EAI Endorsed Transactions
on Scalable Information Systems

Research Article EAI.EU

A Dynamic Scalable Auto-Scaling Model as a Load Balancer in the Cloud Computing Environment

Saroja Kumar Rout[1,*], JVR Ravindra[2], Anudeep Meda[3], Sachi Nandan Mohanty[4], Venkatesh Kavididevi[5]

[1,3,5] Department of Information Technology, Vardhaman College of Engineering (Autonomous), Hyderabad, India,
[2]Center for Advanced Computing Research Laboratory(C-ACRL) Department of Electronics & Communication Engineering, Vardhaman College of Engineering (Autonomous), Hyderabad, India,
[4] School of Computer Science & Engineering (SCOPE), VIT-AP University, Amaravati, Andhra Pradesh, India,

[*]Corresponding author: Saroja Kumar Rout, Email: rout_sarojkumar@yahoo.co.in ; ORCID: 0000-0001-9007-3665,
[2] JVR Ravindra, Email: ravindra@vardhaman.org, , [3]Anudeep Meda, Email: anudeep.m9@gmail.com, [4] Sachi Nandan Mohanty, Email: sachinandan09@gmail.com, [5] Venkatesh Kavididevi, Email: venkateshkavididevi@gmail.com

Abstract

INTRODUCTION: Cloud services are becoming increasingly important as advanced technology changes. In these kinds of cases, the volume of work on the corresponding server in public real-time data virtualized environment can vary based on the user's needs. Cloud computing is the most recent technology that provides on-demand access to computer resources without the user's direct interference. Consequently, cloud-based businesses must be scalable to succeed.
OBJECTIVES: The purpose of this research work is to describe a new virtual cluster architecture that allows cloud applications to scale dynamically within the virtualization of cloud computing scale using auto-scaling, resources can be dynamically adjusted to meet multiple demands of the customers.
METHODS: An auto-scaling algorithm based on the current implementation sessions will be initiated for automated provisioning and balancing of virtualized resources. The suggested methodology also considers the cost of energy.
RESULTS: The proposed research work has shown that the suggested technique can handle sudden load demands while maintaining higher resource usage and lowering energy costs efficiently.
CONCLUSION: Auto-scaling features are available in measures in order groups, allowing you to automatically add or remove instances from a managed instance group based on changes in load. This research work provides an analysis of auto-scaling mechanisms in cloud services that can be used to find the most efficient and optimal solution in practice and to manage cloud services efficiently.

Keywords: Cloud Computing, Auto-Scaling, Virtualization, Virtual Machine

Received on 18 May 2023, accepted on 25 May 2023, published on 03 July 2023

Copyright © 2023 Rout *et al.*, licensed to EAI. This is an open access article distributed under the terms of the CC BY-NC-SA 4.0, which permits copying, redistributing, remixing, transformation, and building upon the material in any medium so long as the original work is properly cited.

doi: 10.4108/eetsis.3356

1. Introduction

Keeping operational efficiency while reducing computational complexity, ensuring energy efficiency, and limiting data exchange is necessary to overcome contention in the Internet of Things (IoT) era.

The event-triggering paradigm, which helps to achieve these objectives in a wide range of engineering

applications, including remote monitoring, performance control, and infrastructure management [1, 2, 3, 4], is at the forefront of accomplishing these objectives. Recently, Cloud Computing [5,6,7] has gained popularity as a new enterprise model for providing on-demand services to users as needed. The cloud provides on-demand access to services via a network as part of a powerful computing paradigm. Cloud computing models can be divided into three categories: software as services (SaaS), platform as services (PaaS), and infrastructure as services (IaaS). By provisioning physical resources such as processors, disc storage, and broadband networks, virtualization technology [9, 11, 12] plays a key role in Cloud Computing. A virtualized platform or an abstracted physical resource is what is often referred to as virtualization. On-demand access to huge, elastic resources can be provided by Cloud computing systems. Nevertheless, ensuring adequate resource availability to meet peak user demand can be costly due to fluctuations in user demand. The source is insufficient to handle peak demands if the user maintains only minimal computing resources. Dynamic scalability, therefore, is a crucial component of Cloud Computing.

In a dynamic resizing configuration, the virtual machine is resized to better meet the demands of the new resource requirements [12]. Dynamic resizing is an effective solution to under or over-provisioned virtual machines. In contrast, Amazon's EC2 business model does not work with this model. It is not necessary to rent a specially resourced unit in Amazon EC2 [13], because virtual machines are the basis of the service. Dynamic resizing cannot solve load balancing because it can only grow the resources of one virtual machine. A virtualized cloud computing environment has been represented [14]. An auto-provisioning system, an on-premise load balancer, and a virtual cluster monitor system make their architecture ideal for virtualized web applications. Virtual machines can be rented, which are not limited to web applications, so users can run many applications on them. A user cannot access resources from other groups if their virtual machines belong to a certain group for security purposes. In some scenarios, the priorities of the flows need to change in the middle of the communication. However, the major problem is that the packets of different applications are not served at the destination by their expected priority ratio even if the priority does not change [10].

A VPC (virtual private cloud) in a cloud computing environment faces the problem of unequal traffic load distribution across its subnets. This chapter examines how load is distributed among the servers within the VPC. In order to distribute the load equally among the servers, auto scaling and load balancing should also be utilized in conjunction [15]. In order to satisfy SLA and budget constraints, an optimization algorithm identifies and parameters the economic system configuration [16]. Over-provisioning can be avoided with the help of auto-scaling, which boosts resource efficiency and reduces costs. Instead of making a company buy resources before starting an application, resources can be gradually added or removed as needed as demand changes over time.

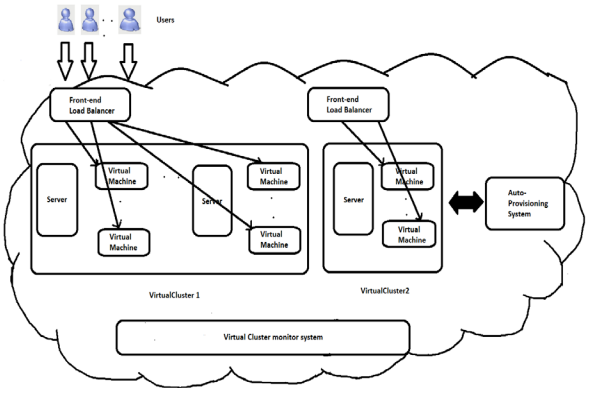

Figure 1. Architecture of the Auto-scaling in a Cloud

The research work describes a virtual cluster environment with dynamic-scaling applications deployed in virtual machines. Following is a summary of the rest of the paper: A virtualized cloud computing environment is illustrated in Section 2. The proposed algorithm for auto-scaling is described in Section 3. Section 4 represents the simulation and result work. Section 5 describes the future direction and conclusions.

2. Architecture Design

In the Cloud computing environment, web services and parallel processing applications are considered in two scenarios. No matter how many users are served by a web service, the speed of response must be as fast as possible. To meet the needs of large and small organizations, the number of servers and web service components should be dynamically scaled by the Cloud service system. By sharing tasks among a large number of computing nodes (virtual machines), the cloud provides users with access to computing power on par with supercomputers. Performing a task efficiently requires a certain number of computing nodes that users may not be aware of. As a result, under-provisioning and over-provisioning occur frequently. The computing nodes in a Cloud computing system should be scaled following the workload. Figure 1 depicts a scalable architecture that effectively handles these two scenarios. Three key components are necessary to implement this architecture: a front-end load balancer, a cluster monitoring system, and an auto-provisioning and auto-scaling algorithm. By using a front-end load balancer in a virtual cluster, requests can be balanced across virtual machines running the same application. All virtual clusters running on the Cloud collect data on resource usage from their VMs. As the workload of the virtual cluster to which the

VMs belong changes, the auto-provisioning system expands/contracts the number of VMs horizontally. It is now possible to perform request balancing between the VMs in the virtual cluster when the virtual cluster's application consumes most of the resources, including the front-end load balancer.

2.1. Load balancer and target groups

To build a web server, several features must be considered one of them is the load balancer. Using the Load Balancer, the user's load is distributed among multiple servers to balance it. If one availability zone server fails, we can distribute the load to another availability zone server. By spreading our server across multiple zones, we increase its availability.

a) AWS Cloud offers three types of load balancers:

1. Network Load Balancer
2. Application Load Balancer and
3. Classic Load Balancer

Network load balancer – Here data can be transferred using TCP and UDP. This system works at Layer 4 and is capable of processing millions of requests at once. A request is balanced based on an IP address, port number, and destination address using the Hash algorithm. The health check for our server can be enabled with load balancing if the server does not respond. If the server does not respond, a 504 error will result.

Application load balancer – This service balances HTTP and HTTPS traffic at the Layer 7 level. A Target Groups-based routing is used in this case. Requests are transferred on their own without using any algorithms.

Classic load balancer – It represented AWS's first load balancer. However, this load balancer does only basic load balancing among servers, so AWS does not recommend using it.

b) Target groups

There will be a Target Group assigned to each load balancer. Load balancers use Target Groups to determine where traffic should be sent and to which server. For the load balancer to distribute loads to the servers in the target groups, it checks the target group first and then the servers that are registered within it. Each server should thus be registered within the target groups. A server that does not register becomes idle, and the load is not distributed to it. Load balancers will enable health checks automatically if one server fails, so traffic is directed to a healthy server. You automatically add a target group listener when you create a target group. In this case, the listener serves as a routing rule. You can specify a port and an IP address to use this listener for more advanced routing. The load balancing of servers is made possible by adding them to target groups without interfering with other servers, and in case of failure, they can be deregistered. As a default, Target Group routes traffic using the Round Robin algorithm among its targets [servers]. A load balancer and its associated EC2 instances are shown in figure- 2.

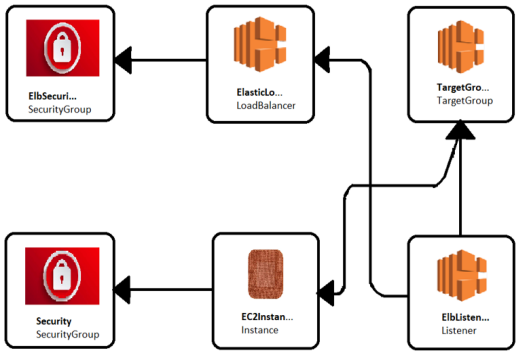

Figure 2. Load Balancer and Target Groups

2.2. Auto scaling group (ASG):

Auto Scaling Groups are used to manage and scale servers automatically. In accordance with the configuration policies specified by ASG, a new server can be created or a server can be terminated ensuring that the server is dynamically scalable and highly available at all times. Based on the attached launch template or configuration, ASG launches a new server. As an example, ASG terminates an unhealthy server in one zone before launching a new one if a server fails in that zone. It is possible to specify a server's minimum, maximum, and desired capacity. New servers will be launched by the ASG as the load increases. With ASG, you can reduce workload and save money by automatically terminating idle servers if the load drops.

a) Scaling policies

An ASG has Scaling policies that determine when new servers will be launched and when idle servers will be terminated based on those policies.

1. Target Tracking Scaling
2. Step Scaling
3. Simple Scaling

b) Launch configuration

An ASG server configuration is based on templates. A launch configuration specifies the types of servers and software to be installed before the launch. An ASG must have attached a launch configuration within it when it was created since, with the help of the launch configuration, it can launch a server with the configurations it specifies. Launch Configuration and Scaling Policies are tied to ASG, as shown in Figure 3. As part of the web server

configuration, NGINX software is installed to serve web pages to browsers.

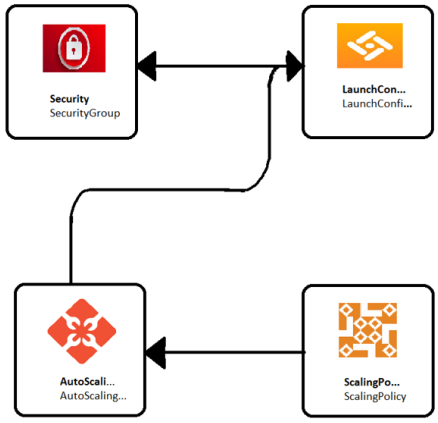

Figure 3. Auto Scaling Group

3. Load Balancing System

Web applications are distributed across a front-end load balancer as part of the proposed work architecture. Front-end load is handled by Apache HTTP Load-Balancer using redirected web applications. Cloud systems can automatically add new web servers to virtual clusters by modifying the configuration of the Apache HTTP Load-Balancer. As the virtual cluster expands, HTTP requests can be processed more quickly.

Algorithm: Auto-Scaling System

Input: n: number of Clusters
V_α: In a Virtual Cluster, the same computation system is run by multiple virtual machines
VM_β: In the virtual system, the number of active sessions
VM_{Max}: The maximum number of sessions for a virtual machine in i^{th} Cluster
VM_{Y1}: The upper threshold for a session
VM_{Y2}: The low-threshold level for a session
$VM_{\hat{C}}$: Virtual machines above the upper threshold is recorded in a virtual machine set

Output: An FLB is a set of front-load balancers

1. For i = 1 to k
2. For each VM ∈ $V_{\alpha i}$
3. If $(VM_\beta / VM_{Max} \geq VM_{Y1})$
4. set $e = e + 1$
5. If $(VM_\beta / VM_{Max} \leq VM_{Y2})$
6. set $b = b + 1$
7. The VM should be recorded to $VM_{\hat{C}}$
8. IF $(e == |V_{\alpha i}|)$ then
9. Using the same system as VC, provision and start a new virtual machine
10. The new VM should be added to FLB
// Set Front Load Balancer
11. If $(b \geq 2)$ then
12. According to each virtual machine in $Ebelow$
13. IF $(VM_\beta == 0)$ then
14. From FLB, remove the VM //Destroy
// the virtual machine by setting the
// front load balancer
15. Empty $VM_{\hat{C}}$

a. Algorithm for autoscaling on the cloud

When a web application has too many HTTP sessions active, a virtual cluster monitor can identify it. Using the Virtual Cluster Monitor System, distributed computing tasks can be monitored for excess virtual machine usage in a virtual cluster. Based on the scaling indicator data, the provisioning system scales up or down according to a scaling algorithm. Installing software on a virtual computer creates a virtual appliance describes in figure 1. A group of virtual appliances on a virtual machine is called a virtual appliance picture. A virtual appliance image, which comes with a guest OS and apps, can do away with the expenses of installing, configuring, and maintaining large software stacks. To simplify the provisioning process, the Cloud system stores an image template that contains the virtual machine appliance image template and its accompanying web server. A virtual cluster can be provisioned quickly by using the associated appliance image template.

The most crucial metrics for measuring service level agreements (SLAs) and quality of service for web service applications are network bandwidth and the session count (QoS). Our auto-scaling technique for web service applications is shown in Figure 2. It begins by identifying which virtual machines (VMs) currently have the bandwidth and active sessions over or below preset threshold values. As soon as all VMs reach a specified upper threshold for network bandwidth and active sessions, a new virtual machine is created, launched, and added to the front-end load balancer. If you wish to terminate an idle VM from the system, you must first remove it from the front-end load balancer and terminate any active sessions that it has.
The most important indicators of how a virtual cluster's workload varies are the physical computing resources needed for distributed computing tasks, such as CPU and memory consumption. Figure. 3 displays our auto-scaling

method for distributed computing jobs. The program begins by determining which virtual machines are utilizing physical resources at rates that are above or below predetermined thresholds. An additional VM is created, provisioned, launched, and performs the same computing tasks in the virtual cluster if the aggregate resource utilization of all the virtual machines exceeds the set upper threshold. If the idle VM's resource utilization drops below a predetermined lower level and there is at least one other VM that is not using any resources, the virtual cluster will delete the idle VM.

4. Simulation and Result Analysis

The simulation purpose, we are using CloudSim open-source framework, which can be used to simulate cloud computing infrastructure and services. It is used for modelling and simulating a cloud computing environment as a means for evaluating a hypothesis prior to software development in order to reproduce tests and results. As shown in Figure 4, CPU usage, reaction time, workload, and VM allocations are depicted during autoscaling. VM counts are computed and updated every time a sample is taken in computing environment.

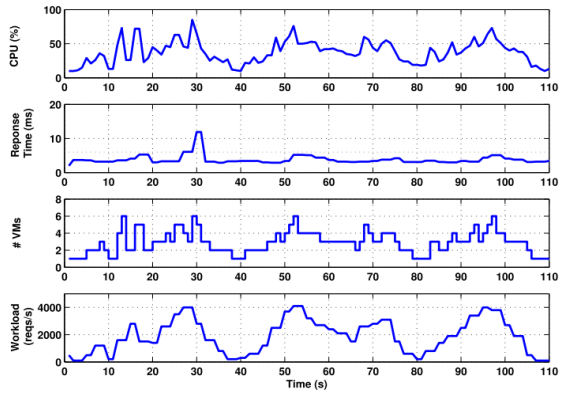

Figure 4. Auto scaling utilization of system

Figure 4 displays the host's actual and anticipated CPU load time for a virtualized environment. a proportion of the instance's CPU resources that represents the rolling average of each database's total CPU usage. To set alerts and examine CPU utilization over a lengthy period, like 24 hours. Also, it describes CPU utilization and its response time during data computing and also VMs utilization with respect to workload distribution in cloud environment.

Figure 5. CPU Utilization

Figure 5 shows the CPU's actual and predicted load usage over time. A summary of our maximum CPU usage recommendations for single-region and multi-region instances is shown in Table 1. If a zone or region closes, these parameters are used to ensure your instance can continue to handle traffic (for multi-region instances) even if the other zones and regions are not available.

Table 1. High CPU utilization in cloud computing region

Metric	The maximum number of instances per region	Maximum per region for multi-region instances
Total high priority	66%	47%
24-hour aggregate has been rounded	91%	91%

After a disc utilization or latency issue has been noticed, use this statistic to determine the bytes-based nature of back-end disc I/O. It is challenging to pinpoint a problem based solely on disc I/O throughput, as seen in Figure 6.

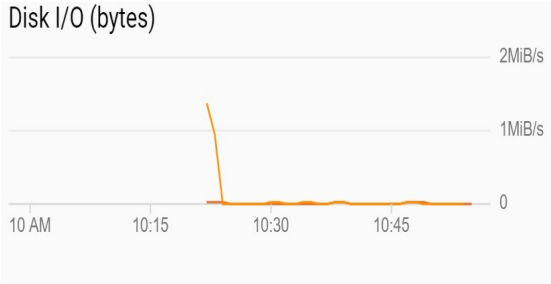

Figure 6. I/O Bytes used in disk

Use this statistic to identify the type of back-end disc I/O based on the number of disc I/O operations per second

when a disc utilization or latency issue has been identified (IOPS). This statistic shows the disc IOPS in a cloud environment after the appliance transformed logical I/O into physical I/O using the share settings and software RAID settings shown in figure 7.

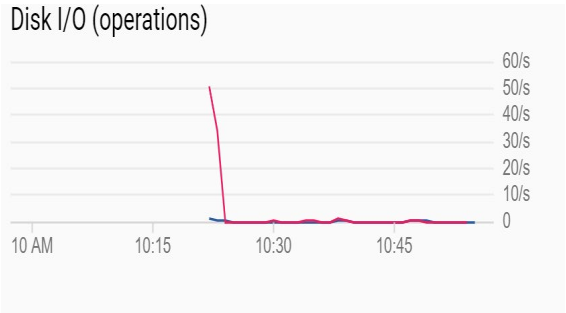

Figure 7. Disk I/O Operations

The total amount of data that the instance received across all network interfaces is described in the system. This metric shows how much incoming network traffic is directed at a single instance. The amount delivered by the instance across all network interfaces based on time factors. This metric measures the amount of network traffic that leaves a certain instance. Bytes transmitted over a network to and from a cloud server in the computing environment shown in Figure 8

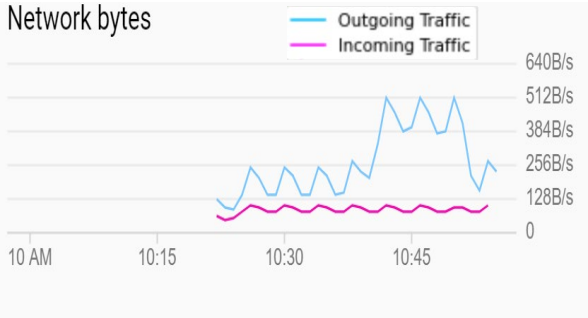

Figure 8. Network Bytes for incoming network traffic

The total amount of packets that the instance has received across all network interfaces. This statistic measures the volume of incoming traffic on a single instance in units of packets. The total amount of packets sent across all network interfaces by the instance. This statistic counts the number of packets sent on a single instance to represent the volume of outgoing traffic.

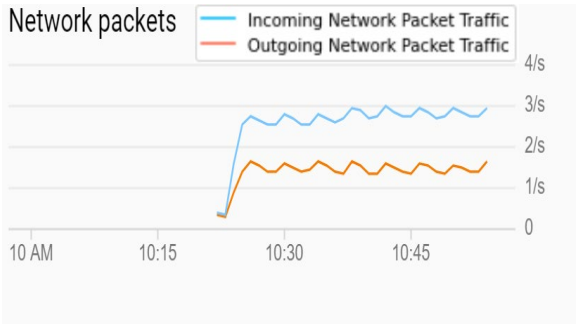

Figure 9. Incoming & Outgoing Network Packet Traffic

5. Conclusion

Using virtual clusters for web applications and distributed computing tasks, this study examines two scaling scenarios. In addition to the load balancer, a virtual cluster monitor, and auto-provisioning systems are included in the Cloud Computing architecture. The front-end load balancer routes and balances user requests to Cloud services installed in a virtual cluster. The virtual cluster monitor system gathers information about how virtual machines use physical resources in a virtual cluster. A virtual cluster's auto-provisioning mechanism dynamically provisioned the virtual machines based on the number of active sessions or resource consumption. The resources can then be released after the idle virtual machines have been destroyed. On the other side, eliminating virtual machines that are idle can lower energy costs. Our research has shown that the suggested algorithm is capable of managing peaks in demand, maintaining higher resource utilization, and consuming less energy. The Cloud system's auto-scaling mechanism is crucial for improving resource utilization and lowering infrastructure and management expenses. Managed instance groups allow instances to be added or removed automatically based on changes in load. The applications can handle surges in traffic more smoothly with the aid of autoscaling, which also lowers costs when fewer resources are needed.

References

[1] Liu, J., Yang, Y., Li, H. and Geng, Y., 2021. Event-triggered output-feedback control for networked switched positive systems with asynchronous switching. *International Journal of Control, Automation and Systems*, *19*(9), pp.3101-3110.

[2] Dhar, N.K., Verma, N.K. and Behera, L., 2017. Adaptive critic-based event-triggered control for HVAC system. *IEEE Transactions on Industrial Informatics*, *14*(1), pp.178-188.

[3] Mohapatra, P.K., Rout, S.K., Bisoy, S.K. and Sain, M., 2022. Training Strategy of Fuzzy-Firefly based ANN in Non-linear Channel Equalization. *IEEE Access*..

[4] Sahu, B., Mohanty, S. and Rout, S., 2019. A hybrid approach for breast cancer classification and diagnosis. *EAI*

Endorsed Transactions on Scalable Information Systems, *6*(20).

[5] Panigrahi, A., Sahu, B., Rout, S.K. and Rath, A.K., 2021. M-Throttled: Dynamic Load Balancing Algorithm for Cloud Computing. In *Intelligent and Cloud Computing* (pp. 3-10). Springer, Singapore.

[6] Saxena, D., Singh, A.K. and Buyya, R., 2021. OP-MLB: An online VM prediction based multi-objective load balancing framework for resource management at cloud datacenter. *IEEE Transactions on Cloud Computing*.

[7] Shafiq, D.A., Jhanjhi, N.Z. and Abdullah, A., 2021. Load balancing techniques in cloud computing environment: A review. *Journal of King Saud University-Computer and Information Sciences*.

[8] Tian, W., Xu, M., Zhou, G., Wu, K., Xu, C. and Buyya, R., 2021. Prepartition: Load Balancing Approach for Virtual Machine Reservations in a Cloud Data Center. *arXiv preprint arXiv:2110.09913*.

[9] Nayyer, M.Z., Raza, I., Hussain, S.A., Jamal, M.H., Gillani, Z., Hur, S. and Ashraf, I., 2022. LBRO: Load Balancing for Resource Optimization in Edge Computing. *IEEE Access*.

[10] Mishra, T.K. and Tripathi, S., 2018. Congestion control and fairness with dynamic priority for ad hoc networks. *International Journal of Ad Hoc and Ubiquitous Computing*, *29*(3), pp.208-220.

[11] Xen,http//www.xen.org.

[12] Kernel-based VirtualMachine(KVM),http://www.linux-kvm.org.

[13] AmazonEC2,http://aws.amazon.com/ec2/.

[14] Gabhane, J.P., Pathak, S. and Thakare, N.M., 2021. Metaheuristics Algorithms for Virtual Machine Placement in Cloud Computing Environments—A Review. *Computer Networks, Big Data and IoT*, pp.329-349.

[15] Bilgaiyan, S., Mishra, B.S.P., Ansari, R. and Sagnika, S., 2022. A Collaborative Cloud Model of Auto Scaling With Load Balancing for Effective E-Commerce. In Empirical Research for Futuristic E-Commerce Systems: Foundations and Applications (pp. 116-130). IGI Global.

[16] Fé, I., Matos, R., Dantas, J., Melo, C., Nguyen, T.A., Min, D., Choi, E., Silva, F.A. and Maciel, P.R.M., 2022. Performance-Cost Trade-Off in Auto-Scaling Mechanisms for Cloud Computing. Sensors, 22(3), p.1221.