# A Load Balancing Algorithm Based on Processing Capacities of VMs in Cloud Computing

**Ramandeep Kaur and Navtej Singh Ghumman**

**Abstract** Cloud Computing is a computing paradigm which has made high-performance computing accessible even to SMEs (small and medium enterprises). It provides various types of services to the users in the form of hardware, software, application platforms. The cloud computing environment is elastic and heterogeneous in nature. Any number of users may join/leave the system at any point of time; it means the workload of the system increases/decreases randomly. Therefore, there is a requirement for a load balancing system which must ensure that the load of the system is fairly distributed among the nodes of the system and aims to achieve minimum completion time and maximum resource utilization. The paper presents a load balancing algorithm based on processing capacities of virtual machines (VMs) in cloud computing. It analyses the algorithm and finds the research gap. It also proposes the future work overcoming the research gap in this field. The simulation of the algorithm is carried out in the CloudSim simulation toolkit.

**Keywords** Cloud computing · Challenges · Load balancing · Existing algorithms · Round-robin · Heuristic approach · Processing capacities

## 1 Introduction

Cloud computing is an emerging technology which permits users to access their stored data, applications and avail various types of cloud services via Internet. The cloud provides users' with various types of services such as physical hardware,

R. Kaur (✉) · N.S. Ghumman
Department of Computer Science and Engineering, SBSSTC,
Ferozepur, Punjab, India
e-mail: ramandipkaur23@gmail.com

N.S. Ghumman
e-mail: navtejghumman@yahoo.com



V.B. Aggarwal et al. (eds.), *Big Data Analytics*, Advances in Intelligent Systems and Computing 654, https://doi.org/10.1007/978-981-10-6620-7_8

programming tools and environments, softwares, business-related applications. These various types of services are as well known by the name, "service models of cloud".

### *1.1 Open Challenges [1]*

- **Load Balancing**: It is a process of balancing the workload among the nodes of the system in order to achieve minimum response time and higher resource utilization.
- **Interoperability and Portability**: Interoperability is the ability to use the same tools and applications across various different cloud service providers' platforms. Portability can be the solution to address this issue. It says that one cloud solution works with various different types of applications, platforms, and other clouds also.
- **Resource Scheduling and Management**: It deals with the resource provisioning and management in order to provide the desired quality of service to a user.
- **Security and Privacy of Users' Data**: The users' data are kept with some third party by the service provider. So, how the cloud service provider addresses the security and privacy concerns is an issue.

## 2 Related Work

There are various types of load balancing techniques which are classified as static and dynamic.

- **Static Load Balancing Techniques**.

The static load balancing techniques are of static nature, which does not adapt themselves to the changing computing environment. Round-robin algorithm, min-min algorithm, max-min algorithm are static load balancing techniques. Static techniques perform well in a homogeneous and predictable computing environment.

- **Dynamic Load Balancing Techniques**.

The dynamic load balancing techniques are of dynamic nature, which adapts themselves to the changing environment. They follow the various policies for the load distribution in a dynamic heterogeneous computing environment. Honey bee Algorithm, active clustering, random biased sampling is dynamic load balancing algorithms.

### 2.1 Performance Metrics for Load Balancing Techniques

- **Response Time**: It is the time after which user's task is completed. It is calculated as subtracting total execution time from the start time. It should be minimized for better performance.
- **Fault Tolerance**: It is the ability to recover from failure by shifting the workload to some other remote working node. Thus, the load balancing algorithm should be highly fault tolerant.
- **Scalability**: It is the ability of an algorithm to perform load balancing for a finite number of nodes. In cloud system, number of users may increase or decrease dynamically and correspondingly the load is dynamic. Therefore, the load balancing algorithm should be highly scalable.

### 2.2 Existing Algorithms

There are various existing load balancing algorithms. The paper [2] proposes a round-robin algorithm, in which the tasks are allocated the virtual machines (VMs) for a fixed time quantum. The paper [3] proposed a min-min algorithm, in which smaller tasks are scheduled before the longer tasks. The paper [4] proposed a dynamic algorithm based on exponential smoothing forecast. It is employed to balance the load of real servers and takes into consideration the unique features of the long connectivity applications.

## 3 Load Balancing Algorithm

The task is actually a users' request which is independent and computational one. Let $k$ be any virtual machine and "$r_i$" be any $i$th task (or cloudlet). As soon as the tasks are submitted, the load balancer first calculates the utilized power PW($k$) of each VM using (1). The task is allocated the virtual machine (VM), which is more powerful in terms of processing capacity. Here, VM with lower value of power is more powerful. Then the task, "$r_i$" is dispatched to the selected VM. After the successful completion of the task, load balancer updates the power PW($k$) of that VM using (1). In case of ties, the tasks are scheduled according to first come first serve (FCFS) basis.

The execution of the tasks continues until there is no task left for execution. The response time, average response time is calculated using (2) and (3) and total turnaround time (TAT) is calculated using (4).

$$\mathrm{PW}(k) = \mathrm{PW}(k) + \mathrm{CPU}(r_i)^{*}\mathrm{Size}(r_i)/\mathrm{CPU}(k) \quad (1)$$

$$\mathrm{RT}(r_i) = \mathrm{FT}(r_i) - \mathrm{ST}(r_i) \quad (2)$$

$$\mathrm{ART} = \sum_{i=1}^{i=n} \mathrm{RT}(r_i)/n \quad (3)$$

$$\mathrm{TAT} = \mathrm{FT} - \mathrm{AT}, \quad (4)$$

where,

| | |
|---|---|
| $k$ | any $k$th virtual machine. |
| $r_i$ | any $i$th task ($i$ = 1 to $n$). |
| $n$ | total number of tasks. |
| Size | task length/number of instructions. |
| CPU ($r_i$) | number of processing elements required by the task. |
| CPU ($k$) | number of processing elements required by the virtual machine (VM). |
| ST | start time of task |
| FT | finish time of task |
| AT | arrival time of task |

## 3.1 Illustrative Example

Here we are going to discuss the working of the load balancing algorithm based on processing capacities of VMs with the help of an illustrative example. Here, we take four virtual machine, namely, VM0, VM1, VM2, VM3 of varying MIPS. Table 1

**Table 1** Shows the characteristics of cloudlets during simulation

| Cloudlet ID | Cloudlet length | File size | Output size | No. of processing elements (CPU) |
|---|---|---|---|---|
| C0 | 1000 | 1300 | 300 | 1 |
| C1 | 3000 | 1300 | 300 | 1 |
| C2 | 5000 | 1300 | 300 | 1 |
| C3 | 7000 | 1300 | 300 | 1 |
| C4 | 4000 | 1300 | 300 | 1 |
| C5 | 3500 | 1300 | 300 | 1 |
| C6 | 8000 | 1300 | 300 | 1 |
| C7 | 9000 | 1300 | 300 | 1 |
| C8 | 3000 | 1300 | 300 | 1 |
| C9 | 4500 | 1300 | 300 | 1 |

**Table 2** Shows the power of VMs before first cloudlet execution

| VM ID | VM0 | VM1 | VM2 | VM3 |
|---|---|---|---|---|
| Utilized power PW($k$) | 0 | 0 | 0 | 0 |

**Table 3** Shows the power of VMs during the execution of 10 cloudlets

| VM ID | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| PW($k$) | 1000 | **0** | **0** | **0** |
| PW($k$) | 1000 | 3000 | **0** | **0** |
| PW($k$) | 1000 | 3000 | 5000 | **0** |
| PW($k$) | **1000** | 3000 | 5000 | 7000 |
| PW($k$) | 5000 | **3000** | 5000 | 7000 |
| PW($k$) | **5000** | 6500 | **5000** | 7000 |
| PW($k$) | 13,000 | 6500 | **5000** | 7000 |
| PW($k$) | 13,000 | **6500** | 14,000 | 7000 |
| PW($k$) | 13,000 | 9500 | 14,000 | **7000** |
| PW($k$) | 13,000 | 9500 | 14,000 | 115,000 |

shows the characteristics of the cloudlets during simulation in this example. We have taken 10 cloudlets having lengths in a range of 1000–10,000. We have assumed that initially the utilized power of each VM is zero. Also, the virtual machine with low utilized power is more powerful.

Here, we know all the virtual machines have the same power. Therefore, in such cases FCFS is used. Tables 2 and 3 show the execution result.

## 4 Simulation Framework and Results

We have used CloudSim [5] as a simulation framework with Netbeans IDE. It is a simulation toolkit to model and simulate the Cloud Computing environment. We have employed four Virtual Machines, namely, VM0, VM1, VM2, VM3 having varying MIPS. We have conducted the experiments to observe the change in average response time and total turnaround time and compared it with round-robin algorithm.

We have obtained the following graphical results by conducting the experiments for 500, 2000, 5000, and 20,000 cloudlets (or tasks) using round-robin algorithm and heuristic load balancing algorithm. Figures 1 and 2 shows the variation of average response time and turnaround time in the case of round-robin approach and heuristic approach respectively.

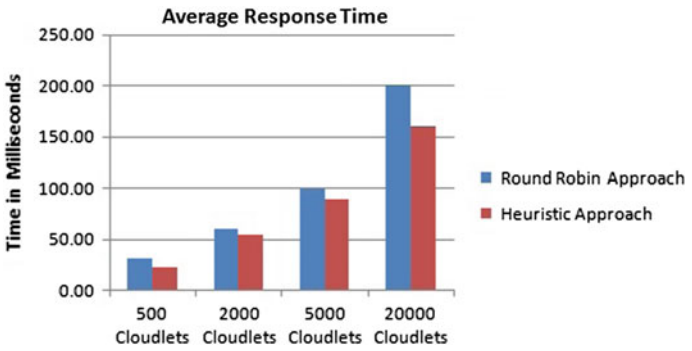


**Fig. 1** Shows the variation in average response time in the case of Round-Robin approach and Heuristic approach

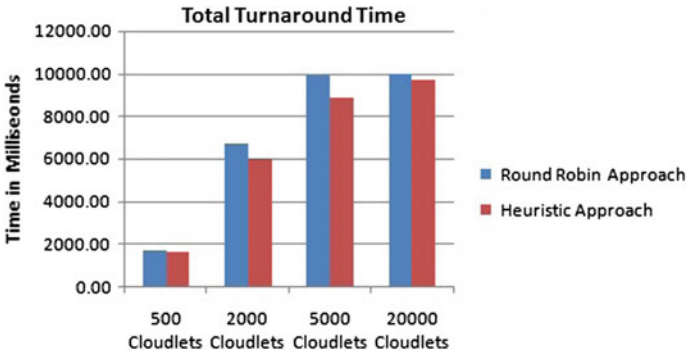


**Fig. 2** Shows the variation in average response time in the case of Round-Robin approach and Heuristic approach

## 5 Research Gap and Proposed Future Work

The heuristic algorithm [6] is based on centralized load balancing strategy. However, the algorithm does not consider the MIPS of the VMs. The actual processing capacities of the VMs in terms of MIPS as well as the utilized power of the VMs would be taken into account in the proposed future work.

The proposed future work would group the tasks as well as the virtual machines into two groups, namely upper class and lower class. The grouping in the tasks would take place on their length as the threshold value and the grouping of the virtual machine would take place in terms their MIPS value as the threshold value. When grouping of the tasks and the virtual machine is done, the tasks of the lower class group are sent to the VMs of the lower class group and vice versa. Then, the task of corresponding group is sent to the virtual machine of corresponding based

on their utilization power of the VMs. Thus, proposed work aims to achieve improvement in response time, processing cost of the algorithm, and the total turnaround time.

## 6 Conclusions

Load balancing is a serious issue in cloud computing. The paper attempts to highlight the issue of load balancing. The paper analyses a heuristic load balancing algorithm and also compares its performance with round-robin algorithm. Finally, this paper highlights the research gap and gives an approach to overcome it.

## References

1. Sajid, M., Raza, Z.: Cloud computing: issues & challenges. In: International Conference on Cloud, Big Data and Trust (2013)
2. Mohapatra, S., Rekha, K.S., Mohanty, S.: A comparison of four popular heuristics for load balancing of virtual machines in cloud computing. Int. J. Comput. Appl. **68**(6):33–38 (2013)
3. Elzeki, O.M., Reshad, M.Z., Elsoud, M.A.: Improved max-min algorithm in cloud computing. Int. J. Comput. Appl. **50**(12):22–27 (2012)
4. Ren, X., Lin, R., Zou, H.: A dynamic load balancing strategy for cloud computing platform based on exponential smoothing forecast. In: 2011 IEEE International Conference on Cloud Computing and Intelligence Systems (CCIS), pp. 220–224, 15–17 Sept 2011
5. Calheiros, R.N., et al.: CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Softw. Pract. Experience **41** (1):23–50 (2011)
6. Haidri, R.A., Katti, C.P., Saxena, P.C.: A load balancing strategy for cloud computing environment. In: 2014 International Conference on Signal Propagation and Computer Technology (ICSPCT), pp. 636–641, IEEE (2014)