for any new or finished cloudlet.
```

3.1. Modification of the existing SRDQ

Note that the SRDQ uses median as a based technique to evaluate the quantum time for each cloudlet. It will then divide the cloudlets into two groups. The use of median are great if the objective of the division is to divide the cloudlets with the queue length or size. In this study, we are going to investigate their effect when the division is between the small and large cloudlet. Thus, we propose the use of mean and best for group division and the quantum time calculation for each of cloudlet.

In this case, *mean* or also known as average is the sum of all numbers, divided by the quantity of the numbers. Since mean consider all the numbers (or cloudlets), we believe that it will give a better value to differentiate between the small and large cloudlets. The formula for calculating the mean value is showed in Eq. 5.

$$q^{mean} = \sum_{k=1}^{n} cloudletID / n \tag{5}$$

Meanwhile, *best* is considered as the average sum of the median and mean. In other words, the sum of median and mean is divided into two. A study conducted by [15] proposed best time quantum for round robin scheduling. It shows a better performance than the conventional RR. Therefore, the uses of best for SRDQ are also being tested in this study. The formula for calculating best is showed in Eq. 6.

$$q^{mean} = \sum_{k=1}^{n} \left(q^{median} + q^{mean}\right) / 2 \tag{6}$$

3.2. Dataset 1

Dataset in Table 1 is adopted from previous study conducted by [8]. This dataset was used to compare the results of the previous author with the result of the SRDQ. This is crucial to ensure the reimplementation is correct before further experiments are done. The experiment for this dataset (dataset 1) is then extended with the use of mean and best values as the quantum time.

Table 1: Sample Data for Scenario 1 with Six Cloudlet

cloudlet ID	arrival time	burst time
1	0	12
2	0	8
3	1	23
4	2	10
5	3	30
6	4	15

Table 2: Sample Data for Scenario 2 with Six Cloudlet

cloudlet ID	arrival time	burst time
1	0	12
2	5	210
3	5	44
4	7	157
5	8	101
6	8	13
7	19	179
8	11	92
9	13	144
10	15	158

The mean of dataset one by using Eq. 5 would be 16.3. Therefore, for the SRDQ mean, Q1 will have cloudlet 1, 2, 4 and 6 whereas Q2 will have cloudlet 3 and 5. As for the SRDQ best, the value would be 14.9 by using Eq.6. Q1 will have cloudlet 1,2 and 4 whereas Q2 will have cloudlet 3, 4 and 5. Note that there is a difference number of cloudlet in Q1 and Q2 for the mean and best. This due to the different formula to divide the cloudlets.

3.3. Dataset 2

Dataset in Table 2 is also from [8]. Again, the results of the previous author are compared and the experiment are extended with other algorithms.

The mean of dataset 2 by using Eq. 5 would be 111. Therefore, for the SRDQ mean, Q1 will have cloudlet 1,3,5,6 and 8 whereas Q2 will have cloudlet 2,4,7,9 and 10. As for the SRDQ, the best value would be 116.75 by using Eq. 6. Therefore, for the best SRDQ, Q1 will have cloudlet 1,3,5,6 and 8 whereas Q2 will have cloudlet 2,4,7,9 and 10.

3.4. Simulation assumptions

In order to simulate the scenario, CloudSim is used to create a data center with a broker and a single user. At the host level, FCFS algorithm is used to allocate the Virtual Machines (VMs) and space- shared policy were used at the VM level in order to execute cloudlets in sequence. A dedicated core will be assigned to each task when this policy is used. So, the execution time of each task will not be affected by the queue size or any incoming tasks. There are few assumptions for the simulation, which are:

- All cloudlets which have to be processed are available
- At run time, no more cloudlets are added
- The environment is also static i.e. no more resources are added at runtime

Finally, the CloudSim code was modified and the scheduling class was extended to implements the algorithms.

4. Results and discussion

The first simulation was done using dataset 1 with six cloudlets. There are eight rounds for the first simulation and each round used a different algorithm. The first round of the simulation used the SJF as the scheduling algorithm. After the results of the first round are noted the scheduling algorithm are changed to RR with time quantum five. The steps are repeated until all the scheduling algorithms are tested and the results are identified. The results of the first simulation are illustrated in this section. The results of the first simulations are then compared and analyse with the results from the previous work. Some manual calculations are also being done to compare the result from this simulation to ensure the validity of results.

Figure 1 illustrates the response time by six cloudlets. It is noticed that the cloudletID 1 has 0 response time value for all the RR related algorithms. This is because RR implementations are based on the FCFS rules. This means that cloudletID 1 will be the first cloudlet among the six cloudlets that get an attention from the scheduler and getting the first chance to be executed. Meanwhile, for the SRDQ algorithm cloudletID [2] will be the first to be executed that shows by the lowest response time. This is due to the SRDQ will sort the cloudlets in increasing order and execute the smallest cloudlet first. Thus, cloudletID 2 has the lowest value of response time among the other cloudlets.

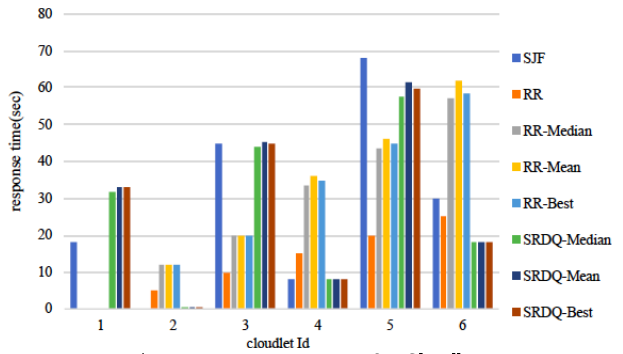

Fig. 1: Response Time over Six Cloudlets.

In short, RR scheduling with time quantum five has the lowest average response time. This is because, with the time quantum, it will hold and switch the cloudlet that are being executed with the next cloudlet every five seconds. Unfortunately, changing state in every five seconds will increase the number of context switching needed to finish all cloudlets, and this will definitely affect the performance. For the RR with median, mean and best; RR median shows the best response time average. This is due to the lowest value of the median among the median, mean and best. For the SRDQ, SRDQ median got the best average response time. SRDQ median got the best average response time also because median has the lowest value among the median, mean and best. SRDQ median does have better performance than RR median for the average response time. This is due to the arrangement of the cloudlet that are being sorted in a SJF mode in the SRDQ, which does help to elevate the performance of the SRDQ.

Figure 2 shows the waiting time by cloudlet. It is noted that the first cloudlet has zero value for the RR median, mean and best because a regular RR implement the FCFS and the cloudlets are finish in the first round. The first cloudlet that are using RR median, mean and best can finish in the first round because the value of median, mean and best are higher than the value of the burst time of the first cloudlet. Even though, the first cloudlet using RR with time quantum five was the first cloudlet to be executed, the fact that the burst time is twelve and it is higher than five make it impossible to finish executing in one round. Hence it needs to wait for at least two times before

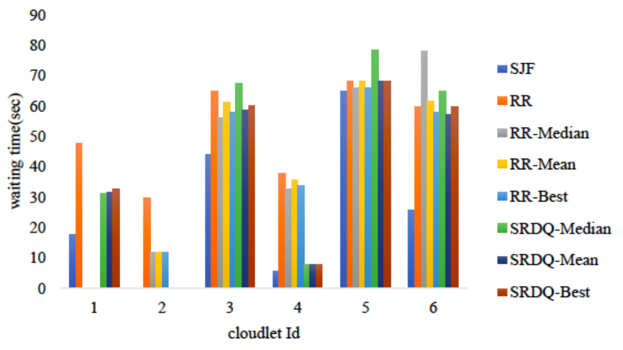

Fig. 2: Waiting Time over Six Cloudlets.

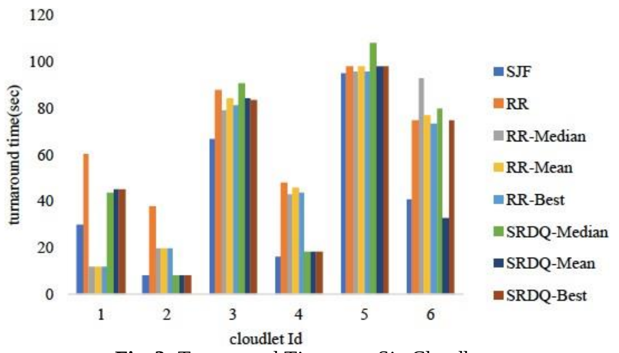

Fig. 3: Turnaround Time over Six Cloudlets.

It can finish. That is the reason why the first cloudlet using the RR with time quantum five has a high waiting time compare to the other RR. For all the SRDQ cloudlet, two will be the first to be executed and also finish in the first round. The second cloudlet that are using SRDQ median, mean and best can finish in the first round because the value of median, mean and best are higher than the value of the burst time of the second cloudlet. For overall, SJF has the lowest average waiting time but there is risk of high waiting time for large tasks. The problem of high waiting time for large tasks in SJF is simply because all the small cloudlets will be executed first, so the largest task will be the last one to receive the resources. For the RR with median, mean and best, RR best got the best average waiting time. For the SRDQ, SRDQ mean got the best average waiting time. SRDQ mean is better than RR best. Again, SRDQ mean is better because the cloudlets are sorted which does help to elevate the performance of the SRDQ.

Figure 3 show the turnaround time by cloudlet. Noted that the cloudletID 1 has low value for the RR median mean best because a regular RR implement the FCFS and the cloudlet are finish in the first round. It is a different case for RR with time quantum five which have lowest turnaround time at the second cloudlet because cloudlet two was the first cloudlet to finish when using the RR with time quantum five. For the SRDQ cloudlet two will be the first to finish in the first round because it is the first cloudlet to be executed and it has a lower burst time compare to the value of median, mean and best. For overall, SJF has the lowest average turnaround time but as mention before in SJF there is risk of high waiting time for large task. For the RR with median, mean and best RR best got the best average which proves the study done by [15]. For the SRDQ, SRDQ mean got the best average turnaround time. SRDQ mean is better than RR best because SRDQ mean got a lower average value of turnaround time compare to the RR best.

4.1. A larger cloudlet

The second simulation was done using dataset 2 with 10 cloudlets. There are also eight rounds for the second simulation and each round uses a different algorithm. The second simulation uses the same steps as the first one but there was no manual calculation made for the second simulation as the validity of the algorithm is verified during the first simulation. The results were compared with the previous works to ensure the same pattern are obtained. The response time, waiting time and turnaround time for each cloudlet is observed and the average performance of each algorithm is computed. Note that for the second dataset, cloudlet one has the lowest burst time. The results for the second simulation are given and discussed in the next sub section.

Graph in Figure 4 shows the response time by 10 cloudlets. The first cloudlet is the lowest cloudlet in the dataset and will be the first to be executed. Overall, RR with time quantum five has the lowest average response time with high context switching. This is due to the small number of quantum assigned. For RR with median, mean and best, RR mean is the best compared to SRDQ. For the lower average value of response time, SRDQ is better than that of RR. Figure in 5 show the waiting time by cloudlet. Note that in the first cloudlet RR with time quantum five has a waiting time because it does not finish executing in the first round. For SRDQ, cloudlet one will be the first to be executed and also finish in the first round. Overall, SJF has the lowest average waiting time but there is a risk of high waiting time for a large task as mentioned before where it is has been a common problem in SJF. For RR with median, mean and best, again RR best is the best average which justify the study done by [15]. In terms of response time, SRDQ is better than that of RR since the waiting time is lower.

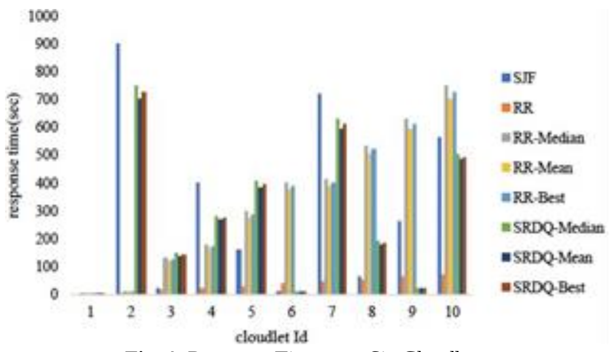

Fig. 4: Response Time over Six Cloudlets.

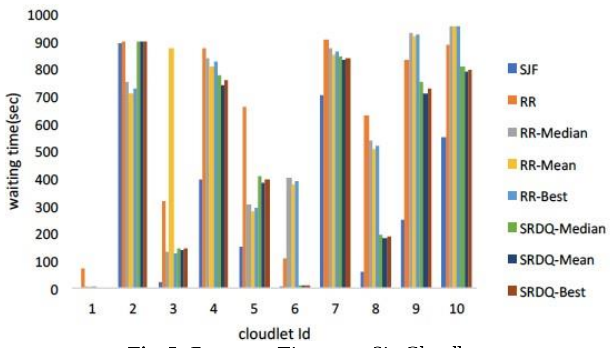

Fig. 5: Response Time over Six Cloudlets

Figure 6 shows the turnaround time for each cloudlet. Note that cloudlet with ID1 has low value for all the algorithms except for RR. This is because the first cloudlet finishes in the first round except for

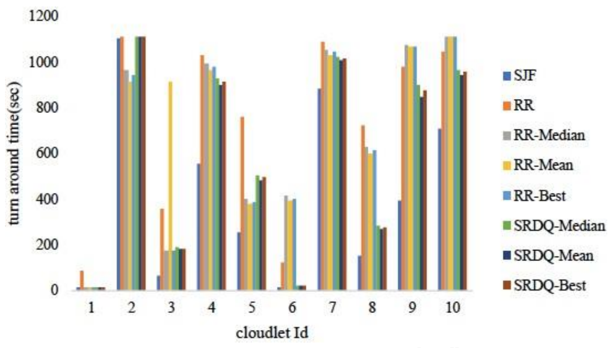

Fig. 6: Response Time over Six Cloudlets.

RR with time quantum five. In this case, SJF has the lowest average turnaround time. Meanwhile, RR with median, mean and best again RR best is the best average. For SRDQ, again SRDQ mean is the best average turnaround time which clarifies SRDQ is better than RR in that way.

Meanwhile, dataset 3 was used for the third simulation. It consist of 50 random number between 1000-50000 round up to the nearest thousand. This dataset was generated by the CloudSim environment. Figure 7 shows the evaluation of 50 random cloudlets on 1, 2 and VMs. For this experiment, only SRDQ-Median, SRDQ-Mean, and SRDQ-Best are tested. From the previous simulations, we notice that these three algorithms are better than the rest. This simulation is subjected in determining the best algorithms to schedule a total number of 50 cloudlets. Figure 7 shows that the value of time is decreasing as the number of VM used is increasing. This is because the resources are being increased. Note that, SRDQ mean is considered outperform for average of response time, waiting time and turnaround time for all the three cases. Therefore, for the dataset of 50 cloudlets, SRDQ mean is the best scheduler compare to the other SRDQ.

Figure 8 show the evaluation of 100 random cloudlet on 1,2,3 VMs. The results of this simulation are also shown in average due to high number of cloudlets used. The pattern of the graph is the same as the previous simulation (Figure 7) which is expected because this experiment just increases the number of cloudlets. As for the results for response time SRDQ-Median got the lowest average. Meanwhile, waiting time and turnaround time, SRDQ-Best has the lowest average value. Thus, we can conclude that for this simulation, SRDQ-Best is the best algorithm for the scenario.

After the simulations and the results are identified, we notice that SRDQ has better performance than that of the primitive algorithm. It does combine the advantages of both SJF and RR algorithm. For the simulations of 50 cloudlets and less, we could see that SRDQ mean performs better than the others, but when we increase the number.

RR with time quantum five. In this case, SJF has the lowest average turnaround time. Meanwhile, RR with median, mean and best again RR best is the best average. For SRDQ, again SRDQ mean is the best average turnaround time which clarifies SRDQ is better than RR in that way.

Meanwhile, dataset 3 was used for the third simulation. It consist of 50 random number between 1000-50000 round up to the nearest thousand. This dataset was generated by the CloudSim environment.

Figure 7 shows the evaluation of 50 random cloudlets on 1, 2 and VMs. For this experiment, only SRDQ-Median, SRDQ-Mean, and SRDQ-Best are tested. From the previous simulations, we notice that these three algorithms are better than the rest. This simulation is subjected in determining the best algorithms to schedule a total number of 50 cloudlets. Figure 7 shows that the value of time is decreasing as the number of VM used is increasing. This is because the resources are being increased. Note that, SRDQ mean is considered outperform for average of response time, waiting time and turnaround time for all the three cases. Therefore, for the dataset of 50 cloudlets, SRDQ mean is the best scheduler compare to the other SRDQ.

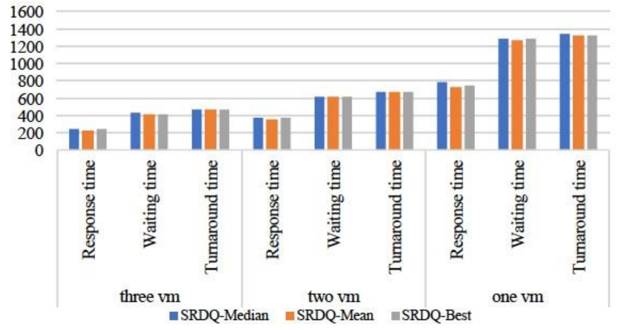

Fig. 7: 50 Cloudlets.

Figure 8 show the evaluation of 100 random cloudlet on 1,2,3 VMs. The results of this simulation are also shown in average due to high number of cloudlets used. The pattern of the graph is the same as the previous simulation (Figure 7) which is expected because this experiment just increases the number of cloudlets. As for the results for response time SRDQ-Median got the lowest average. Meanwhile, waiting time and turnaround time, SRDQ-Best has the lowest average value. Thus, we can conclude that for this simulation, SRDQ-Best is the best algorithm for the scenario.

After the simulations and the results are identified, we notice that SRDQ has better performance than that of the primitive algorithm. It does combine the advantages of both SJF and RR algorithm. For the simulations of 50 cloudlets and less, we could see that SRDQ mean performs better than the others, but when we increase the number of cloudlets to 100 the performance of the SRDQ mean dropped and SRDQ best perform better than SRDQ mean. Therefore, SRDQ best is more suitable to be implemented in the cloud computing environment.

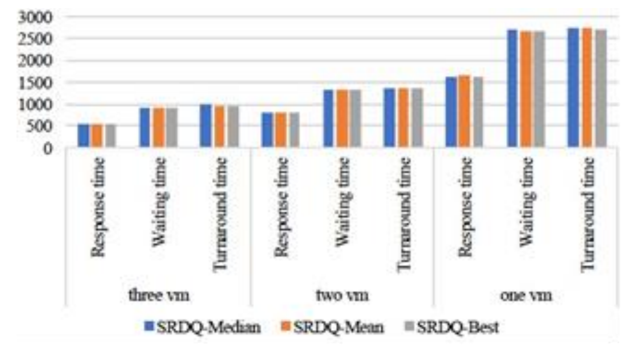

Fig. 8: 100 Cloudlets.

5. Conclusion

This study has shown that the arrangement of task and the dynamicity for the value of the quantum time does affect the performance of the scheduling algorithms. The SRDQ algorithm which is proposed by the previous researcher does perform better than that of the conventional scheduling algorithms which are commonly used in the cloud computing environment these days. The use of Best for the quantum time does improve the division between the small and large cloudlets. Hence the performance of the SRDQ is considered to perform better. Overall, we conclude that the SRDQ-Best performs better than the original SRDQ in the cloud computing environment. For the future work, the researcher could continue the experiments and compare the SRDQ-Best with other scheduling algorithms and find another way that could improve the performance of the SRDQ-Best in minimizing the waiting time and the starvation problems.

Acknowledgment

The authors would like to thank the Malaysia Ministry of Higher Education (MOHE) for for funding this study through Fundamental Research Grant Scheme (FRGS) Grant No. 5524662 and MyRA.

References

[1] R. Buyya, R. Ranjan, and R. N. Calheiros, “Modeling and simulation of scalable cloud computing environments and the cloudsim toolkit: Challenges and opportunities,” in *Proceedings of the 2009 Interna- tional Conference on High Performance Computing and Simulation, HPCS 2009*, 2009.

[2] J. Thaman and M. Singh, “Green cloud environment by using robust planning algorithm,” *Egyptian Informatics Journal*, vol. 18, no. 3, pp. 205–214, 2017.

[3] R. Garg and A. K. Singh, “Multi-objective workflow grid scheduling Using ε -fuzzy dominance sort based discrete particle swarm optimiza- tion,” *Journal of Supercomputing*, vol. 68, no. 2, pp. 709–732, 2014. https://doi.org/10.1007/s11227-013-1059-8.

[4] A. A. Chandio, K. Bilal, N. Tziritas, Z. Yu, Q. Jiang, S. U. Khan, and C. Z. Xu, “A comparative study on resource allocation and energy efficient job scheduling strategies in large-scale parallel computing systems,” *Cluster Computing*, vol. 17, no. 4, pp. 1349–1367, 2014. https://doi.org/10.1007/s10586-014-0384-x.

[5] O. M. Elzeki, M. Z. Reshad, and M. A. Elsoud, “Improved Max-Min Algorithm in Cloud Computing,” *International Journal of Computer Applications*, 2012.

[6] G. Patel, R. Mehta, and U. Bhoi, “Enhanced Load Balanced Min-min Algorithm for Static Meta Task Scheduling in Cloud Computing,” in *Procedia Computer Science*, 2015. https://doi.org/10.1016/j.procs.2015.07.385.

[7] S. Santra, H. Dey, S. Majumdar, and G. S. Jha, “New simulation toolkit for comparison of scheduling algorithm on cloud computing,” in *2014 International Conference on Control, Instrumentation, Communication and Computational Technologies, ICCICCT 2014*, 2014.

[8] S. Elmougy, S. Sarhan, and M. Joundy, “A novel hybrid of Shortest job first and round Robin with dynamic variable quantum time task scheduling technique,” *Journal of Cloud Computing*, vol. 6, no. 1, 2017.

[9] J. Zhang, H. Huang, and X. Wang, “Resource provision algorithms in cloud computing: A survey,” *Journal of Network and Computer Applications*, vol. 64, pp. 23–42, 2016. [Online]. Available: https://doi.org/10.1016/j.jnca.2015.12.018.

[10] B. Muthulakshmi and K. Somasundaram, “A hybrid ABC-SA based optimized scheduling and resource allocation for cloud environment,” *Cluster Computing*, vol. 7, no. 4, pp. 1–9, 2017. https://doi.org/10.1007/s10586-017-1174-z.

[11] D. Gabi, A. S. Ismail, A. Zainal, and Z. Zakaria, “Solving task schedul- ing problem in cloud computing environment using orthogonal taguchi- cat algorithm,” *International Journal of Electrical and Computer Engi- neering*, vol. 7, no. 3, pp. 1489–1497, 2017.

[12] M. Aruna, D. Bhanu, and S. Karthik, “An improved load balanced metaheuristic scheduling in cloud,” *Cluster Computing*, pp. 1–9, 2017. https://doi.org/10.1007/s10586-017-1213-9.

[13] Q. Guo, “Task scheduling based on ant colony optimization in cloud environment,” vol. 040039, 2017, p. 040039. [Online]. Available: http://aip.scitation.org/doi/abs/10.1063/1.4981635

[14] W. Lin, B. Peng, C. Liang, and B. Liu, “Novel resource allocation model and algorithms for cloud computing,” in *Proceedings - 4th International Conference on Emerging Intelligent Data and Web Tech- nologies, EIDWT 2013*, 2013, pp. 77–82.

[15] D. Khokhar and A. Kaushik, “Best Time Quantum Round Robin Cpu,” *International Journal of Scientific Engineering and Applied Science (IJSEAS)*, vol. 3, no. 5, pp. 3–7, 2017.