International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 10 Issue: 12 | Dec 2023 www.irjet.net p-ISSN: 2395-0072

An Optimized-Throttled Algorithm for Distributing Load in Cloud Computing

Tanjina Jahan[1], Prof. Dr. Mohammod Abul Kashem [2], Md. Toufecul Islam [3]

[1]*Tanjina Jahan: Student, Dept. of Computer Science and Engineering, Dhaka University of Engineering and Technology, Bangladesh*
[2]*Dr. Mohammod Abul Kashem: Professor, Dept. of Computer Science and Engineering, Dhaka University of Engineering and Technology, Bangladesh*
[3]*Md.Toufecul Islam: Student, Dept. of Computer Science and Engineering, Dhaka University of Engineering and Technology, Bangladesh*

Abstract - *Cloud computing is a new area with many promising applications for both established businesses and individual users. This is a development of the idea of decentralized computing. Based on the concept of "on-demand" services, it delivers data, applications, and infrastructure in response to users' immediate requests [1]. Load balancing, in which the load is distributed among many cloud servers or nodes, improves efficiency. It's the single most crucial element in maximizing the use of available resources. Load balancing has emerged as a critical process in cloud computing infrastructures [2]. To meet the needs of such a large user base, a distributed solution is necessary, since centrally managing one or more idle services is neither practical nor cost-effective. It's impossible to give certain users control over individual computers. The "cloud" in "cloud computing" refers to the large network of distributed nodes [3]. As a result, it needs load balancing to distribute the workload across its many servers or virtual machines. In order to improve cloud performance, the research suggested an algorithm that prioritizes load distribution to minimize virtual machine overload and underload. Cloud Analyst is used to do in-depth research and comparisons. We compared our results to those of the more seasoned Round Robin and Throttled algorithms. In addition, simulation findings show that the proposed algorithm has improved response times and processing times in the cloud data center, demonstrating its superiority over current methods.*

Key Words: Cloud Computing, Load Balancing, Processing Time, Response Time.

1. INTRODUCTION

Corporations and educational institutions use distributed computing's new cloud computing technology to store and access data. The key difficulty is planning for incoming queries in a way that minimizes response time while optimizing resource use. In order to process client requests with the quickest feasible response time and allocate them to virtual machines, a number of algorithms are utilized, including FCFS, Round Robin, and Throttled. Many resources are not involved in the execution of requests, and the cloud system is unbalanced because of restrictions such as excessive communication delays and underutilization of resources [4].

Load balancing is essential for maximizing throughput and reaction time in a cloud environment because each virtual machine performs the same function. To evenly distribute work between computers, we may dynamically move tasks from one machine to another. This enhances the system's performance ratio, optimizes user satisfaction, decreases response time, maximizes resource usage, and decreases task rejections. Virtualization technology can effectively manage the dynamic resources of a cloud-based platform. By allowing several virtual machines to share a single physical server, it introduces a novel approach to enhancing the power efficiency of datacenter consolidation. This means that the cloud computing system's energy usage may be lowered by turning off or putting some of the servers into sleep mode [5].

In this research, we provide a new approach for distributing incoming jobs across available virtual machines. Here, new requests are sent to the least-loaded VM once the least-used VM has completed its current workload. This technique is compared to the more traditional round-robin and throttled approaches to reducing virtual machine underutilization. The proposed algorithm focuses on efficiently dividing up incoming work across available virtual machines. When there are none in the VM index, it looks for the least busy ones and adds them.

2. ROUND ROBIN ALGORITHM

One of the simplest static load-balancing algorithms is the round-robin algorithm, which simply rotates amongst workers in a loop until all requests have been processed. The request is then sent to a pool of randomly chosen virtual machines by the data center controller. All servers and nodes are grouped together into clusters according to how quickly they handle data. When a virtual machine (VM) is selected for a task, it moves to the front of the queue. However, as long as the virtual machine isn't empty, new incoming workloads must wait in line, which is the core problem with this allocation. Because of this issue, resource management would suffer, leading to longer response times and decreased productivity. The Round Robin algorithm prevents feeling full almost entirely while having low throughput [6].

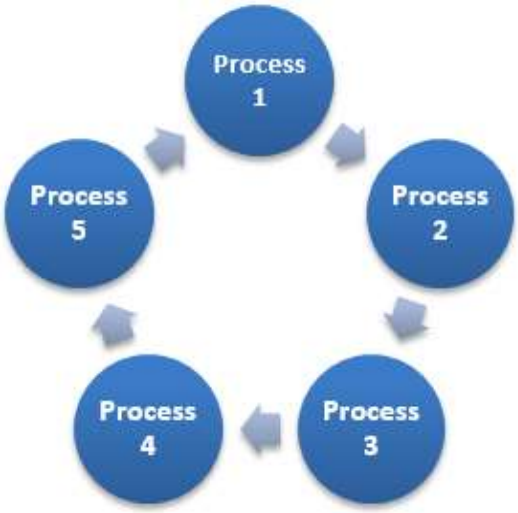

Fig -1: Round Robin Algorithm

3. THROTTLED LOAD BALANCING ALGORITHM (TLB)

In this approach, the load balancer is in charge of updating the index table. The virtual machine's availability (busy or not) is shown. When a job comes in, a load balancer sends it to the right virtual machine. Which may be used to fulfil the request made by the user. However, when throttled checks for available virtual machines, it starts at index in the database [6].

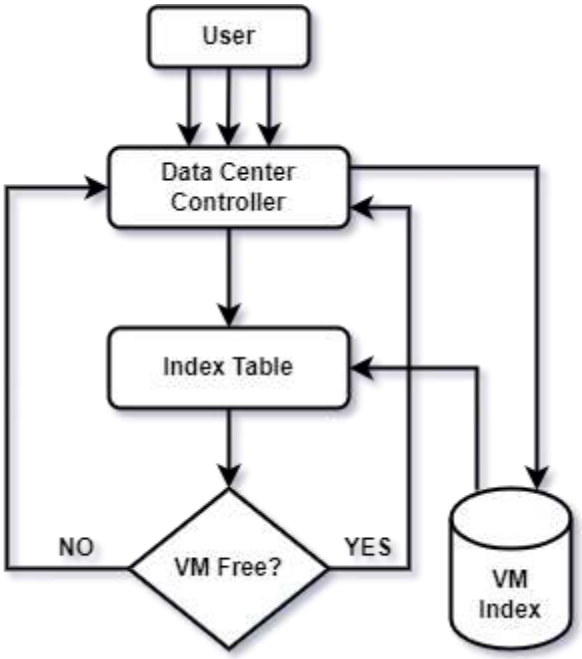

Fig -2: Throttled Algorithm

4. OPTIMIZED THROTTLED ALGORITHM (OTA)

Step 1: In the first phase of the load balancing procedure, the load balancer maintains and refreshes an index table that contains the availability (free, "0") or occupation (occupied, "1") of all VMs. Virtual machine IDs start at '0'.
Step 2: Data Center Controller received a request and then requested the new assignment from the Load Balancer.
Step 3: The load balancer checks the index table and determines that the first available VM is free to be used.

The following actions are taken if a VM is located:

- The VM's ID is sent to the VM index.
- The data center controller receives the VM ID from the VM index.
- When a fresh allocation becomes available, the data center controller notifies the load balancer.
- After receiving requests from the data center controller, the load balancer updates the index and waits for them to complete.

If a virtual machine (VM) cannot be located:

- Identifies the least busy virtual machines and adds them to the VM index.
- The ID of the least-loaded virtual machine is sent to the data center's control system through the formula. $VM_{loadi} <= C_{VMi} * TLL$

  Where $C_{VM}=Pe_{num}*Pe_{mips}$ and $VM_{loadi}=\frac{\sum_{j=1}^{n} TL}{n}$

- VM Index transmits the VM ID to the Data Center Controller.

Step 4: The controller goes back to Step 2 and continues working there.

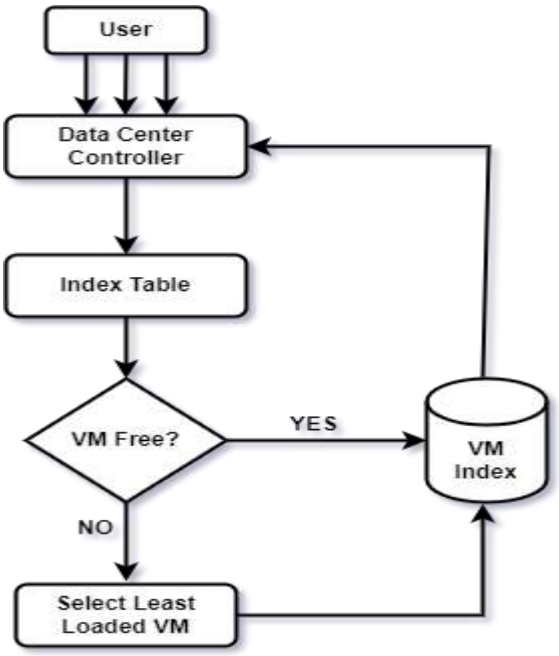

Fig -3: Optimized Throttled Algorithm

5. RELATED WORK

The following section offers a high-level summary of the load balancing approaches made use of in cloud computing. The main goal is to distribute all incoming tasks to available virtual machines so they may be processed immediately.

To ensure maximum utilization of each server-created virtual machine on cloud platforms, Garg *et al.* [7] introduced the Synchronized Throttled VM Load Balancing (STVMLB) Algorithm, which focuses on load balancing to reduce the possibility of overload and underload on virtual machines. This is a more effective approach to synchronizing all virtual computers. By adapting the basic idea behind Throttled, a load balancing algorithm, the system was created, and it increases virtual machine utilization even higher than Throttled and Active Monitoring. The suggested dynamic load-balancing technique performs well in a cloud setting, evenly distributing requests and maximizing response time. The Throttled algorithm, on the other hand, reduces the speed of response.

In this research, N. Xuan Phi *et al.* [8] evaluate an existing method and propose an enhanced version of it to better balance loads than previous algorithms. By recommending this strategy, we were able to accomplish our goals of decreasing the backlog of inquiries waiting to be sent and speeding up the processing and response time of cloud data centres in contrast to two previous methods. It also shows that the suggested method is more efficient than Round Robin and Throttled when it comes to cloud computing. The goal of this study is to improve end-user performance in cloud computing by proposing a throttled modified algorithm (TMA) for increasing the response time of virtual machines.

To avoid either overburdening or underutilizing virtual machines, G. Soni *et al.* [9] offer a Central Load Balancer (CLB) technique. Using criteria such as priority and status, CLB divides work among many virtual computers. Author simulations showed that CLB-based load balancing algorithms performed better than Round Robin (RR) and Throttled algorithms. Although doing so would provide for a steadier and more dynamic load distribution, the suggested method does not factor in the present utilization of resources like CPU and memory.

In this study, A.A. Alkhatib *et al.* [10] present a thorough examination of load balancing strategies that use a wide variety of VMs to accomplish the task. Several static and dynamic load balancing techniques, along with their benefits and drawbacks, are presented in this work. Furthermore, this study offers a comparison of several load-balancing techniques. There is a unique identification and a summary of the major qualities and limits of each load balancing method. The purpose of this study is to investigate the notion of load balancing in cloud computing as well as the advantages and disadvantages of a specific load balancing method. Algorithms including "honeybee foraging," "throttled," "Min-Min," "weight," "Max-Min," and "round robin" are among those studied, as are the metrics and issues related to them.

With the aid of a distinctive VM-assign load balancing technique, S. G. Domanal *et al.*'s [11] study evenly distributes new requests among all available virtual machines. In this situation, the virtual machine is allocated depending on how it will be used. The first request is given to the virtual machine that has received the fewest requests. This technique is then compared to the currently used Active-VM algorithm, which has been shown to significantly increase virtual machine underutilization. The results are analyzed to prevent under- and over-loading of virtual devices and guarantee their full usage. We also compare the results to the current Active-VM load balancing technique and assess its effectiveness using the CloudSim simulator. However, it cannot respond in real time to changes in user input.

The LDAB scheduling method, used for QoS and load balancing, was created by Atyaf Dhari *et al.* [12]. Load balancing is essential for keeping the whole system stable. The system's efficiency may be improved, therefore, by distributing the task across several virtual machines. The goal of the suggested load balancing algorithm (LBDA) is to decrease processing and response times while maintaining workload parity across a data centre's virtual machines. There are three phases to the LBDA's operation: First, determine the VM's capabilities and workload to determine the VM's current stage. Figure out how long it will take to do the task using all of the virtual machines. Depending on the availability of virtual machines and the timing of tasks, decide how to distribute the workload among them. Three other algorithms—MaxMin, Shortest Job First, and Round Robin were used to evaluate this one. In comparison to these methods, LBDA produces superior results.

Cloud computing and load balancing are two of the cloud computing resource allocation approaches identified in the current research by S. H. Sabeti *et al.* [13]. The author places an emphasis on load balancing and tries to ensure that all servers have about the same amount of work to do. To speed up responses and processes, this research suggests using a load-balancing algorithm that combines elements of the ESCE and Throttled algorithms. To reduce the time spent checking for a suitable virtual machine that can handle longer tasks and improve response time, the algorithm first proposes the least busy machine. Two more virtual algorithms, Throttled and ESCE, are combined into a single hybrid algorithm that is proposed. All four algorithms were simulated using the same framework, and the results showed that the suggested method completed tasks more quickly and had a lower total number of iterations than the other three. Additional goals, such as reduced costs and enhanced performance, have not yet been attained due to scheduling and technical constraints.

The Cloud Analyst Simulator was used to evaluate the Modified Throttled Load Balancing Algorithm, the FCFS Algorithm, and the Particle Swarm Optimization Algorithm by P. A. Pattanaik *et al.* [14]. According to the findings, Particle Swam is the optimalization method that yields the quickest response time compared to the other two. Moreover, Particle Swam optimization has lower total server costs than the other two techniques. Since costs play a major role in the cloud, minimizing them should be a key concern in terms of both efficiency and customer happiness. Using the Particle Swam Optimization Algorithm, we were able to find a better distribution map that represents the ideal option for our resources. The simulation outcomes are recorded in terms of response time, datacenter processing time, efficiency, and arrival costs for all three methods.

The Modified Throttled method, introduced by S. G. Domanal *et al.* [15], is similar to the Throttled algorithm in that it maintains an index table of VMs and VM states. Response times have been improved, and free virtual machines have been used to their full potential. The proposed technique is a way to choose a VM to handle a client's request, with the VM at the first number being picked mostly based on its state. If the virtual machine is available, its ID is transmitted to the data center with the query; otherwise, -1 is returned. In contrast to the Throttled algorithm, which parses the index table about the first index each time the data centre asks the load balancer to assign VMs, the next VM at the index next to the currently allocated VM is chosen based on its status when the next query comes in.

In this work, S. Y. Mohamed *et al.* [16] present the Balanced Throttled Load Balancing (BTLB) method. Results from other load balancing algorithms, including round robin and AMLB, as well as the throttled load balancing algorithm, are compared with those from BTLB. All four of these algorithms' efficacy will be shown in this analysis. The proposed technique is shown to decrease response times. The results were calculated using a cloud analyst simulation. After comparing simulation results with the four methods, the author may conclude that the Balanced Throttled

load balancing approach has the fastest average response time.

6. SIMULATION AND EVALUATION

New algorithms and approaches to cloud computing must be thoroughly verified in a simulation before being deployed in the real world. It may be quite costly and time-consuming to create a functional cloud computing environment in which to evaluate freshly suggested algorithms and methods. As a result, the simulator is used to model both the cloud environment and the implementation of policies. The simulator does a great job of finding algorithms with little time and money spent on it [17]. In this research, we used the Cloud Analyst simulation tools together with the Round-Robin and Throttled algorithms to model and assess the suggested (OTA) method. We assess variables such as the total response time of the cloud system and the processing time of the data center.

A cloud analyst's main functions are as follows [17]:

- The user-friendly interface of Cloud Analyst facilitates experimentation.
- It's easy to run several tests with the same or different settings and see the outcomes visually using Cloud Analyst.
- A cloud analyst has remarkable flexibility and customization options.

Fig -4: Snapshot of Simulator

6.1 Simulation Setup

Table -1: UserBase Settings Variable

Name	Region	Requests per User per Hr	Data Size per Request (bytes)	Peak Hours Start (GMT)	Peak Hours End (GMT)	Avg Peak Users	Avg Off-Peak Users
UB1	0	60	128	8	10	400000	40000
UB2	1	60	128	10	12	100000	10000
UB3	2	60	128	1	3	300000	30000
UB4	3	60	128	20	22	150000	15000
UB5	4	60	128	22	24	50000	5000
UB6	5	60	128	9	11	80000	8000

The best way to check an algorithm is via simulation.

Where:

Peak Hour: Period of day with the most users online

Average Peak Users: The typical number of people using the service during peak hours.

Average Off-Peak Users: Number of users on average who log in outside of peak hours.

7. RESULTS AND ANALYSIS

Here, run the simulation three times, representing two distinct strategies. Particularly:

- When using the simulator for the first time, it is recommended that the Round Robin policy be used.
- To use the Throttled policy for a second time.
- The third time around, implement the policy using our suggested OTA methodology.

7.1 Case 1: Run a simulation using 25 simulated virtual machines.

For case 1, we picked six user bases that correspond to the six geographical areas of the world and five data centers. Many users visit the application after work for around two hours each evening. Every internet user makes a new request every five minutes. Each data center includes five virtual computers, and each user base has the characteristics indicated in Table -1.

Table -2: Parameters for Virtual Machines

Data Center	Number of VM	Memory	Bandwidth
DC1	5	512	1000
DC2	5	512	1000
DC3	5	512	1000
DC4	5	512	1000
DC5	5	512	1000

Table -3: Specifics about the Inner Workings of Each Data Center

Name	Region	Arch	OS	VMM	Cost per VM $/Hr	Memory Cost $/s	Storage Cost $/s	Data Transfer Cost $/Gb	Physical HW Units
DC1	0	×86	Linux	Xen	0.1	0.05	0.1	0.1	5
DC2	1	×86	Linux	Xen	0.1	0.05	0.1	0.1	2
DC3	2	×86	Linux	Xen	0.1	0.05	0.1	0.1	1
DC4	3	×86	Linux	Xen	0.1	0.05	0.1	0.1	1
DC5	4	×86	Linux	Xen	0.1	0.05	0.1	0.1	1

	Avg (ms)	Min (ms)	Max (ms)
Overall response time:	175.40	37.01	535.10
Data Center processing time:	93.74	0.09	230.13

Fig -5: Round Robin Algorithm

	Avg (ms)	Min (ms)	Max (ms)
Overall response time:	110.44	37.03	514.15
Data Center processing time:	47.66	0.09	96.33

Fig -6: Throttled Algorithm

	Avg (ms)	Min (ms)	Max (ms)
Overall response time:	86.62	37.03	514.14
Data Center processing time:	24.11	0.09	90.51

Fig -7: Optimized Throttled Algorithm (OTA)

Chart -1: The End Outcome of a Simulation Using 25 Virtual Machines

Our OTA method is faster at both processing at the Data Center and responding to user input than the Throttled and Round-Robin algorithm, but only by a small margin. That's why we ran another round of comparisons using 50 virtual PCs with the identical settings as before.

7.2 Case 2: Run a simulation using 50 simulated virtual machines.

For case 2, we picked six user bases that correspond to the six geographical areas of the world and five data centers. A large number of users visit the application after work for around two hours each evening. Every internet user makes a new request every five minutes. Each data center includes 10 virtual computers, and each user base has the characteristics indicated in Table -1.

Table -4: Parameters for Virtual Machines

Data Center	Number of VM	Memory	Bandwidth
DC1	10	512	1000
DC2	10	512	1000
DC3	10	512	1000
DC4	10	512	1000
DC5	10	512	1000

Table -5: Specifics about the Inner Workings of Each Data Center

Name	Region	Arch	OS	VMM	Cost per VM $/Hr	Memory Cost $/s	Storage Cost $/s	Data Transfer Cost $/Gb	Physical HW Units
DC1	0	×86	Linux	Xen	0.1	0.05	0.1	0.1	5
DC2	1	×86	Linux	Xen	0.1	0.05	0.1	0.1	2
DC3	2	×86	Linux	Xen	0.1	0.05	0.1	0.1	1
DC4	3	×86	Linux	Xen	0.1	0.05	0.1	0.1	1
DC5	4	×86	Linux	Xen	0.1	0.05	0.1	0.1	1

	Avg (ms)	Min (ms)	Max (ms)
Overall response time:	137.58	37.03	510.13
Data Center processing time:	65.39	0.09	193.19

Fig -8: Round Robin Algorithm

	Avg (ms)	Min (ms)	Max (ms)
Overall response time:	94.99	37.03	468.13
Data Center processing time:	32.53	0.09	89.36

Fig -9: Throttled Algorithm

	Avg (ms)	Min (ms)	Max (ms)
Overall response time:	79.04	37.03	438.13
Data Center processing time:	16.56	0.09	83.46

Fig -10: Optimized Throttled Algorithm (OTA)

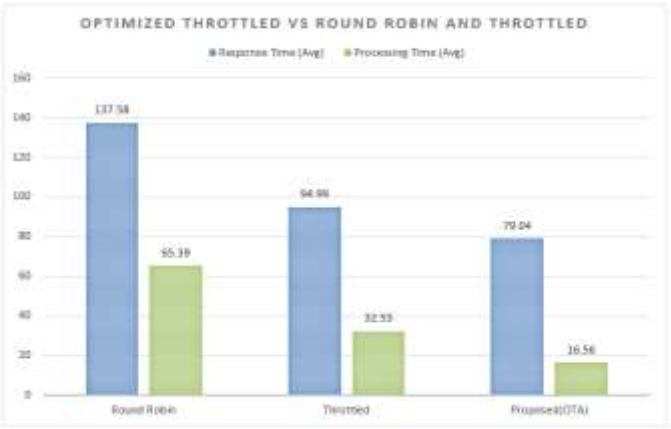

Chart -2: The End Outcome of a Simulation Using 50 Virtual Machines

Based on the experiments' outcomes in the first two cases, the results show that OTA is faster than both of the other methods when it comes to data center processing and system responsiveness. Load balancing is improved by the OTA algorithm compared to the Throttled and Round-robin methods.

8. CONCLUSIONS

When it comes to improving corporate efficiency and customer happiness with cloud computing, minimizing response times is the biggest issue. Throttled Load Balancing Algorithm, Round Robin Algorithm, and Proposed Algorithm are three of the most well-known dynamic load-balancing algorithms; we compared them while keeping these considerations in mind. The proposed algorithm's response time was found to be faster than that of the other two techniques. The OTA algorithm was created by expanding on the ideas behind the Throttled algorithm for balancing load. In order to find the best method for allocating our resources, which will lead to a more efficient distribution map, we have taken into account the Proposed Algorithm. New modified improvisation algorithms may be developed and implemented in the real world as the focus of future studies.

REFERENCES

[1] A. M, N. N. Sharma, and M. A. S., "An Enhancement of Throttled Load Balancing Algorithm in Cloud using Throughput," *International Journal of Circuit Theory and Applications*, vol. 9(15), pp. 7603–7611, 2016.

[2] G. J. Mirobi and L. Arockiam, "Dynamic Load Balancing Approach for Minimizing the Response Time Using An Enhanced Throttled Load Balancer in Cloud Computing," *IEEE Xplore*, Nov. 01, 2019.

[3] A. Jyoti, M. Shrimali, S. Tiwari, and H. P. Singh, "Cloud computing using load balancing and service broker policy for IT service: a taxonomy and survey," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, no. 11, pp. 4785–4814, Feb. 2020.

[4] E. Jafarnejad Ghomi, A. Masoud Rahmani, and N. Nasih Qader, "Load-balancing algorithms in cloud computing: A survey," *Journal of Network and Computer Applications*, vol. 88, pp. 50–71, Jun. 2017.

[5] H. Ren, Y. Lan, and C. Yin, "The load balancing algorithm in cloud computing environment," Dec. 2012.

[6] Amrutanshu Panigrahi, B. Sahu, Saroj Kumar Rout, and Amiya Kumar Rath, "M-Throttled: Dynamic Load Balancing Algorithm for Cloud Computing," pp. 3–10, Oct. 2020.

[7] S. Garg, R. K. Dwivedi, and H. Chauhan, "Efficient utilization of virtual machines in cloud computing using Synchronized Throttled Load Balancing," in *2015 1st International Conference on Next Generation Computing Technologies (NGCT)*, Sep. 2015.

[8] N. Xuan Phi, C. T. Tin, L. N. Ky Thu, and T. C. Hung, "Proposed Load Balancing Algorithm to Reduce Response Time and Processing Time on Cloud Computing," in *International journal of Computer Networks & Communications*, vol. 10, no. 3, pp. 87–98, May 2018.

[9] G. Soni and M. Kalra, "A novel approach for load balancing in cloud data center," in *2014 IEEE International Advance Computing Conference (IACC)*, Feb. 2014.

[10] A. A. Alkhatib, A. Alsabbagh, R. Maraqa, and S. Alzubi, "Load Balancing Techniques in Cloud Computing: Extensive Review," *Advances in Science, Technology and Engineering Systems Journal*, vol. 6, no. 2, pp. 860–870, Apr. 2021.

[11] S. G. Domanal and G. R. M. Reddy, "Optimal load balancing in cloud computing by efficient utilization of virtual machines," in *2014 Sixth International Conference on Communication Systems and Networks (COMSNETS)*, Jan. 2014.

[12] A. Dhari and K. I. Arif, "An Efficient Load Balancing Scheme for Cloud Computing," in *Indian Journal of Science and Technology*, vol. 10, no. 11, pp. 1–8, Mar. 2017.

[13] S. H. Sabeti and M. Mollabgher, "Proposing a load balancing algorithm with an integrative approach to reduce response time and service process time in data

centers," in *Brazilian Journal of Operations & Production Management*, vol. 16, no. 4, pp. 627–637, Nov. 2019.

[14] P. A. Pattanaik, S. Roy, and P. K. Pattnaik, "Performance study of some dynamic load balancing algorithms in cloud computing environment," in *2015 2nd International Conference on Signal Processing and Integrated Networks (SPIN)*, Feb. 2015.

[15] S. G. Domanal and G. R. M. Reddy, "Load Balancing in Cloud Computing using Modified Throttled Algorithm," in *2013 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM)*, Oct. 2013.

[16] S. Y. Mohamed, M. H. N. Taha, H. N. Elmahdy, and H. Harb, "A Proposed Load Balancing Algorithm Over Cloud Computing (Balanced Throttled)," in *International Journal of Recent Technology and Engineering (IJRTE)*, vol. 10, no. 2, pp. 28–33, Jul. 2021.

[17] Hamid Shoja, Hossein Nahid, and R. Azizi, "A comparative survey on load balancing algorithms in cloud computing," Jul. 2014.

BIOGRAPHIES

Tanjina Jahan is pursuing her M.Sc degree in Computer Science and Engineering (CSE) at Dhaka University of Engineering and Technology (DUET).

Prof. Dr. Mohammod Abul Kashem received his B.Sc. and M.Sc.Engg. Degrees from State University "Lvivska Polytechnica," Ukraine, in 1996 and 1997, respectively. In 2001 he earned Ph.D. in Control Systems and Processes from National University "Lviv Polytechnic" Ukraine. Subsequently, Dr. Kashem completed his Post Doctorate fellowship at University Lumiera Lyon2, France. (Erasmus Mundas Scholarship, European Commission) in 2016 and he was appointed as a professor at the CSE Department of Dhaka University of Engineering and Technology (DUET) in the year 2013.

Md. Toufecul Islam is pursuing his M.Sc degree in Computer Science and Engineering (CSE) at Dhaka University of Engineering and Technology (DUET).