PROPOSED LOAD BALANCING ALGORITHM TO REDUCE RESPONSE TIME AND PROCESSING TIME ON CLOUD COMPUTING

Nguyen Xuan Phi[1], Cao Trung Tin[2], Luu Nguyen Ky Thu[3] and Tran Cong Hung[4]

[1,2,3,4] Posts and Telecommunications Institute of Technology, Ho Chi Minh, Vietnam.

ABSTRACT

Cloud computing is a new technology that brings new challenges to all organizations around the world. Improving response time for user requests on cloud computing is a critical issue to combat bottlenecks. As for cloud computing, bandwidth to from cloud service providers is a bottleneck. With the rapid development of the scale and number of applications, this access is often threatened by overload. Therefore, this paper our proposed Throttled Modified Algorithm(TMA) for improving the response time of VMs on cloud computing to improve performance for end-user. We have simulated the proposed algorithm with the CloudAnalyts simulation tool and this algorithm has improved response times and processing time of the cloud data center.

KEYWORDS

Load balancing; response time; cloud computing; processing time.

1. INTRODUCTION

Cloud computing, also known as virtual server computing, is a computer model that uses computer technology and developing based on the Internet. The term "cloud" here refers to the Internet (based on its layout in the computer network topology) and complexity level of the infrastructure contained within it. In this computing model, all possibilities related to information technology are provided in the form of "services", which allow users to access technology services from a certain provider "in the cloud "without the knowledge and experience of that technology, nor should it consider the infrastructure that serves that technology. Now with the explosive growth of the Internet, the exchange of data of organizations and businesses is a timely issue. Cloud computing allows applications to be less dependent on network infrastructure, saving users money when not investing more in hardware. All data will be uploaded to the cloud, users will only be able to access and use it anywhere, anytime. On that basis, cloud computing technology has emerged and increasingly developed, the problem of exchange, processing, data security and especially load balancing in cloud computing is Challenges are set for researchers as well as cloud service providers.

Because of the apparent efficiencies it brings, cloud computing has become a new technology trend, a solution in which all computing resources (hardware, software, networking, storage, etc) are provided promptly to the user as they request. But it is because of the explosion of data exchange that this technology is posing challenges for developers, experts and researchers around the world, especially the load balancing on cloud data center. Because load balancing on cloud data centers is about improving the quality of service, optimizing computing resources, or otherwise improving the efficiency of the cloud system of the Cloud Service Provider. A few years ago, the amount of data transmitted on a global network if stored on DVDs, the number of disks

DOI : 10.5121/ijcnc.2018.10307

lined up would be two times as long as the distance to the moon. It is estimated that this amount of data will increase by 44 times by 2020. The growth of data traffic with more than 5 billion people using mobile devices is one of the key factors driving growth up of cloud computing. So, to address this threat, we need to incorporate several different approaches to loading balancing for cloud data centers. One of the methods is to reduce the response time of cloud services when users request access to services. Load balancing aims to find strategies to save computing resources and increase user service, which directly affects the service provider's business to make a profit.
The paper is organized as follows: Part 1 Introduction: Overview of the need for load balancing in the cloud. Part 2 Related works: Survey, evaluate some recently published work on load balancing. Part 3 Proposed load balancing model. Part 4 Simulation and results of the proposed algorithm. Part 5 Conclusion.

2. RELATED WORK

Load balancing on cloud computing has attracted many researchers around the world and has also gained important achievements [1-6],[8-11]. Load balancing in communication is a very important factor in improving the performance of cloud data centers.

Syed Hamid Hussain Madni [1] has studied and evaluated the resource allocation techniques in the cloud environment. The article studied and pointed out the parameters to improve the performance of the cloud system. This article also outlines the importance of allocating resources in the cloud, requiring resource allocation policies, strategies, and algorithms to distribute and migrate resources to best support both suppliers and users.

Shubham Sidana et al.has present NBST algorithm [2] for load balance on cloud based on the resource sort in the speed of VMs and will allocation of resources the request to the users. In this algorithm, authored to try to be load balancing by sort speed of VMs and sort the length of the cloud. The VMs list and Cloudlet list is sent to Broker for the allocation. The list of VMs and cloudlets is then sent to the broker for allocation. Broker allocates through the middle point algorithm, this algorithm divides the VM list and cloudlet list until it have maximum of one cloudlet or one virtual machine in the list and then allocate the resource to be executed. This algorithm allocates resources in a way that requires less processing work than allocated to high-capacity machines and vice versa. The limitation of this algorithm is that there is no mechanism for moving requests for virtual machines directly (live migration).

Feilong Tang et al. [3] and his colleagues proposed an algorithm is DLBS-dynamical load-balanced scheduling for the cloud environment proposed by. The author proposes a new method of dynamic load balancing (DLBS) to maximize throughput. Based on the development of a set of heuristic scheduling algorithms, the DLBS algorithm is effective for the OpenFlow network model, in which the data stream is balanced through time slots. Initially experimental results showed that the hypothesis was more efficient than the Round Robin and Lobus algorithms [3]. The algorithm provides the parameter $\delta(t)$ - the parameter unbalance load, through updating this parameter, the algorithm can adapt to different network conditions at all times.

Sambit Kumar Mishra [4] present a novel load balancing approach to organizing the virtualized resources of the data center efficiently. In this approach, the load to a VM scales up and down according to the resource capacity of the VM. The proposed scheme minimizes the makespan of the system, maximizes resource utilization and reduces the overall energy consumption. The proposed approach balances the load at VM-level to avoid overloading VM node. A task with a high priority gets service first to maximize the profit of Cloud Services Provider. The proposed algorithm has compared with the FCFS and RR algorithm and simulation in CloudSim environment. The result is reduces the waiting time and optimized the makespan of the cloud data center.

In this paper [5], Sobhan Omranian-Khorasani et al. presented a heuristic algorithm for scheduling deadline- constrained workflows is DCLB (Deadline Constrained Level Based). The algorithm used Level Load Balancing to refine deadline distribution as well as attaining lower communication cost in order to reach the algorithm's goals. Experimental results (based on Amazon EC2) demonstrate that DCLB compared to existing algorithms achieves higher cost efficiencies when workflow deadline is met.

Atyaf Dhari et al. [6] have proposed the LDAB scheduling algorithm to achieve load balancing and QoS. Load balancing becomes an important point to make and stabilize the system. Therefore, it is essential to improve the performance of the system by balancing the workload between virtual machines. Method: The proposed load balancing algorithm (LBDA) is to manage and balance the load between virtual machines in a data center along with reducing the completion time (Makespan) and response time. The LBDA's operational mechanism is based on three phases: first, calculate virtual machine capacity and load on the VM to classify virtual machine states (under load, load balancing, overload). Second, calculate the time needed to perform the task on each virtual machine. Finally, make the decision to distribute tasks between virtual machines based on virtual machine state and task timing. The algorithm was compared to MaxMin, Shortest Job First and Round Robin. The results of LBDA is more effective than these algorithms.

Mark van der Boor et al. [8] introduce two enhancements of the ordinary JIQ scheme where tokens are either distributed non-uniformly or occasionally exchanged among the various dispatchers. Join the-Idle-Queue (JIQ) algorithms, which rely on tokens issued by idle servers in dispatching tasks. Specifically, JIQ strategies involve minimal information exchange, and yet achieve zero blocking and wait in the many-server limit. Therefore, the author used product-form representations and fluid limits to show that the basic JIQ scheme fails to deliver zero blocking and wait for any asymmetric dispatcher loads, even for an arbitrarily low overall load. Remarkably, it is the least-loaded dispatcher that throttles tokens and leaves idle servers stranded, thus acting as bottleneck. The enhancements of JIQ has increase in large-scale systems.

To improve the availability and continuity of cloud computing [9],[11], the authors introduce a load balancing approach that reduces the response time and the cloud latency. By studying how to copy data from the source VM to the target VM where the source VM is faulty, the goal is to have users access information continuously. In paper [9], the idea is to combine the Weighted Round Robin and Max Min algorithms to form an efficient load balancing algorithm Weighted MaxMin and this algorithm has reduced two important parameters: waiting time and response times.
Mohammad Riyaz Belgaum et al. [10], the various task scheduling algorithms are studied to present the dynamic allocation of resources under each category and the ways each of this scheduling algorithm adapts to handle the load and have high-performance computing. The task scheduling is done by the cloud service provider using preemption and non-preemption based on the requirements in a virtualized scenario which has been focused here. The results of simulation show that execution load policy is better for the data centers to use in different regions.

3. PROPOSED ALGORITHM

To improve the response time for the user (UserBase) and processing time of data center. Our proposed Throttled Modified Algorithm (TMA) by effective reallocation the tasks, it had deployment at the VmLoadBalancer in Datacenter Controller, it was improved based on Throttled Algorithm.

3.1. Theoretical basis

3.1.1. Roud Robin algorithm

The Round Robin algorithm is one of the simplest algorithms based on quantum theory. Round Robin tries to distribute the load to the VMs in the order of fair rotation. The idea of Round Robin is that all VMs in the data center receive the same load in circular order without regard to their processing power when task allocation. This is effective for data centers that have all VMs that have the same processing power. As for data centers there are large VMs capable of power processing large disparities, which are ineffective.

3.1.2. Throttled Algorithm

The sequence of steps:

Step 1. Throttled Load Balancer execution load balancing by update and maintain an index table contains the status information (available '0' or not available '1') of all VMs. At start, all VM at the status is available '0'.
Step 2. Data Center Controller received a new request.
Step 3: Data Center Controller query to Throttled Load Balancer for the new task.
Step 4: Throttled Load Balancer will be checked VM on the top table, determined the first VM is available.

If found VM:

- Throttled LoadBalancer sends the ID of VM to Data Center Controller.
- The Data Center Controller sends a request to the VM specified by that ID.
- Data Center Controller notifies the Throttled LoadBalancer a new allocation.
- Throttled LoadBalancer updates the index and waits for new requests from the Data Center Controller.
-

If not found VM:

- Throttled LoadBalancer will return a value of -1 to the Data Center Controller.
- The Data Center Controller arranges the request.

Step 5: As for the VM, after processing the request and the Data Center Controller receives a response, it will notify to Throttled LoadBalancer is stopped.
Step 6: If there are multiple requests, the Data Center Controller repeats Step 3 with the next index and the process is repeated until the index table size is empty.
This algorithm optimizes the response time than the Round Robin algorithm. But the limitation is to detect the VM is ready '0' with the index table size out.

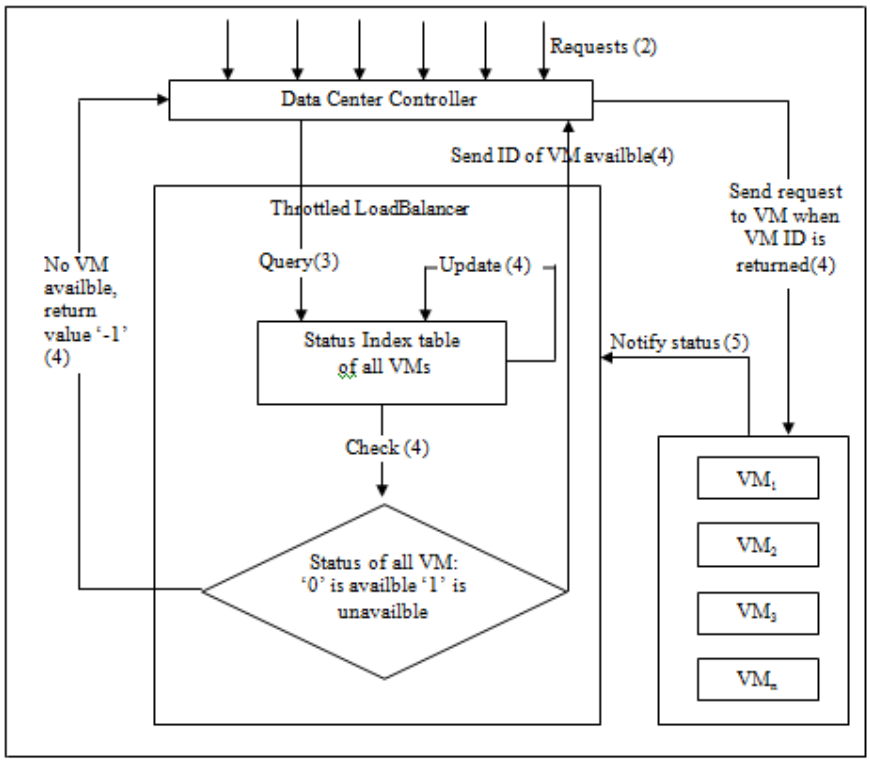

Figure 1: Throttled algorithm operation diagram

3.2. Proposed algorithm – Throttled Modified Algorithm (TMA)

The sequence of steps:

Step 1. The TMA Load Balancer performs load balancing by updating, maintaining two index tables.

- Available Index: Status of VMs is available is '0'
- Busy Index: Status of VMs is not available '1'.

At the beginning, all VMs are updated in the "Available Index" table and the "Busy Index" table is empty.

Step 2. The Data Center Controller receives a new request.

Step 3: Data Center Controller queries to the TMA Load Balancer for next allocations.

Step 4: TMA Load Balancer detects and sends VM ID (VM) from the top down in the "Available Index" table of the Data Center Controller.

- The Data Center Controller sends the request to the specified VM by that ID.
- The Data Center Controller informs the TMA Load Balancer for a new allocation.
- The TMA LoadBalancer will update the this VM into the Busy Index and wait for the new request from Data Center Controller.

In case, if the table "Available Index" is empty (all VMs unavailable):

- TMA Load Balancer will return a value of -1 to the Data Center Controller.
- The Data Center Controller arranges the request.

Step 5: As for the VMs, after processing the request, and the Data Center Controller receives the response from VM, it will notify to the TMA LoadBalancer tthen update the "Available Index" table.

Step 6: If there are multiple requests, the Data Center Controller repeats Step 3 and the process is repeated until the "Available Index" table is empty.

With the our proposed algorithm (TMA), it will be possible to detect the VM availble (status '0') with the size of the table "Available Index" more flexible than the Throttled Algorithm. This improves the performance of the system.

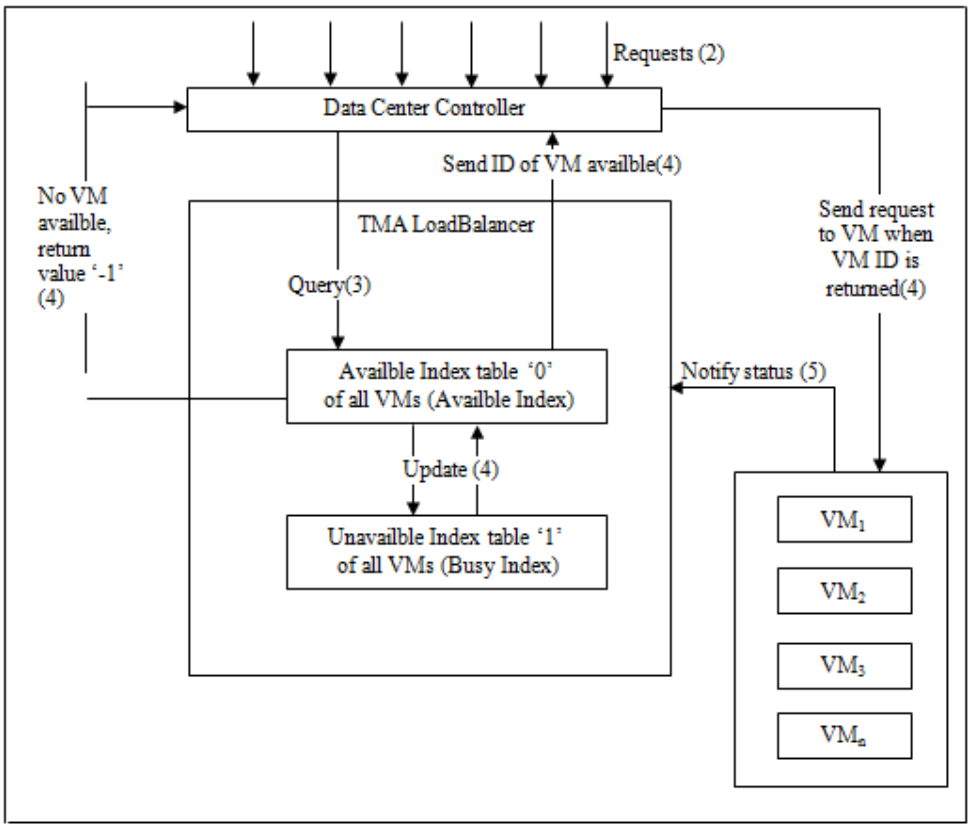

Figure 2: TMA operation diagram

4. SIMULATION AND EVALUATION

In this paper, we used the Cloud Analyst simulation toolkit to simulation and evaluate the proposed algorithm with two algorithms: Round-Robin and Throttled. We consider the parameters such as the overall response time of the cloud system, the processing time of the data center

4.1. Cloud Analyst Simulation Kit

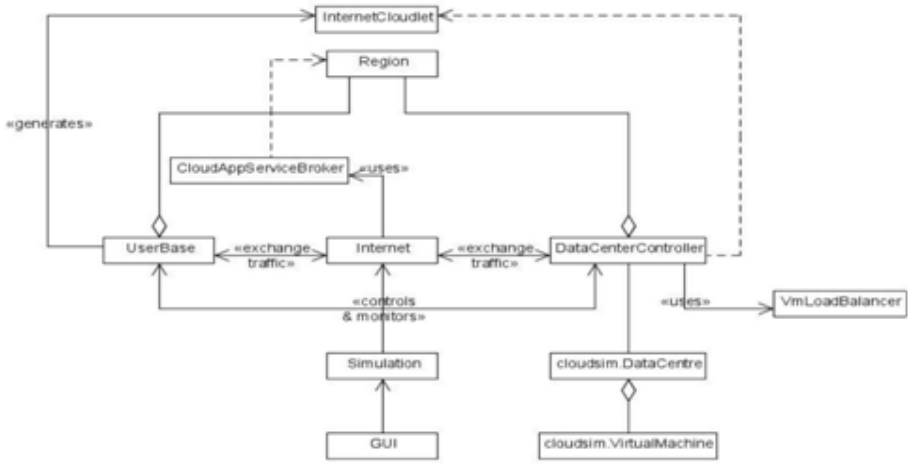

Figure 3. CloudAnalyst main components [7]

The implementation process of the Cloud Analyst toolkit:

- UserBase will create Internet Cloudlets and transmit them over the Internet. Internet Cloudlets are assigned additional Internet Characteristics such as Latency, Transmission Delay to Data Center, assigned through the Cloud App Service Broker distribution policy.

- Here, the Data Center Controller will decide which VMs will receive or process the Internet Cloudlet through the load balancing policies of the Vm Load Balancer. And processing the Internet Cloudlet and returning results is done under the CloudSim background.

- After receiving the results from the VM returned, the Data Center Controller will send back to the UserBase via the Internet and updated with Service Latency parameters from Internet Characteristics.

- UserBase when receiving the result returned it will update the response time.
This is repeated until the simulation is finished and the simulation results reported. Figure 3 shows the details of the simulation process.

4.2. Simulation setup

We simulated 6 UserBbase that corresponds to six zones with a specific time zone, and most users use the app in the evening for about 2 hours after work. That every 5 minutes, each user sends a new request while online:

Table 1: UserBase configuration parameters.

User Base	Region	Time Zone	Peak Hour	Simulataneous Online Users During Peak Hrs	Simulataneous Online Users During Off-Peak Hrs
UB1	0	GMT - 6.00	13:00-15:00	400,000	40,000
UB2	1	GMT – 4.00	15:00 – 17:00	100,000	10,000
UB3	2	GMT + 1.00	20:00 – 22:00	300,000	30,000
UB4	3	GMT + 6.00	01:00 – 03:00	150,000	15,000
UB5	4	GMT + 2.00	21:00 – 23:00	50,000	5,000
UB6	5	GMT +10.00	09:00 – 11:00	80,000	8,000

Where:

- *Peak Hour*: peak time period of access
- *Simulataneous Online Users During Peak Hrs*: number of users accessing during peak times period.
- *Simulataneous Online Users During Off-Peak Hrs*: number of users accessing during low time.

These parameters are configured in the Main Configuration tab of the Configure Simulation class. Also in this tab is the configuration of the virtual machine (VM) (Figure 4).

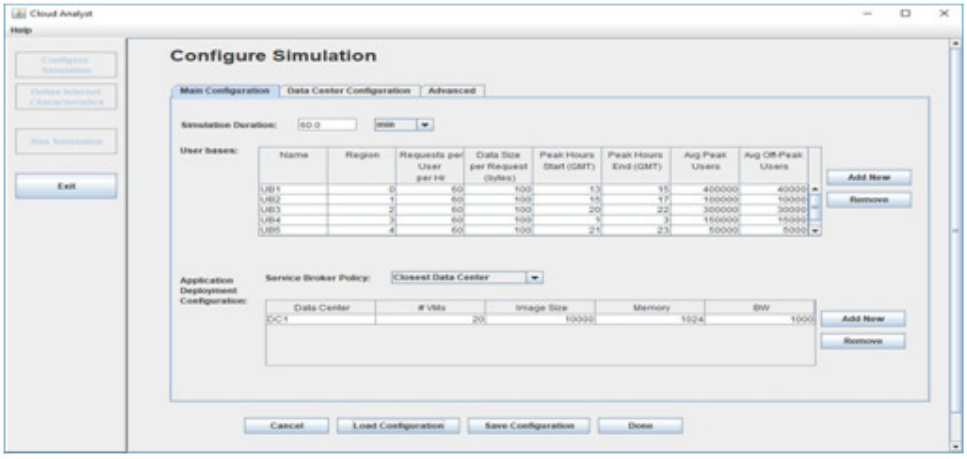

Figure 4: User and VMs configuration settings

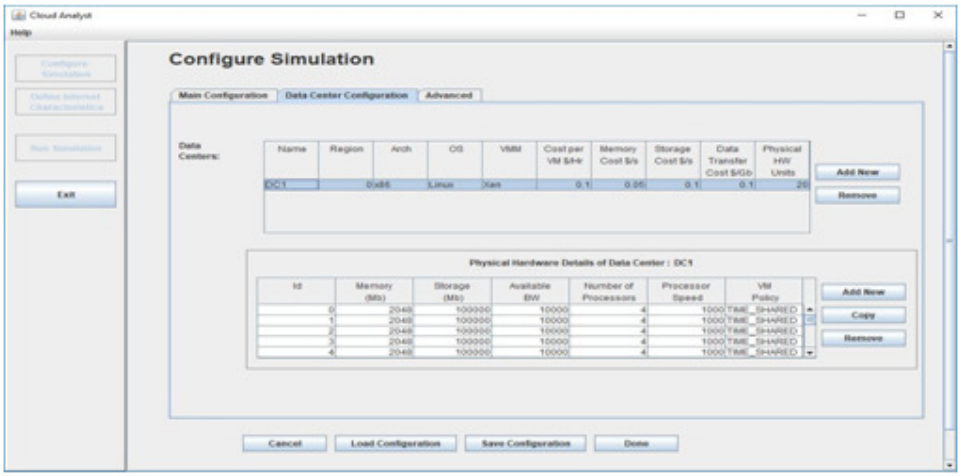

Figure 5: Data center configuration parameters

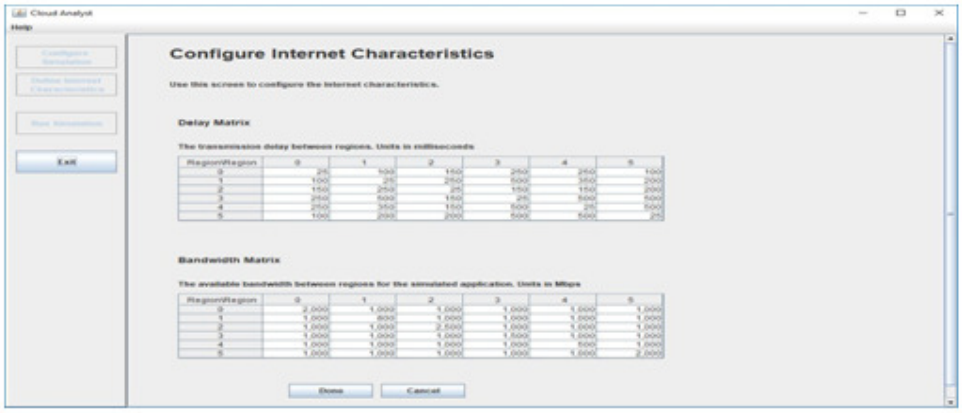

Figure 6: Internet feature configuration.

Here, simulate 3 times corresponding to 3 different policies. Specifically:

- 1st time: apply Round Robin policy (this policy is available in the simulator).
- 2nd time: apply Throttled policy
- 3rd time: apply the policy with our proposed TMA algorithm.

4.3 Simulation and Analysis Results

Scenario 1: Simulate with 20 virtual machines (VMs).

For the Round Robin algorithm, the requests are distributed evenly over the VMs so there is no need for queues to be distributed. As for the Throttled algorithm, the detection of virtual machines in the state index table by the detection method from the beginning of the table to the end of the table will lead to the status of requests to queue when the system has the number of virtual machines (VM) large. With the TMA, it used of two status index table (Available Index and Busy Index), the system only needs to distribute requests to VMs in the Available index table without having to search for them. This eliminates the need to queue up the system, improving the processing time of the data center.

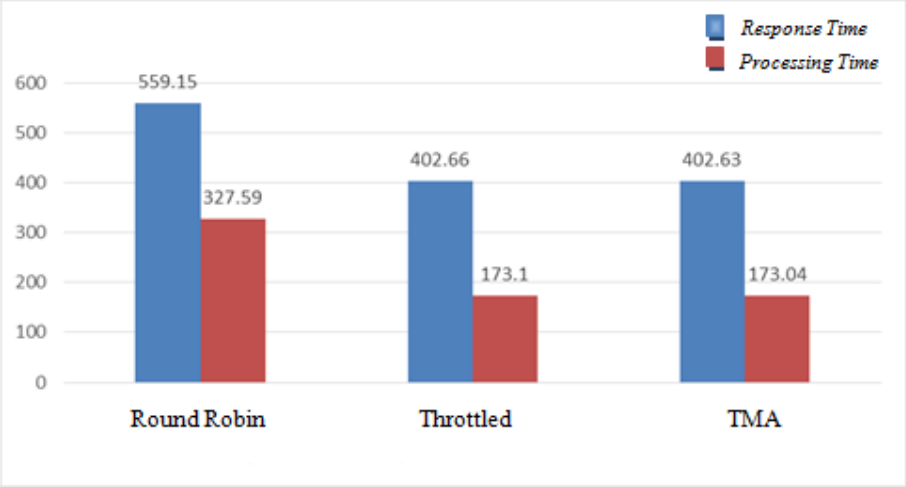

Figure 7: Simulation results with 20 VMs.

In Figure 7, show that for the Round Robin algorithm, the required distribution to the VM rotates in a circle without considering the state of the VM, resulting in the Data Center's processing time and response time. The system to UserBase user base is much higher than the other two algorithms. For the other two algorithms, our TMA algorithm has a Data Center processing time and the system response time is lower than the Throttled algorithm, albeit very little. Therefore, our have tried increasing the number of VMs to 50 machines with the same parameters as above for comparison again.

Scenario 2: Simulation with 50 virtual machines (VMs)

From Figure 8, show that the data center's processing time and the average response time of the TMA algorithm are much lower than the Throttled algorithm when the number of VMs increases.

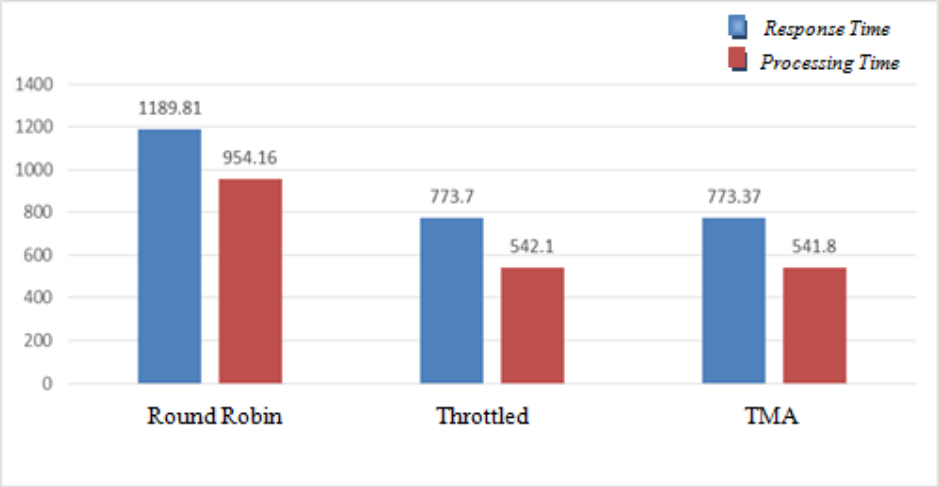

Figure 8: Simulation results with 50 VMs.

From experimental results simulated in the above two cases. It helps us to see that with the TMA algorithm, the number of requests that have to queue has decreased, as well as the processing time of the data center and the response time of the system is improved than the two algorithms. This means that the TMA algorithm has better load balancing than the Throttled and Round Robin algorithms.

5. CONCLUSION

This paper focuses on the popular load balancing algorithms in today's cloud environment, analyzing and proposing an improved algorithm (TMA) based on an algorithm already in place to improve Improved load balancing over older algorithms, and has accomplished the following goals. The results obtained from the proposed algorithm have met these goals, such as limiting the number of requests queued for delivery, improving processing time and response time of hubs cloud compared to two old algorithms. This also means that with the proposed algorithm, the performance of cloud computing is improved compared to the two algorithms Round Robin and Throttled. Our proposed algorithm has shown efficiencies when the number of VMs increases: reducing the response time and processing time of cloud data centers. In the future, we will study improvements to optimize the performance of the algorithm.

REFERENCES

[1] Syed Hamid Hussain Madni (2016), "Recent advancements in resource allocation techniques for cloud computing environment: a systematic review", Springer Science+Business Media New York 2016, DOI 10.1007/s10586-016-0684-4,.

[2] Shubham Sidana, Neha Tiwari, Anurag Gupta Inall Singh Kushwaha (2016), "NBST Algorithm: A load balancing algorithm in cloud computing", International Conference on Computing, Communication and Automation (ICCCA2016), DOI: 10.1109/CCAA.2016.7813914, 29-30 April, Noida, India.

[3] Feilong Tang, Laurence T. Yang, Can Tang, Jie Li and Minyi Guo (2016), "A Dynamical and Load-Balanced Flow Scheduling Approach for Big Data Centers in Clouds", DOI 10.1109/TCC.2016.2543722, IEEE TRANSACTIONS ON CLOUD COMPUTING.

[4] Sambit Kumar Mishra, Md Akram Khan, Bibhudatta Sahoo, Deepak Puthal, Mohammad S. Obaidat, and KF Hsiao (2017), "Time efficient dynamic threshold-based load balancing technique for Cloud Computing" IEEE International Conference on Computer, Information and Telecommunication Systems (CITS), 21-23 July, Dalian, China.

[5] Sobhan Omranian-Khorasani and Mahmoud Naghibzadeh (2017), "Deadline Constrained Load Balancing Level Based Workflow Scheduling for Cost Optimization", 2017 2nd IEEE International Conference on Computational Intelligence and Applications, Beijing, China 8-11 Sept.

[6] Atyaf Dhari, Khaldun I. Arif (2017), "An Efcient Load Balancing Scheme for Cloud Computing", Indian Journal of Science and Technology, Vol 10(11), DOI: 10.17485/ijst/2017/v10i11/110107.

[7] Bhathiya Wickremasingle (2010), "Cloud Analyst: A CloudSim- based Tool for Modelling and Analysis of Large Scale Cloud Computing Environments", MEDC Project Report, in 433-659 Distributed Computing Project, CSSE Dept, University of Melbourne.

[8] Mark van der Boor1, Sem Borst1,2, and Johan van Leeuwaarden (2017), "Load Balancing in Large-Scale Systems with Multiple Dispatchers", IEEE INFOCOM 2017 - IEEE Conference on Computer Communications, 1-4 May, Atlanta, GA, USA.

[9] Bharat Khatavkar, Prabadevi Boopathy (2017), "Efficient WMaxMin Static Algorithm For Load Balancing In Cloud Computation", DOI: 10.1109/IPACT.2017.8245166, International Conference on Innovations in Power and Advanced Computing Technologies [i-PACT2017], Vellore, India.

[10] Guo Xiao, Wu Wenjun, Zhao Jiaming, Fang Chao and Zhang Yanhua (2017), " An OpenFlow Based Dynamic Traffic Scheduling Strategy for Load Balancing," 2017 3rd IEEE International Conference on Computer and Communications (ICCC),DOI: 10.1109/CompComm.2017.8322602, 13-16 Dec, Chengdu, China.

[11] Jananta Permata Putra, Supeno Mardi Susiki Nugroho, Istas Pratomo (2017), "Live Migration Based on Cloud Computing to Increase Load Balancing", 2017 International Seminar on Intelligent Technology and Its Application, 10.1109/ISITIA.2017.8124096, Publisher: IEEE, 28-29 Aug. Surabaya, Indonesia.

AUTHORS

Nguyen Xuan Phi was born in Vietnam in 1980. He received Master in Posts & Telecommunications Institute of Technology in Ho Chi Minh, Vietnam, 2012, major in Networking and Data Transmission. Currently, he is a PhD candidate in Information System from Post & Telecommunications Institute of Technology, Vietnam. He is working at the Information Technology Center of AGRIBANK in Ho Chi Minh city, Vietnam. His main research areas are load balancing on cloud computing, optimizing the performance of cloud computing.

Cao Trung Tin was born in Vietnam in 1991. He received the B.E in Information Technology from Ho Chi Minh University of Technology (HUTECH) in Vietnam, 2013. He is currently MSc. Candidate in Information System from Post & Telecommunication Institute of Technology, Vietnam in 2018. He is working at the Ho Chi Minh City Power Corporation – Information Technology Company, Vietnam.

Luu Nguyen Ky Thu was born in Vietnam in 1968 He received Master in University of Science in Ho Chi Minh, Vietnam, 2002, major in Information System. He is working at the Posts and Telecoms Institute of Technology in HOCHIMINH, Vietnam.

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 Ph.D 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.