# MSRTIA: A PROPOSAL TO REDUCE THE RESPONSE TIME FOR LOAD BALANCING ON CLOUD COMPUTING

**Tran Cong Hung*, Nguyen Ngoc Thang[+], Kieu Trong Duc[+]**
*Posts and Telecommunications Institute of Technology
[+]Saigon University

***Abstract:*** Cloud computing is a model that provides everything related to information technology in the form of services through the Internet. The primary benefit of it is to save the original system investment cost, optimize the data processing, calculating and storing data. Nowadays, cloud computing faces many challenges in ensuring the quality of service throughout. In which the problem of overloading physical servers or virtual servers of data centres is concerned specially. So as to qualify the above requests, setup an effective load balancing method and using resources with the most optimization is the target which cloud computing wants to gain. In this paper, we propose Max-Min Scheduling Response Time Improved Algorithm (MSRTIA) basing on Max Min Scheduling algorithm. Our algorithm calculates the Cloudlet aggregation value of requests then pairs the request with the largest value found with the fastest executing virtual machine (VM). In which, Cloudlet aggregation value is the association of length, output size and file size parameters. The simulation result proves that MSRTIA has less response time in comparison with Max-Min Scheduling and Round-Robin algorithms.

***Keywords:*** Cloudlet aggregation value, Max-Min Scheduling, Round-Robin

## I. INTRODUCTION

Cloud computing is a growing technology today. With the advent of cloud computing, a new era of Internet has taken birth. The uses of cloud computing exceed more than any application or service that has run on the Internet so far. Even though the number of users on the Internet are large compared to those using the cloud computing facility, the rate of growth of users interested in cloud computing is going up. Many corporations like Amazon, Google, Alibaba, Microsoft and other small scale enterprises, have already started using the cloud and are giving the cloud facility to their customers [1].

Cloud computing [2] is a wide research direction, an information technology service model, inherited from previous networks, helping users store and access to complex data systems quickly and easily.

Contact author: Tran Cong Hung,
Email: conghung@ptithcm.edu.vn
Arrival: 3/2020, Revised: 4/2020, Accepted: 4/2020.

Cloud computing bases on virtualization technology [3], through network to provide users different services. It makes use of computer resources on demand such as bandwidth, storage or software applications rapidly and flexibly.

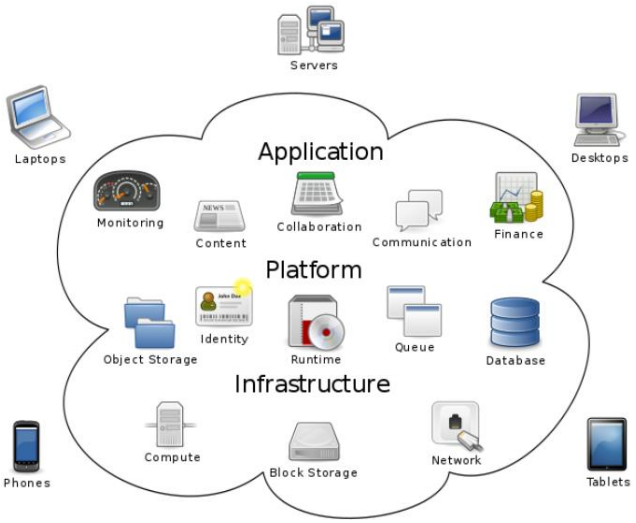



*Figure 1. Cloud Computing overview.*

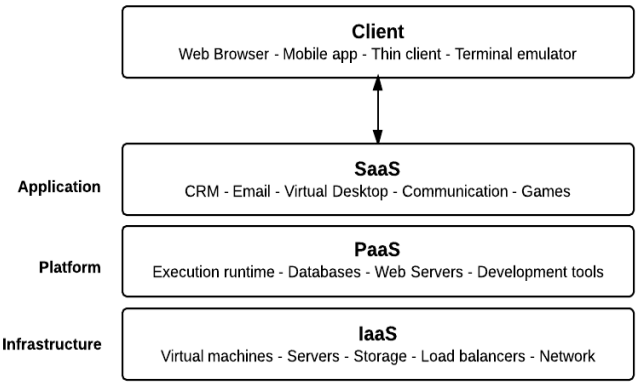



*Figure 2. Cloud Computing Service Models.*

The above figure shows the different cloud computing service models:

Infrastructure as a Service (IaaS): The service requires computation of resources (RAM, CPU, GPU, Disk) and is scalable on a fast-changing basis. [4] [5].

Platform as a Service (PaaS): PaaS provides an environment for developers that they can build and use to create applications that can be customized and continue to develop. They do not need to care about the network, storage, server, and operating systems below. [4] [5].

Software as a Service (SaaS): The service provider provides the full stack. The subscriber merely consumes the cloud service [4] [5].

Cloud computing is becoming increasingly popular with incredible speed. The transition between the old system to this new system is completely outside the general trend of progress. Businesses that do not follow the trend will face the risk of rejection. Load Balancing is a very important technology feature in the computer networking industry, helping virtual servers to operate more synchronously and efficiently through the uniform distribution of resources. [6].

Load balancing is a method of distributing the load on multiple computers or a cluster of computers so that resources can be used optimally, maximizing throughput, reducing response time and avoiding over-loading server [7]. Due to the benefits and applications of cloud computing, we suggest an improvement of Max Min Scheduling algorithm to reduce the response time and enhance the effectiveness for load balancing. It will calculate the Cloudlet aggregation value of requests and allocate the request with the maximum value found for virtual machine having minimum executing time. More details will be in next parts. This paper is organized as follows. Section I presents Introduction, Section II presents Related works. Section III Proposal. Next section gives out the details of Simulation and Evaluation. And section V Conclusion.

## II. RELATED WORK

In order to optimize the resources available in the network and achieve less executing time, needs to provide a new scheduling algorithm is crucial [8]. These algorithms assign tasks to the resources and provide the best conditions for quality of services.

The Round-Robin algorithm (RR) [9] [10] is one of the most famous and widely used algorithms, especially designed for Time-shared systems. Because the Round-Robin algorithm is a ring-standard algorithm, it records the number of connections established per server. Once a request is scheduled for the server, that server is added to the number of connections. When all servers have the same processing performance, RR can send the load variation to the same server. However, when dealing with all servers that their capacity is not the same, this algorithm is not ideal because it will turn to the standby status after sending the connection to request processing, the standby status is usually 2 minutes, connecting at this point also takes up server resources. Thus, the response time of the processing request is not highly effective for virtual machines with different processing capacity in the system and can lead to idle on virtual machines with more powerful processing power than the remaining virtual machines.

Weighted Round-Robin algorithm [11] inherits the advantage of circular circle distribution of Round-Robin algorithm. Weighted Round-Robin algorithm will take into account the different processing performance of different servers. Processing requests or service applications are delivered to the virtual machine in alternate circular order. It also incorporates the processing power of each virtual machine based on the available weighting tables. So virtual machines that have more powerful processing power will be allocated more tasks to handle than the remaining virtual machines. Thus, it is possible to allocate virtual machines to handle tasks in a more optimal way, helping to improve response time in the load balancing system. However, when there is a large change in service request time, this algorithm has a single weight that can lead to load unbalance between servers, they do not consider the processing time for each request. This algorithm does not rely on the current load status information of each virtual machine, only based on information known before the load distribution. Therefore, it has not achieved high efficiency in the process of load balancing.

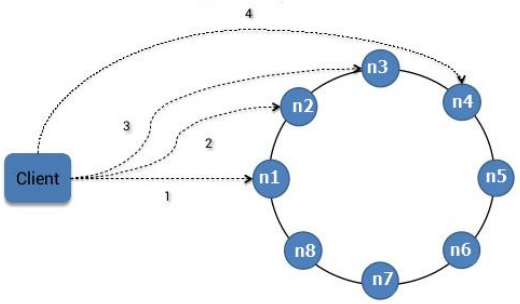



*Figure 3. Round-Robin algorithm*

Throttled algorithm load balancer uses a single job scheduler, which makes it centralized in nature. The job scheduler maintains a table named VM allocation table, which stores the id and status of all the VMs. A VM can have only two states: occupied or idle, denoted by 1 or 0 respectively in the array. Initially, all VMs are idle. On receiving a task, job scheduler searches the VM which is not busy. If it finds an idle virtual machine, then it assigns the task to that VM. If no VMs are available to accept, the task has to wait in job scheduler's queue. No queues are maintained at the VM level. A VM can accommodate only one task and another task can be allocated only when the current task has finished [12]. With this algorithm, it takes less time to respond than the Round Robin algorithm. But there is a restriction that each allocation must detect the "0" ready virtual machine in the entire index table, including the large index.

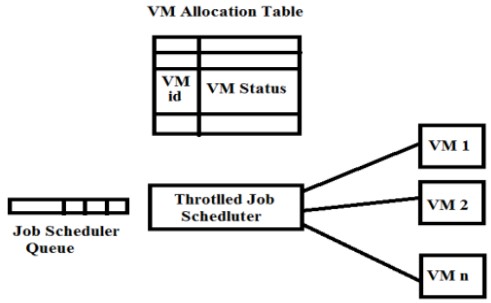



*Figure 4. Throttled algorithm*

In Max-Min algorithm [13] [14], cloud manager chooses to allocate virtual machines according to Max-Min rules. When a request comes, it will be put into the queue. By Max-Min rule, cloud manager will choose which requests in the queue have the time to be allocated the most resources. Then select the virtual machine capable of completing this request as quickly as possible to assign the task. Selecting which requests need to be allocated the most resources by scanning the queue and selecting the max value. Virtual machine capable of the fastest finishing work based on the state of the virtual machine table. Limitation: Although it has improved the response time, completion time of the requests, minimizing the load unbalance in the system, this algorithm still has the disadvantage in calculating the completion time required by a virtual machine. When receiving a request, the virtual machine must recalculate the expected completion time for that request, similar for other requests, and then find the minimum expected completion time of virtual machine for each request. This leads to the expected completion time of a virtual machine will consume lots of time and processing costs.

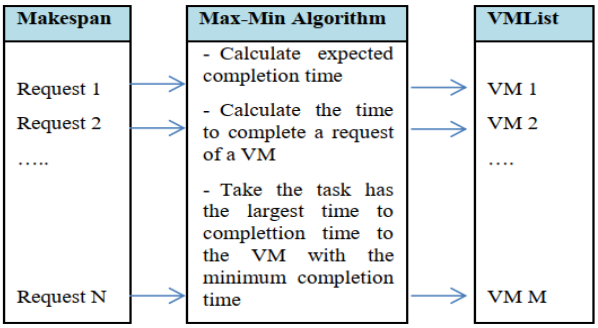



*Figure 5. Max-Min Scheduling algorithm operation*

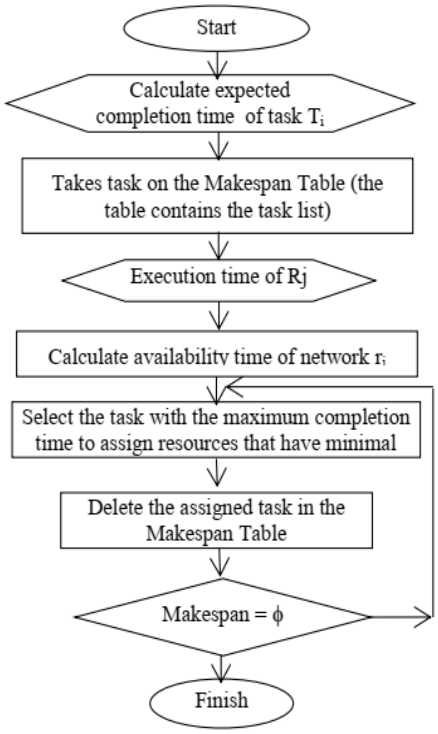



*Figure 6. Max-Min Scheduling algorithm schema*

## III. PROPOSAL

### MSRTIA: Max-Min Scheduling Response Time Improved Algorithm

#### Assumption

MSRTIA supposes the requests need to process on VMs with the same memory and different CPUs and MIPS.

#### Target

- Reduce the respond time in comparison to the Max-Min Scheduling and Round-Robin algorithms.
- Reduce the processing time of input requests.
- Increase the processing capacity of VMs but not losing the load balancing of the system.

#### Model

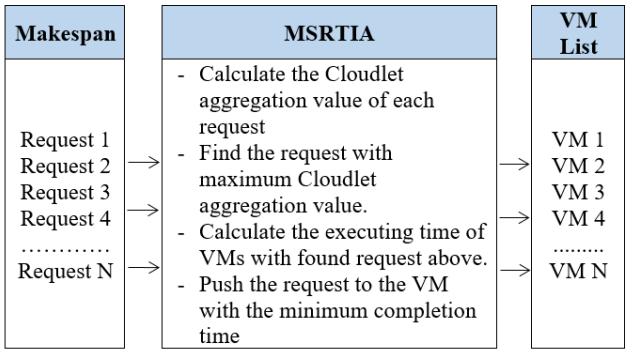



*Figure 7. MSRTIA based on Cloudlet aggregation value*

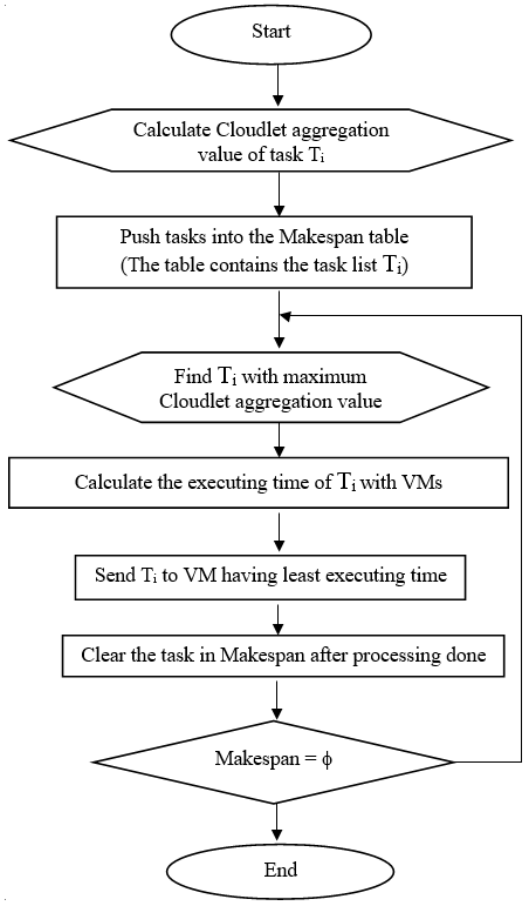



*Figure 8. MSRTIA schema*

Description of the MSRTIA algorithm:

The idea is to calculate the Cloudlet aggregation value of requests, chooses the request with largest value, then pairs with the fastest executing VM following the formula:

```
public double getAverageSize(Cloudlet cloudlet) {
double resAverage;
resAverage = cloudlet.getCloudletLength() * 0.5 +
cloudlet.getCloudletOutputSize() * 0.2 +
cloudlet.getCloudletFileSize() * 0.3;
return resAverage; }
```

The MSRTIA algorithm will repeat until the *Makespan* tables are empty. At that time the requests will be processed faster, shorten the finishing time, increased load balancing capability for cloud computing.

The response time is explained in details as below.

**Basis of assessment**

The efficiency of load balancing can be based on many factors, but the most important factors are loading and performance. Loading is the CPU queue index and utilization [15]. Performance is the average response time to user requests. Load balancing algorithm is based on input parameters such as: configuring virtual machines, configuring Cloudlet tasks, arrival time, and time to complete tasks, then estimating the expected completion time of each task. Response time is the processing time plus the cost of transmitting request, queuing through the network nodes. Expected response time is calculated according to the following formula [16]:

$$Expected\ Response\ Time = F - A + T_{delay}$$

*F: time to complete the task, A: arrival time of the task, $T_{delay}$: transfer time of the task*

Because the algorithm that performs load balancing is that of *DatacenterBroker*, the level of the algorithm only affects the processing time in a local environment of a data centre. Hence the communication delay parameter can be omitted, so $T_{delay} = 0$.

Determining the expected time to complete task [16]:

If the scheduling policy is Space shared – Space shared or Time shared - Space shared, it is determined by the following formula:

$$eft(p) = est(p) + \frac{rl}{capacity \times cores(p)} \quad (3.1)$$

$$capacity = \sum_{i=1}^{np} \frac{cap(i)}{np} \quad (3.2)$$

If the scheduling policy is Space shared – Time shared or Timeshared-Timeshared, it is determined by the formula

$$eft(p) = ct + \frac{rl}{capacity \times cores(p)} \quad (3.3)$$

$$capacity = \frac{\sum_{i=1}^{np} cap(i)}{\max(\sum_{j=1}^{cloudlets} cores(j), np)} \quad (3.4)$$

- *eft(p)* is the expected completion time of Cloudlet p;
- *est* is the arrival time of Cloudlet p;
- *rl* is the total number of instructions that Cloudlet p needs to execute on a processor;
- *capacity* is the average processing power (in MIPS) of a core for Cloudlet p;
- *ct* is the current simulation time;
- *cores (p)* is the number of cores required by Cloudlet;
- *np* is the number of actual cores that the host is considering;
- *cap* is the processing power of the core.

The *capacity* parameter specifies the actual capacity for task processing on each VM. Apparently capacity depends on the scheduling of computing resources on the virtualized system. The total processing power on a physical host is unchanged and depends on the number of physical cores and processing power of each core. However, when this processing resource is shared for multiple tasks simultaneously, each task requires a certain number of cores and if the total number of cores is greater than the number of physical cores, the concept of virtual core appears, each virtual core will have lower processing power than the physical core. In other words, the capacity of a virtual core for a task can only be equal to or smaller than the physical core and how much depends on the resource sharing policy. Capacity is the processing power of a virtual core [15] [16].

From this analysis and based on the resource sharing policy to develop formulas for capacity. Resource sharing policy is specified through scheduling mechanism in cloud computing. We have two levels of scheduling: scheduling virtual machines to share physical host machine resources and scheduling tasks to share virtual machine resources. There are two scheduling mechanisms: Time shared and Space shared. Within the scope of this paper, we will perform algorithms and simulations based on the Time shared – Time shared policy, respectively, to virtual machines and tasks. Therefore, the calculation base for the proposed algorithm will be based on the formulas (3.3) and (3.4).

## IV. SIMULATION & EVALUATION

Cloud environment emulator uses CloudSim 3.0 library and programming in JAVA language, includes 1 to 4 VMs. It will create a random request environment for services on the cloud containing virtual cloud services, CloudSim provisioning and user provisioning services for testing [17].

*Table 1. Data center configuration parameters*

| Data Center | Host in Data center |
|---|---|
| - Number of host in datacenter: 4<br>- Do not use SAN Storage<br>- Architecture (arch): x86<br>- Operating System (OS): Linux<br>- Processing (VMM): Xen<br>- Time Zone: +7 GMT<br>- Cost of processing: 3.0<br>- Cost of memory usage: 0.05<br>- Cost of using capacity: 0.1<br>- Cost of bandwidth usage: 0.1 | Each host in the datacenter is configured:<br>- The CPU has 4 cores, each core has a processing speed of 1000 (mips).<br>- RAM: 2048 (MB)<br>- Storage: 1000000<br>- Bandwidth: 10000 |

*Table 2. VM configurations parameters when initialized*

| Capacity | RAM | Mips | Bandwidth | CPU | VMM |
|---|---|---|---|---|---|
| 10000 MB | 512 MB | 100 | 1000 | 1 | Xen |
| 10000 MB | 512 MB | 250 | 1000 | 2 | Xen |
| 10000 MB | 512 MB | 300 | 1000 | 1 | Xen |
| 10000 MB | 512 MB | 220 | 1000 | 2 | Xen |

The requests (WebRequest) are represented by Cloudlet in CloudSim and the size of Cloudlets is randomly generated using the JAVA random function.

*Table 3. Requests configuration parameters*

| Length | File Size | Output Size | Number of CPU |
|---|---|---|---|
| 1700 ~ 3000 | 5000 ~ 45000 | 450 ~ 750 | 1 |

The function to create randomly 1000 requests in Table 2:

```
public DataInput() {
this.v_length =
ThreadLocalRandom.current().nextInt(1700, 3000);
this.v_fileSize =
ThreadLocalRandom.current().nextInt(5000,45000);
this.v_outputSize=
ThreadLocalRandom.current().nextInt(450,750);
}
```

**Result and Evaluation**

Experiments apply Timeshared – Timeshared scheduling policy for VM – task and calculate response time according to formula (3.3) and (3.4) as described in *Base of assessment.*

The simulation will make out 1000 requests with 4 times, each will have 4 VMs and the number of requests is 100, 200, 500 and 1000 respectively.

From the Figure 9-13, we can see that the response time of MSRTIA is less than Max-Min and Round-Robin for all scenarios with the number of requests from 10 to 1000. The more requests are tested, the better response time of MSRTIA demonstrates in comparison with Max-Min and Round-Robin. In other words, MSRTIA is effective especially for large amount of requests.

Through 4 experiments, it shows that MSRTIA has the response time for VMs better and load balancing more efficiently than Max-Min and Round-Robin scheduling techniques. Specifically, the response time of MSRTIA is faster 9.63% than Max-Min and 15.32% than Round-Robin algorithms.

Compared to the previous methods, the proposed algorithm doesn't need to perform the calculation for completion time of requests again. From which, MSRTIA will reduce unnecessary processing time and costs as well as minimize load unbalancing in the cloud system.

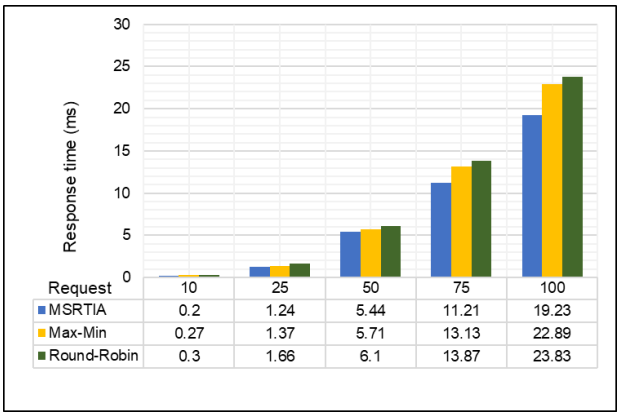
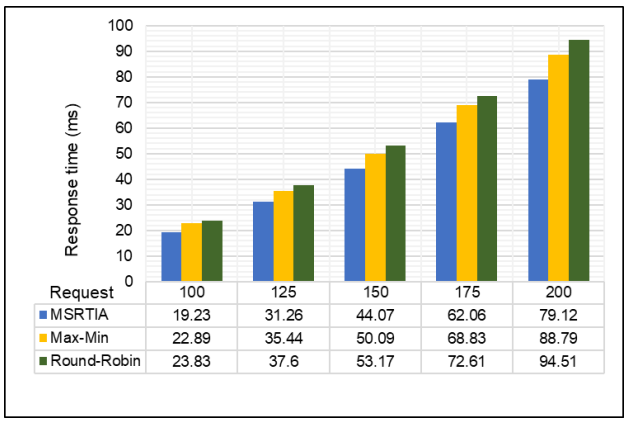
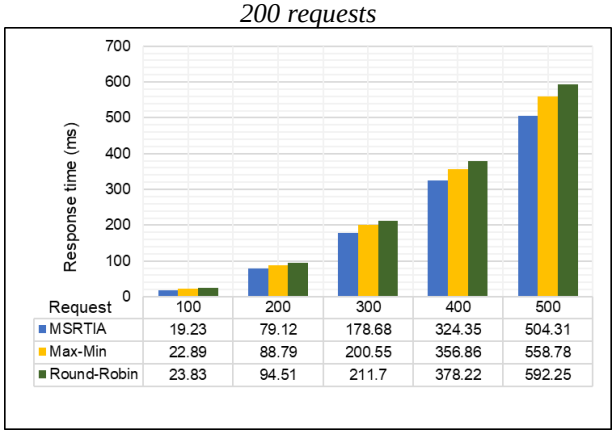



*Figure 9. Experimental result on 4 virtual machines with 100 requests*



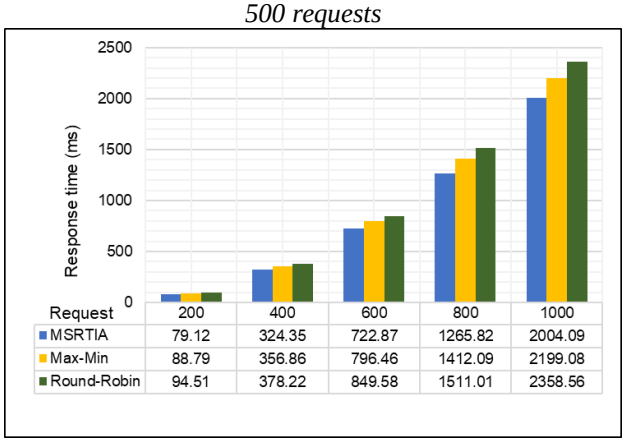

| Request | 100 | 125 | 150 | 175 | 200 |
|---|---|---|---|---|---|
| MSRTIA | 19.23 | 31.26 | 44.07 | 62.06 | 79.12 |
| Max-Min | 22.89 | 35.44 | 50.09 | 68.83 | 88.79 |
| Round-Robin | 23.83 | 37.6 | 53.17 | 72.61 | 94.51 |

*Figure 10. Experimental result on 4 virtual machines with 200 requests*




| Request | 100 | 200 | 300 | 400 | 500 |
|---|---|---|---|---|---|
| MSRTIA | 19.23 | 79.12 | 178.68 | 324.35 | 504.31 |
| Max-Min | 22.89 | 88.79 | 200.55 | 356.86 | 558.78 |
| Round-Robin | 23.83 | 94.51 | 211.7 | 378.22 | 592.25 |

*Figure 11. Experimental result on 4 virtual machines with 500 requests*




| Request | 200 | 400 | 600 | 800 | 1000 |
|---|---|---|---|---|---|
| MSRTIA | 79.12 | 324.35 | 722.87 | 1265.82 | 2004.09 |
| Max-Min | 88.79 | 356.86 | 796.46 | 1412.09 | 2199.08 |
| Round-Robin | 94.51 | 378.22 | 849.58 | 1511.01 | 2358.56 |

*Figure 12. Experimental result on 4 virtual machines with 1000 requests*

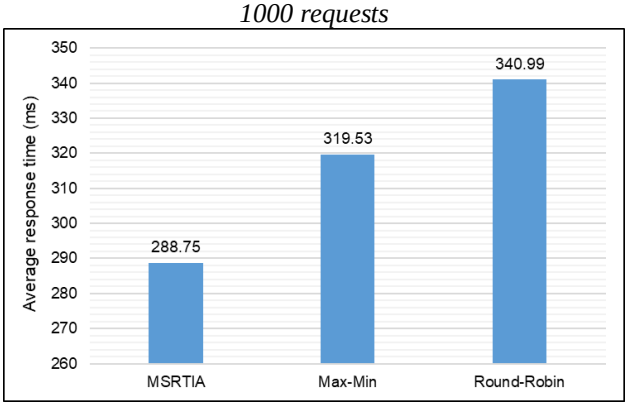



*Figure 13. Experimental result after 4 times counted on average*

## V. CONCLUSION

MSRTIA calculates the Cloudlet aggregation value of requests and searches for the request owning the maximum value then assign to VM having the minimum completion time. It is very clear to see the results that MSRTIA has ameliorated the response time for load balancing, optimized the performance compared to Max-Min and Round-Robin algorithms with the better rate 9.63% and 15.32% respectively.

For future research, the improvements may include the following: simulate the algorithm with more configuration cases in terms of data centres, VMs, different scheduling policies (Time shared – Space shared or vice versa), combined with other machine learning methods.

## REFERENCES

[1] Agraj Sharma, Sateesh K. Peddoju, (2014), "Response Time Based Load Balancing in Cloud Computing", *International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT).*

[2] J. Zhao, K. Yang, X. Wei, Y. Ding, L. Hu and G. Xu, (2016) "A Heuristic Clustering-Based Task Deployment Approach for Load Balancing Using Bayes Theorem in Cloud Environment", *in IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 2, pp. 305-316.

[3] J. Zhang, Q. Liu and J. Chen, (2016) "An Advanced Load Balancing Strategy for Cloud Environment", *International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT)*, Guangzhou, pp. 240-243.

[4] L. Pallavi, V. Pradeep Kumar, (2014), "Mobile Cloud Computing: Service Models", *International Conference on Computer & Communication Technologies, INDIA.*

[5] Mohammad UbaidullahBokhari, Qahtan Makki Shallal, YahyaKordTamandani, (2016), "Mobile Cloud Computing Service Models: A Comparative Study", *IEEE Network, Institute of Electrical and Electronics Engineers network.*

[6] Mohammad Riyaz Belgaum, Safeeullah Soomro, Zainab Alansari, Muhammad Alam, Shahrulniza Musa, Mazliham Mohd Su'ud, (2017), "Load Balancing with preemptive and non-preemptive task scheduling in Cloud Computing", *International Conference on Engineering Technologies and Social Sciences (ICETSS).*

[7] Divya Chaudhary, Rajender Singh Chhillar, (2013) "A New Load Balancing Technique for Virtual Machine Cloud Computing Environment", *International Journal of Computer Applications (0975 – 8887)*, Volume 69– No.23.

[8] Soheil Anousha, Mahmoud Ahmadi, (2013), "An Improved Min-Min Task Scheduling Algorithm in Grid Computing", *Conference Paper Analysis and Tools for Integrated Circuits and Systems pp.103-113.*

[9] Poonam Kumari1, Mohit Saxena, (2016) "A Round-Robin based Load balancing approach for Scalable demands and maximized Resource availability", *International Journal of Engineering and Computer Science*, ISSN: 2319-7242. Volume 5, Page No. 17375-17380.

[10] Mustafa ElGili Mustafa, (2017) "Load Balancing Algorithms Round-Robin (RR), Least connection, And Least Loaded Efficiency*", GESJ: Computer Science and Telecommunications*, No.1(51).

[11] Shivangi Mayur, Nidhi Chaudhary, (2019) "Enhanced Weighted Round Robin Load Balancing Algorithm in Cloud Computing", *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*, ISSN: 2278-3075, Volume-8, Issue- 9S2.

[12] Nguyen Xuan Phi, Cao Trung Tin, Luu Nguyen Ky Thu and Tran Cong Hung, (2018), "Proposed Load Balancing Algorithm To Reduce Response Time And Processing Time On Cloud Computing", *International Journal of Computer Networks & Communications (IJCNC)* Vol.10, No.3.

[13] Mao Y., Chen X., Li X, (2014) "Max–Min Task Scheduling Algorithm for Load Balance in Cloud Computing", *In: Patnaik S., Li X. (eds) Proceedings of International Conference on Computer Science and Information Technology. Advances in Intelligent Systems and Computing*, vol 255. Springer, New Delhi.

[14] Bhavisha Kanani, Bhumi Maniyar, (2015) "Review on Max-Min Task Scheduling Algorithm for Cloud Computing", *JETIR*, (ISSN-2349-5162), Volume 2, Issue 3.

[15] Nguyen Xuan Phi and Tran Cong Hung, (2017) "Load Balancing Algorithm to improve response time on cloud computing", *International Journal on Cloud Computing: Services and Architecture (IJCCSA).* Vol.7, No.6.

[16] Rodrigo N. Calheiros, Rajiv Ranjan, Anton Beloglazov, C´ esar A. F. De Rose and Rajkumar Buyya, (2010) "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms", *Software: Practice and Experience (SPE)*, Volume 41 Number 1, pp.23-50.

[17] Tran Cong Hung, Phan Thanh Hy, Le Ngoc Hieu, Nguyen Xuan Phi, (2019) "Improved Max-Min Scheduling Algorithm for Load Balancing on Cloud Computing", *ICMLSC Proceedings of the 3rd International Conference on Machine Learning and Soft Computing*, pp.60-64.

## MSRTIA: MỘT ĐỀ XUẤT ĐỂ GIẢM THỜI GIAN ĐÁP ỨNG CHO CÂN BẰNG TẢI TRÊN ĐIỆN TOÁN ĐÁM MÂY

***Tóm tắt:*** Điện toán đám mây là mô hình cung cấp mọi thứ liên quan đến công nghệ thông tin dưới dạng dịch vụ thông qua Internet. Lợi ích chính của nó là tiết kiệm chi phí đầu tư hệ thống ban đầu, tối ưu hóa việc xử lý dữ liệu, tính toán và lưu trữ dữ liệu. Ngày nay, điện toán đám mây phải đối mặt với nhiều thách thức trong việc đảm bảo chất lượng dịch vụ xuyên suốt. Trong đó vấn đề quá tải máy chủ vật lý hoặc máy chủ ảo của các trung tâm dữ liệu được đặc biệt quan tâm. Vì vậy, để đủ điều kiện cho các yêu cầu trên, thiết lập một phương pháp cân bằng tải hiệu quả và sử dụng các tài nguyên tối ưu hóa nhất là mục tiêu mà điện toán đám mây muốn đạt được. Trong bài báo này, chúng tôi đề xuất thuật toán MSRTIA dựa trên thuật toán lập lịch Max Min. Thuật toán của chúng tôi tính toán giá trị tổng hợp Cloudlet của các yêu cầu sau đó ghép yêu cầu với giá trị lớn nhất được tìm thấy với máy ảo thực thi nhanh nhất. Trong đó, giá trị tổng hợp của Cloudlet là sự kết hợp của các tham số độ dài, kích thước đầu ra và kích thước tệp. Kết quả mô phỏng chứng minh rằng MSRTIA có thời gian phản hồi nhanh hơn so với các thuật toán Max Min và Round-Robin.

***Từ khóa:*** Giá trị tổng hợp Cloudlet, Max-Min, Round-Robin

## AUTHORS

**Tran Cong Hung** was born in Vietnam in 1961. He received the B.E in electronic and Telecommunication engineering with first class honors from HOCHIMINH University of technology in Vietnam, 1987. He received the B.E in informatics and computer engineering from HOCHIMINH University of technology in Vietnam, 1995. He received the Master of Engineering degree in telecommunications engineering course from postgraduate department Hanoi University of technology in Vietnam, 1998. He received PhD. at Hanoi University of technology in Vietnam, 2004. His main research areas are B – ISDN performance parameters and measuring methods, QoS in high speed networks, MPLS. He is, currently, Associate Professor PhD. of Faculty of Information Technology II, Posts and Telecoms Institute of Technology in HOCHIMINH, Vietnam.

**Nguyen Ngoc Thang** was born in Vietnam in 1993. He received B.E in Computer Science with first class honors from Industrial University of Ho Chi Minh City,

Vietnam, 2015. He received the Master of Computer Science degree with first class honors from Saigon University, Vietnam in 2018.

**Kieu Trong Duc** was born in Vietnam in 1989. He received his undergraduate degree in 2011, major in Information Technology from Saigon University, Viet Nam. Currently, he is a Master candidate in Computer Science of Saigon University, Vietnam. He is working for the Vietnam Mobile Telecom Services One Member Limited Liability Company.