Priority Weighted Round Robin Algorithm for Load Balancing in the Cloud

Ajay Katangur
Department of Computer Science
Missouri State University
Springfield, USA
ajaykatangur@missouristate.edu

Somasheker Akkaladevi
Department of Computer Information Systems
Virginia State University
Virginia, USA
sakkaladevi@vsu.edu

Sadiskumar Vivekanandhan
Department of Computer Science
Texas A&M University-Corpus Christi
Corpus Christi, USA
svivekanandhan@islander.tamucc.edu

***Abstract*—Cloud computing, which helps in sharing resources through networks, has become one of the most widely used technologies in recent years. Vast numbers of organizations are moving to the cloud since it is more cost-effective and easy to maintain. An increase in the number of consumers using the cloud, however, results in increased traffic, which leads to the problem of balancing tasks on the loads. Numerous dynamic algorithms [1] have been proposed and implemented to handle these loads in different ways. The performance of these dynamic algorithms are scaled with different parameters, such as response time, throughput, utilization, efficiency, etc. The weighted round-robin algorithm is one of the most widely used load balancing algorithms. The proposed algorithm is an improvement of the weighted round-robin algorithm, which considers the priority of every task before assigning the tasks to different virtual machines (VMs). The proposed algorithm uses the priority of tasks to decide to which VMs the tasks should be assigned dynamically. The same process is used to migrate the tasks from overloaded VMs to under-loaded VMs. The simulations are conducted using CloudSim by varying cloud resources. Simulation results show that the proposed algorithm performs equivalent to the dynamic weighted round robin algorithm in all the QoS factors, but it shows significant improvement in handling high-priority tasks.**

***Index Terms*—CloudSim, Cloud computing, Completion Time, Execution Time, Load balancing, Round-Robin Algorithm, Turnaround Time, Waiting Time**

I. Introduction

Cloud computing is a technology that helps to provide services through the internet. SaaS (Software as a Service), PaaS (Platform as a Service), and IaaS (Internet as a Service) are the three types of services provided by the cloud [2]. These services provide resources, such as memory, infrastructure, software, virtual machines, and so on, through the internet. Due to various benefits of cloud computing, increasingly more organizations are moving towards the cloud [1]. As a result of this growth, the usage and traffic across the cloud becomes high, which increases the need for more research in load balancing.

Load balancing is a method of sharing the entire load of a network to all nodes in the cloud. The goal of load balancing is to achieve maximum efficiency of the network by increasing both the throughput and the performance of each node [3]. To achieve this step, many algorithms have been formulated and implemented. The two most widely used algorithms to schedule tasks are round-robin (RR) and weighted round-robin (WRR) algorithms [4]. The round-robin algorithm distributes tasks in the cyclic order of virtual machines. A RR algorithm does not consider the processing capacity, length, or priority of the tasks. The weighted round-robin algorithm is an improved form of the RR. The WRR considers the processing capacity of virtual machines (VMs), and it assigns more number of tasks to the VM with more processing capacity based on the weight given to each VM. The proposed algorithm, which is the priority weighted round-robin algorithm (PWRR) is also an improved version of the WRR algorithm.

In this work, the priority of the tasks is considered as an important factor to efficiently select the appropriate VM for scheduling and migrating the tasks, ranging from low to high priority. The proposed algorithm also considers length of the task, processing capacity of each node, and number of tasks assigned to the nodes. In general, when the priority of tasks is considered, it is used to arrange incoming tasks from high to low priority. This ensures that the high-priority tasks are executed before other tasks that come under different priorities.

This material is based upon work supported by the National Science Foundation under Grant OAC-1828380

II. Related Work

Due to drastic growth in cloud computing and growth in the number of users, researchers have increasingly turned their focus towards load balancing and scheduling [5] [6] [7]. The overall performance of an entire cloud network completely relies on the load balancing techniques [8] [9] [3]. Load Balancer as a Service (LBaaS) [1] is one of the more popular methods; rather than building a load balancer for each cloud, companies prefer to consume LBaaS. In this method, the load balancer will be utilized through the network, and it requires information about the entire network, including information about every node within it. The majority of LBaaS providers use round-robin or weighted round-robin algorithm as their static algorithm [1]. These two algorithms work better than first come first served algorithm(FCFS), shortest job first algorithm(SJF), MaxMin algorithm, MinMin algorithm, etc. [4].

Round-Robin Algorithm: The RR algorithm uses the concept of distributing the request to the nodes in circular fashion

[10]. The weighted round-robin algorithm is an improved version of RR, in which each VM is assigned a weight [11], which is calculated based on the processing capacity of each node. Based on the weight, VMs are ordered from high to low. VMs with more weight are assigned more requests compared to VMs with less weight. This approach produces better performance and throughput.

Weighted Round-Robin Algorithm: Many researchers are also engaged in successive improvisation of the WRR algorithm by using different parameters, such as node processing power, queue length, job size, and so on, which makes the WRR algorithm a dynamic load balancing technique [12] [13] [14] [15]. This helps the algorithm to decide where the request should land by frequently calculating the load on each VM. In this process, the weight of each VM is calculated with the help of the load, which makes this algorithm dynamic. Along with all these parameters, inclusion of priority reduces the waiting time for the requests with higher priorities [16] [15].

Dynamic Weighted Round-Robin Algorithm: Recently, an improvised WRR algorithm was proposed that lifts WRR algorithms to the next level by handling the unevenness of VM loads in the network [13] [17]. The dynamic process of the WRR algorithm considers the load of VMs, along with the processing capacity and length of the task, to decide which VM should be allocated with the next job.

III. System Model

Figure 1 depicts the overall idea of the process of scheduling and load balancing in a cloud. The scheduler and load balancer are the most important components of any cloud. The task manager helps to collect information about each task arriving through the task queue. The scheduler and load balancer are the components that use a load balancing algorithm to distribute the loads evenly across the resources. The scheduler is a component that helps in deciding the appropriate VM to which an arrived task should be allocated. It ensures that the task is assigned to a VM that takes less time to complete the task by considering the amount of load in that specific VM and the total time needed to complete the entire load [13].

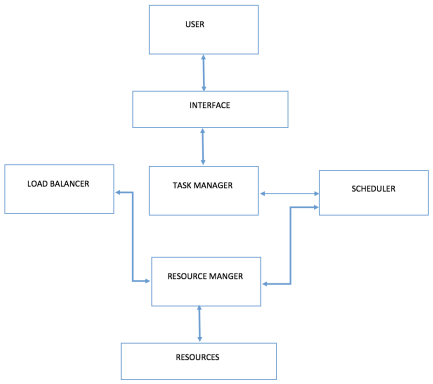

Fig. 1. Scheduler and Load Balancer of a Cloud Network

The incoming task is provided into the system through different types of interfaces, such as web applications, servers, and so on. The task first reaches the task manager, which identifies the length and priority of the task. The priority of tasks are given as integers (1-10). A task which has low priority value is considered as high priority task. The incoming tasks are then passed to the scheduler, which helps in allocating the incoming tasks to an appropriate VM by using the proposed algorithm. The scheduler makes use of the information about the resources from the resource manager to find the appropriate VM to handle the incoming task.

The load balancer helps in finding load unevenness in the system by calculating the ratio between the total number of jobs in the queue and the number of available VMs. In case of job migration, the load balancer considers the priority of the task to be migrated to identify which VM is under-loaded and has the least number of tasks of equal or higher priorities.

A. Priority Weighted Round Robin Algorithm

The two widely used load balancing algorithms are RR and weighted RR algorithms. The RR algorithm allocates the incoming requests to the available resources in a circular fashion, irrespective of the length of the task or the processing capacity of resources. The WRR algorithm considers the processing capacity of resources, and a higher number of incoming tasks are assigned to the resource with more processing power. These two algorithms are used to compare the performance of the proposed priority weighted round-robin algorithm. Along with these algorithms, the dynamic WRR algorithm is also used for comparison. In this algorithm, whenever a new request is received, it finds an appropriate VM by calculating the waiting time.

The proposed priority weighted round-robin algorithm uses the work of the WRR algorithm as a base, but it shifts the algorithm from static to dynamic. This algorithm helps to find an appropriate VM with the help of information that includes the processing power of VM, length of the tasks, priority of incoming tasks, and processing time required for VMs to complete the tasks of equal or higher priorities. Once a task arrives in the system, information about the task, such as the length and the priority of the task is noted. Then the weight of each VM is calculated, as mentioned earlier. Next, the incoming task is forwarded to the VM with more weight. Basically, the weight of a VM refers to the amount of time that specific VM requires to complete the tasks with equal or higher priorities in its queue. The weight of VM increases if the completion time is shorter, and vice versa. In general, the time taken to process a task can vary in runtime; in that situation, task migration is performed. The high priority task from the queue of the overloaded VM is migrated to the under-loaded or ideally loaded VM, by finding out which VM has fewer of equal or higher priority tasks. This helps the high-priority tasks to be completed faster, and the migration helps in increasing the proper utilization of resources.

IV. System Implementation

To implement the proposed priority weighted round-robin algorithm and to compare its effectiveness with the existing algorithms, the CloudSim 3.0.3 (hereinafter referred to as "CloudSim") simulation tool has been utilized. CloudSim is a framework used to simulate the cloud environment and its services [18]. This section provides information about the hardware and software used for the implementation. It also provides a detailed explanation of the architecture of CloudSim and important components involved, followed by the implementation of the round-robin algorithm, weighted RR algorithm, dynamic weighted RR algorithm, and priority weighted RR algorithm.

A. Implementation of RR Algorithm, Weighted RR Algorithm, and Dynamic Weighted RR Algorithm

The RR algorithm is implemented as in [10]. The weighted RR algorithm is implemented as in [8]. The dynamic weighted RR algorithm is implemented as in [13].

B. Implementation of Priority Weighted RR Algorithm

Algorithm 1 lists the step-by-step approach for the priority weighted RR algorithm. This algorithm is an improvement of the dynamic weighted RR algorithm. In the dynamic weighted RR algorithm, the waiting time of each VM is calculated for each request, and an incoming cloudlet is assigned to VM with less waiting time. But in this algorithm, the priority of each task is considered while calculating the waiting time of VMs.

Algorithm 1 Priority Weighted RR Algorithm Pseudocode

(1) Identify the priority of an incoming task.
(2) Calculate waiting time for each VM based on the priority on an incoming tasks.
(3) A weight is given to each VM based on the waiting time, VM with less waiting time should have more weight.
(4) Assign an incoming task to VM which has less waiting time based on priority.
(5) Once the incoming task is assigned, calculate threshold and load in each VM.
(6) If any VM is found to be overloaded
  (a) Pick high priority task from the task waiting queue of the VM.
  (b) Identity the under loaded VMs.
  (c) Select the VM which has less number of tasks of same or higher priority.
  (d) Continue till it has overloaded VMs.
(7) Process the high priority task from the waiting queue.
(8) If the low priority task is waiting for more than fixed number of iterations, increase the priority of the waiting task by one level
(9) Allow the VM to pick the next task from the waiting queue, it should select the high priority task from the queue and scheduler will continue from step 1 simultaneously

TABLE I
Data Center Specifications

Architecture	OS	VMM	Processing Cost	Memory Cost	Storage Cost	BW Cost
x86	Linux	Xen	3.0	0.05	0.01	0.1
x86	Linux	Xen	3.0	0.05	0.01	0.1

TABLE II
Configurations of Hosts

RAM (in MB)	Bandwidth (in Mbps)	Storage (in MB)	No. Of Cores
163,840	100,000	10,000,000	quad-core
163,840	100,000	10,000,000	dual-core

V. Results and Evaluation

This chapter features a detailed discussion about the cloud model and its setup for performing simulations and explains the results of each simulation. The quality of service (QoS) parameters used to compare the performance of the proposed algorithm with the existing algorithms are execution cost, overall execution time of each algorithm in milliseconds(ms), average execution time for one cloudlet of each algorithm in ms, time taken to complete level 1 priority tasks in ms, time taken to complete level 2 priority tasks and, time taken to complete level 5 priority tasks in ms.

A. Experimental Setup

This experiment focuses on scheduling the incoming tasks based on the priority of the tasks, along with the task length and the processing capacity of each resource. In CloudSim, the incoming tasks are also known as cloudlets. Each cloudlet has its own size, length, id, priority, and so on. In this experiment, different numbers of cloudlets with different sizes and different priorities, ranging from 1 to 10, are used. Each cloudlet is submitted to the datacenter broker, which is responsible for submitting the cloudlets to the data centers. Each data center has its own characteristics, such as architecture, OS, virtual machine monitor (VMM), processing cost, memory cost, storage cost, and bandwidth cost, as shown in Table I. Load to the cloud is also given dynamically by providing varying number of cloudlets of different sizes and priorities, as shown in Table IV.

As mentioned earlier, the QoS parameters considered in this experiment to compare the proposed algorithm with the

TABLE III
Configurations of VMs

MIPS	No. of CPUs	RAM (in MB)	Bandwidth (in Mbps)	Image Size (in MB)	VMM
250	1	512	1000	10000	Xen
500	1	512	1000	10000	Xen
500	1	512	1000	10000	Xen
750	1	512	1000	10000	Xen

TABLE IV
Configurations of Cloudlets

Length (in MI)	FileSize (in bytes)	OutputSize (in bytes)	Priority (range)
400	300	300	1-10
800	300	300	1-10
2000	300	300	1-10
4000	300	300	1-10

existing algorithms are execution cost, total completion time, total waiting time, total turnaround time, average execution time of one cloudlet, and time taken to complete level 1, level 2 and level 5 priority tasks.

B. Execution Cost

The execution cost refers to the total cost of executing a cloudlet. In this experiment, the total execution cost is calculated for the proposed algorithm in comparison with the existing algorithms. Execution cost is calculated using "(1)", where k represents the number of cloudlets in the cloud network, S_i represents the cloudlet size (in MI), V_m represents the processing capacity of a VM (in MIPS), and C_s represents cost per second required to process a cloudlet.

$$C_E = \sum_{i=1}^{k} \frac{S_i}{V_m} * C_s \tag{1}$$

C. Total Completion Time

Total completion time refers to the total time taken for the cloud network to complete the executions of all the cloudlets submitted to the network. The total completion time in ms is calculated using "(2)", where k represents the number of cloudlets in the cloud network, S_i represents the cloudlet size (in MI), V_m represents the processing capacity of a VM (in MIPS), and N_d represents the total network delay (in ms).

$$T = \sum_{i=1}^{k} \frac{S_i}{V_m} + N_d \tag{2}$$

D. Total Waiting Time

The total waiting time refers to the total time taken by all the cloudlets during processing in the cloud network. Total waiting time in ms is calculated using "(3)", where k represents the number of cloudlets in the cloud network, T_{C_i} represents completion time of a cloudlet (in ms), T_{A_i} represents arrival time of a cloudlet to the network (in ms), and T_{B_i} represents the burst time of the cloudlet (in ms).

$$T_W = \sum_{i=1}^{k} (T_{C_i} - (T_{A_i} + T_{B_i})) \tag{3}$$

E. Total Turnaround Time

Turnaround time refers to the total time taken by all cloudlets to complete processing in the cloud network. Total turnaround time in ms is calculated using "(4)", where k represents the number of cloudlets in the cloud network, T_{C_i} represents completion time of a cloudlet (in ms), and T_{A_i} represents arrival time of a cloudlet to the network (in ms).

$$T_{TA} = \sum_{i=1}^{k} (T_{C_i} - T_{A_i}) \tag{4}$$

TABLE V
RR PERFORMANCE WITH DIFFERENT VMS

QoS	20 VMs	25 VMs	35 VMs	50 VMs	60 VMs	70 VMs	80 VMs
Execution Cost	23874.99	32767.5	32274.4	23874.99	23874.99	23874.99	23874.99
Tot. Comp. Time	7835.33	10811.71	10644.55	7838.51	7838.51	7838.51	7838.51
Avg. Exec. Time	3.74	4.32	4.25	3.73	3.73	3.73	3.73

TABLE VI
DWRR PERFORMANCE WITH DIFFERENT VMS

QoS	20 VMs	25 VMs	35 VMs	50 VMs	60 VMs	70 VMs	80 VMs
Execution Cost	27458.42	27997.2	27833.3	22205.42	22817.8	22932.21	23003.43
Tot. Comp. Time	9068.95	9258.97	9226.8	7364.9	7375.3	7401.92	7432.7
Avg. Exec. Time	3.68	3.7	3.69	3.63	3.64	3.64	3.65

F. Average Execution Time of a Cloudlet

The average execution time of a cloudlet refers to the average time taken to complete the execution of a cloudlet. Average execution time of a cloudlet is calculated using "(5)", where T represents the total time taken to complete all cloudlets, as represented in equation (3.2) (in ms), and N represents the total number of cloudlets.

$$T_{Avg} = \frac{T}{N} \tag{5}$$

G. Execution Time of Priority Tasks

The execution time of priority tasks refers to the total time taken by a cloud model to complete the execution of all the cloudlets belonging to a specific priority. In this experiment, all level-1 priority, level-2 priority and level-5 priority tasks are considered for comparing and analyzing the performance of the proposed algorithm against the existing algorithms. This QoS parameter is the most important in the current experiment since it is the main motive of the proposed algorithm to prove its effectiveness compared to other existing algorithms

To find the best possible number of VMs for the simulation, performance evaluation of different algorithms with different VMs are performed as shown in Table V, VI and VII. From these tables, it is evident that the performance is better with 50 VMs in all the compared algorithms. So, for the rest of the experiments, the same set of VMs are considered for performance evaluation.

H. Experiment Results and Performance Analysis

The performance of the priority weighted RR algorithm has been implemented, and the analysis was performed using the CloudSim tool. The implementation is achieved by extending the classes already available in the simulation tool. The following illustrations of the execution cost, total completion time, total waiting time, total turnaround time, average execution time of a cloudlet, execution time of level-1 priority tasks, execution time of level-2 priority tasks and, execution time

TABLE VII
PWRR PERFORMANCE WITH DIFFERENT VMS

QoS	20 VMs	25 VMs	35 VMs	50 VMs	60 VMs	70 VMs	80 VMs
Execution Cost	27511.12	28108.5	27964.5	22299.42	23423.5	23789.1	23791.4
Tot. Comp. Time	9080.86	9296.69	9259	7388.14	7392.72	7421.32	7465.21
Avg. Exec. Time	3.68	3.71	3.7	3.67	3.68	3.68	3.69

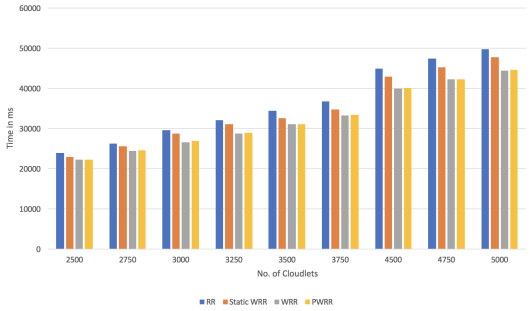

Fig. 2. Execution Cost of RR, WRR, DWRR and PWRR

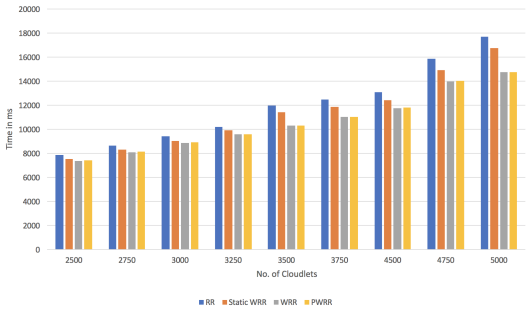

Fig. 3. Total Completion Time (in ms) of RR, WRR, DWRR and PWRR

of level-5 priority tasks are analyzed in round-robin (RR), weighted round-robin (WRR), dynamic weighted round-robin (DWRR), and priority weighted round-robin (PWRR) algorithms in heterogeneous conditions.

The execution cost of the all the algorithms is calculated with the cloud model of 50 VMs and different sets of heterogeneous cloudlets. Figure 2 shows the comparative analysis of all the algorithms. The proposed PWRR experiences a similar level of execution cost when compared to the DWRR.

Total completion times (in ms) of all the algorithms were calculated with the cloud model of 50 VMs and different sets of heterogeneous cloudlets. Figure 3 shows the analysis of all the algorithms. The proposed PWRR took the same amount of time to complete all the tasks when compared to DWRR.

Total waiting times (in ms) of the all the algorithms were calculated with the cloud model of 50 VMs and different sets of heterogeneous cloudlets. Figure 4 shows the comparative analysis of all the algorithms. The proposed PWRR performed better when compared to the DWRR.

Total turnaround times (in ms) of the all the algorithms were calculated with the cloud model of 50 VMs and different sets of heterogeneous cloudlets. Figure 5 shows the comparative analysis of all the algorithms. The proposed PWRR performed better when compared to the DWRR.

Average execution times of a cloudlet (in ms) for all the algorithms were calculated with the cloud model of 50 VMs and different sets of heterogeneous cloudlets. Figure 6 shows the comparative analysis of all the algorithms. The proposed PWRR took the same amount of time to complete a cloudlet when compared to the DWRR.

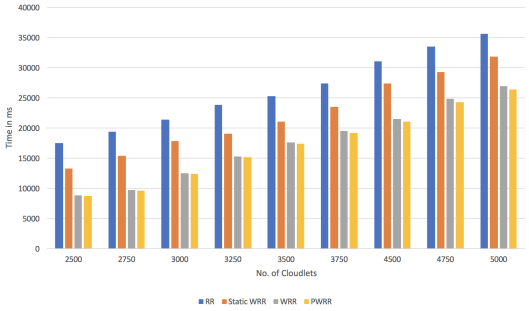

Fig. 4. Total Waiting Time (in ms - Time Shared) of RR, WRR, DWRR and PWRR

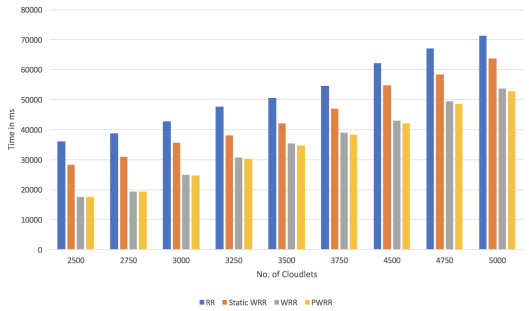

Fig. 5. Total Turnaround Time (in ms - Time Shared) of RR, WRR, DWRR and PWRR

The time taken to complete level 1, level 2 priority tasks and level 5 priority tasks (in ms) of the all the algorithms was calculated with the cloud model of 50 VMs and different sets of heterogeneous cloudlets. Figure 7, Figure 8 and Figure 9 show the comparative analysis of all the algorithms. The proposed PWRR shows a significant difference in completing the high priority tasks when compared to the DWRR. The time taken by PWRR is so much lower than RR, WRR and DWRR.

From the results and analysis, the proposed PWRR algorithm in terms of execution cost, total time taken to execute all the cloudlets, waiting time, turnaround time, and the average time taken to complete a single cloudlet is similar to that of the DWRR algorithm, that is widely used as a scheduling algorithm by popular cloud providers. But when it comes

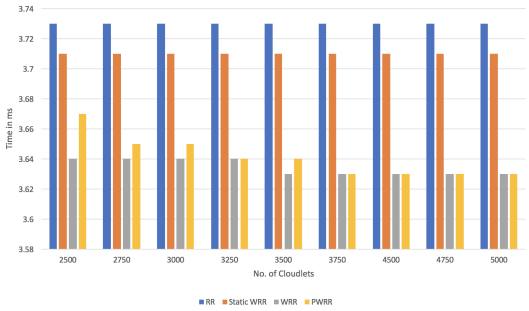

Fig. 6. Average Execution Time of a Cloudlet (in ms) of RR, WRR, DWRR and PWRR

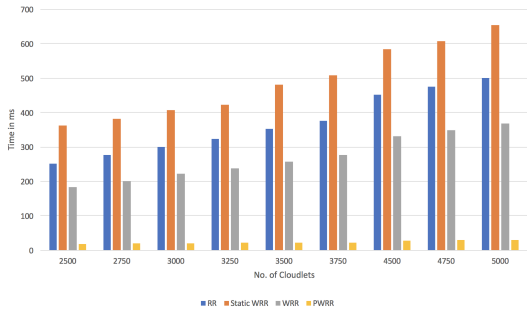

Fig. 7. Completion Time of Level 1 Cloudlets (in ms) of RR, WRR, DWRR and PWRR

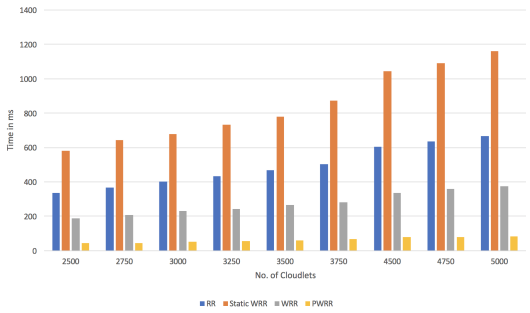

Fig. 8. Completion Time of Level 2 Cloudlets (in ms) of RR, WRR, DWRR and PWRR

to handling the tasks with priorities, the PWRR algorithm outperforms the DWRR algorithm by a noticeable margin.

VI. Conclusion

This work provides a new scheduling algorithm, the PWRR algorithm, to efficiently handle the incoming tasks of different priorities. In general, it performs better than the RR and WRR algorithms in terms of QoS parameters: execution cost, total waiting time, total turnaround time, average time taken to complete a single task, and total time taken to complete all tasks. It also performs equivalent to the DWRR algorithm, which is widely used by cloud providers for task scheduling, in terms of the above QoS parameters. In terms of handling the tasks with priority, the PWRR algorithm shows significant improvement when compared to all the three (RR, WRR, and DWRR) algorithms. From the results, it can be concluded that the PWRR algorithm performs equivalent to the DWRR algorithm in all the QoS factors, but it shows significant improvement over the DWRR algorithms when handling high-priority tasks. Hence, the PWRR algorithm is more suitable than the DWRR algorithm for task scheduling, which has the ability to handle tasks with and without priority.

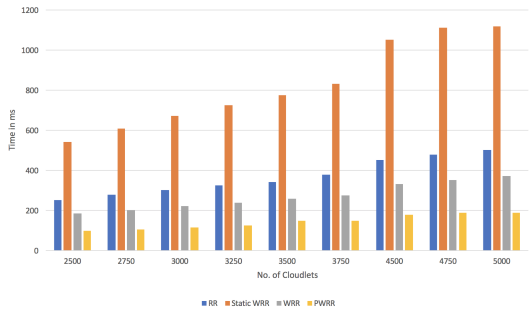

Fig. 9. Completion Time of Level 5 Cloudlets (in ms) of RR, WRR, DWRR and PWRR

References

[1] M. Rahman, S. Iqbal, and J. Gao, "Load balancer as a service in cloud computing," in *2014 IEEE 8th International Symposium on Service Oriented System Engineering*, April 2014, pp. 204–211.

[2] R. Support, "Understanding the cloud computing stack: Saas, paas, iaas," 2017, [Online; accessed 05-Nov-2017]. [Online]. Available: https://support.rackspace.com/whitepapers/understanding-the-cloud-computing-stack-saas-paas-iaas/

[3] A. K. sidhu and S. Kinger, "Analysis of load balancing techniques in cloud computing," in *International Journal of Advanced Research in Computers and Technology*, April 2013, pp. 737–741.

[4] H. singh and R. C. Gangwar, "Comparative study of load balancing algorithms in cloud environment," in *International Journal on Recent and Innovation Trends in Computing and Communication*, vol. 2, October 2014.

[5] M. Anicas, "An introduction to haproxy and load balancing concepts," 2014, [Online; accessed 26-July-2017]. [Online]. Available: https://www.digitalocean.com/community/tutorials/an-introduction-to-haproxy-and-load-balancing-concepts

[6] S. Joshi and U. Kumari, "Load balancing in cloud computing: Challenges issues," in *2016 2nd International Conference on Contemporary Computing and Informatics (IC3I)*, Dec 2016, pp. 120–125.

[7] P. Kaur and D. P. D. Kaur, "Efficient and enhanced load balancing algorithms in cloud computing," in *International Journal of Grid Distribution Computing*, vol. 8, 2015, pp. 9–14.

[8] G. Megharaj and D. M. K.G., "A survey on load balancing techniques in cloud computing," in *IOSR Journal of Computer Engineering (IOSR-JCE)*, vol. 18, April 2016, pp. 737–741.

[9] D. Puthal, B. P. S. Sahoo, S. Mishra, and S. Swain, "Cloud computing features, issues, and challenges: A big picture," in *2015 International Conference on Computational Intelligence and Networks*, Jan 2015, pp. 116–123.

[10] M. B. J and K. B. R, "Review on round robin algorithm for task scheduling in cloud computing," in *International Journal of Emerging Technologies and Innovative Research*, vol. 2, March 2015, pp. 788–793.

[11] M. M. Alipour and M. R. F. Derakhshi, "Two level fuzzy approach for dynamic load balancing in the cloud computing," in *Journal of Electronic Systems*, vol. 6, March 2016, pp. 17–31.

[12] D. C. Devi and V. R. Uthariaraj, "Improved round robin algorithm for data center selection in cloud computing," in *INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES AND RESEARCH TECHNOLOGY*, vol. 2014, February 2014.

[13] D. Devi and V. R. Uthariaraj, "Load balancing in cloud computing environment using improved weighted round robin algorithm for non-preemptive dependent tasks," in *The Scientific World Journal)*, vol. 2016, October 2016.

[14] G.THEJESVI and T.ANURADHA, "Distribution of work load at balancer level using enhanced round robin scheduling algorithm in a public cloud," in *International Journal of Advances in Electronics and Computer Science*, vol. 3, Feb 2016.

[15] A. Manirabona, S. Boudjit, and L. C. Fourati, "A priority-weighted round robin scheduling strategy for a wban based healthcare monitoring system," in *2016 13th IEEE Annual Consumer Communications Networking Conference (CCNC)*, Jan 2016, pp. 224–229.

[16] G. T. Hicham and E. A. Chaker, "Cloud computing cpu allocation and scheduling algorithms using cloudsim simulator," in *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 6, August 2016, pp. 1866–1879.

[17] H. G. Tani and C. E. Amrani, "Smarter round robin scheduling algorithm for cloud computing and big data," in *Journal of Data Mining and Digital Humanities*, Jan 2017.

[18] R. D. 1 and S.Sujan, "A survey on application of cloudsim toolkit in cloud computing," in *International Journal of Innovative Research in Science, Engineering and Technology*, vol. 3, June 2014, pp. 3–4.