INTERNATIONAL JOURNAL
FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Hybrid Cost Optimized Load Balancing in Cloud

S. Jyothsna, Dr. K.Radhika
[1]Osmania University
[2]Professor, CBIT

Abstract: Cloud infrastructure goods and a collection of servers are housed in a cloud datacenter. When a client requests that a particular task be performed, the request is routed to a server that has the resources to do the task. A suitable virtual machine (VM) operating on it in turn handles the request. The current research suggests a cloud-based cost-optimized hybrid load balancing method that chooses the virtual machine to assign jobs depending on the VM's availability and its active connections at that precise moment. Also, it examines the prices of the various virtual machines and data transfer cost to choose the one that will fulfill the request for the lowest cost. In almost all test cases, the suggested technique reduced overall response time and cost. Resources are de-provisioned from a pool of resources once they have been used for a set of specific workloads. The quality of service(QOS) for the end customers must be ensured during this processing period. The main objective of this research is to make it possible for end customers to access services at the lowest possible cost and response time.
Keywords: Cloud Datacenter; Virtual machine; Load balancing; Optimum response time; Cost

I. INTRODUCTION

A cloud services provider manages a remote datacenter where computer resources, applications, servers (both physical and virtual), data storage, development tools, networking capabilities, etc. are available on demand over the internet (or CSP). Cloud service companies can utilize the full potential of their data center resources through virtualization.

The biggest obstacle to offering effective services to users without violating SLAs (Service level agreements) is load balancing in a cloud. The load balancing algorithm's goal is to evenly spread the load among the servers in order to ensure the consistency of cloud services.
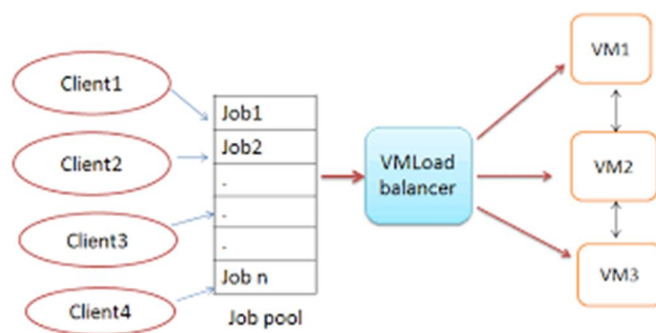

Fig.1.Load Balancing in Cloud

There are two approaches for load balancing.
1) *Static Load Balancing*: Methods of static load balancing distribute traffic based on the past performance of the nodes without accounting for the servers' or system's current condition. Because load and request volume on cloud servers can change quickly, a static load balancing method might not effectively disperse the load in a fluid environment like the cloud.
2) *Dynamic Load Balancing:* Dynamic load balancing (DLB) strategies offer a way to dynamically distribute the loads based on intelligent and self-adaptive distribution, and the system's present condition is watched for each allocation.

In the cloud, cost-optimized load balancing distributes the burden among several servers to increase effectiveness and cut expenses. Here are few methods to accomplish this.
a) *Autoscaling:* With autoscaling, the number of servers is automatically adjusted based on workload. The number of servers is decreased when there is little demand. This guarantees effective resource use and cuts down on wasteful spending.

b) *Use of multiple availability zones:* If one zone fails, the workload can be transferred to the other zones using the cloud provider's multiple availability zones. Due to the ability to divide the effort among several zones, this also aids in load balancing.

c) *Spot Instances:* Spot instances are unused computing resources that are being sold for less money. Spot instances can be used to significantly reduce costs for non-critical tasks. However, we must exercise caution when employing spot instances for crucial workloads because the cost can change depending on demand.

d) *Employ a load balancer:* To split the workload amongst several servers, use a load balancer. Traffic can be automatically distributed by effective load balancers according to workload. This guarantees effective resource use and cuts down on wasteful spending.

e) *Monitor and Optimize*: Always keep an eye on the workload and adjust the resource allocation in accordance with traffic patterns. By doing so, any bottlenecks can be found and the resources can be used more effectively.

An algorithm for cost-optimized hybrid distributed load balancing in the cloud (COHLBC) is suggested in this paper. This algorithm is hybrid because it determines whether a virtual machine is available as in a throttled algorithm and assigns the tasks to virtual machines based on the minimal number of active monitoring load monitoring load balancing algorithm. The suggested load balancing method also chooses the lowest-cost virtual machine out of all those that meet the previous requirements.

This approach is used in the cloud analyst simulator and contrasted with three baseline load balancing algorithms—round robin, throttled, and active monitoring—to see which is the best. The four algorithms were examined for the performance criteria total cost, datacenter processing time, and overall response time. When used in the cloud analyst simulator, the proposed cost-optimized load-balancing method significantly cut costs when compared to existing load balancing techniques.

The cloud analyst simulator, which is divided into six different areas throughout the world, allows users to add resources, load balancing rules, and test processing by sending requests from anywhere.

The rest of the paper is organized as follows: A brief summary of the research on load balancing techniques in the cloud is given in Part II. The workflow and algorithm are described in Part III. The simulation environment is described in Part IV. Comprehensive experimental analysis can be found in Section V. The conclusion and future work described in Section VI.

II. LITERATURE REVIEW

In this section, existing and current load balancing algorithms will be examined.

The Priority Weighted Round algorithm An enhancement to the weighted round-robin algorithm that takes into account each task's importance before allocating it to a separate virtual machine(VM)[1].This method determines which VMs the tasks should be dynamically assigned to based on their priority. This algorithm may not be flexible in a dynamic environment because the weights are static and based on the user class, and the priority may vary as user requests are received dynamically in the cloud.

Chowdary and Fatema [2][5] Re-modified throttled algorithm (RTMA) balances loads by updating and maintaining two index tables to keep virtual machines' availability statuses as busy and available and to recognise available VMs with "Available Index"-like throttled algorithm. and addressed the current load on the VM to enhance the Throttled restriction. The performance of this enhanced throttled algorithm will be at its peak when serving the same userbase of requests; nevertheless, it should be further enhanced for other userbases.

In order to enhance VM throughput and achieve load balancing between VMs based on makespan, cost, and resource use, Boonhatai Kruekaewand Warangkhana Kimpan [3] devised a hybrid optimised bee colony algorithm. The ABC algorithm imitates bee colonies' foraging strategies, which call for adaptation to changing habitat and food circumstances. This strategy took into account choosing appropriate VMs based on determining each VM's fitness.

Ant colony optimisation (ACO)[4] makes use of natural parallelism to maximise results by taking into account the constructive feedback of the ants. Ants must deal with a condition that resembles overtraining in reinforcement learning systems, even though this is essentially self-organization rather than learning. The method guarantees convergence, but because it relies on random judgements and the pace of convergence is uncertain, its theoretical analysis is difficult.

In their comparison of the datacenter broker policies "optimise response time," "nearest data centre," and "reconfigure dynamically with load" in cloud analyst, Biswajit Nayak et al. [6-7] found that, on average, both closest data centre and optimised response time rules functioned similarly. Although though dynamic reconfiguration with load policy takes longer than the other two, it still performs well since load balancing takes a little longer.

Three Dynamic Tables were utilised in the dynamic load balancing approach [8] to contain several virtual machines with various configurations and processing rates in various data centres. It focuses on allocating jobs to virtual machines based on load, and it is frequently scheduled as an overloaded virtual machine. The load balancer can find available VMs in Data Centers more quickly thanks to dynamic tables that update their list of available machines on a regular basis. Moreover, jobs may be distributed effectively among various machines with various processing speeds. demonstrated a reduction in the average makespan and response times.

Aggarwal et al. [9] used the algorithm weighted active monitor load balancing policy, which determines the weight of each virtual machine based on factors including RAM, bandwidth, the number of processors, and processor speed. virtual machines were chosen with the highest weights that are accessible to do the work after weighing each virtual machine. By intelligently assigning incoming requests based on the present condition of VMs with weight values, the method controls the server's load.

A genetically based improved min-min task scheduling algorithm for load balancing in cloud computing was proposed by Rajput, (2016) [10]. In order to reduce the makespan and maximise resource efficiency, this work suggested an improved load balanced min-min (ILBMM) algorithm employing genetic algorithms (GA). First, the virtual machine's task execution time is determined. Hence, the virtual machine's (VM) task's minimum or maximum duration is known. Then, to improve task execution, a genetically based strategy was applied to the task's million instructions (MI) and the virtual machine's million instructions per second (MIPS). employed the GA's fitness function, crossover, and mutation.

III. WORKFLOW AND ALGORITHM

In addition to developing a hybrid algorithm that took into account the availability status from throttled and the minimum active connections from active monitoring, the proposed algorithm also attempted to reduce service costs by choosing the active virtual machine (VM) with the lowest cost from the list of available machines to fulfil user requests.

For each job allocation, the proposed optimal method took into account the following:

1) A virtual machine's change in state from "AVAILABLE" to "BUSY"
2) Calculate the load (active connections) and update the VM count for each task allocation.
3) Prices vary based on the data center's location and the size of the virtual machine.

Fig.2 describes the above sequence of workflow, selection of VM for task allocation. For each request, this keeps happening. The consumer prefers to employ cloud services with reduced costs because the cloud itself is a cost-based service paradigm
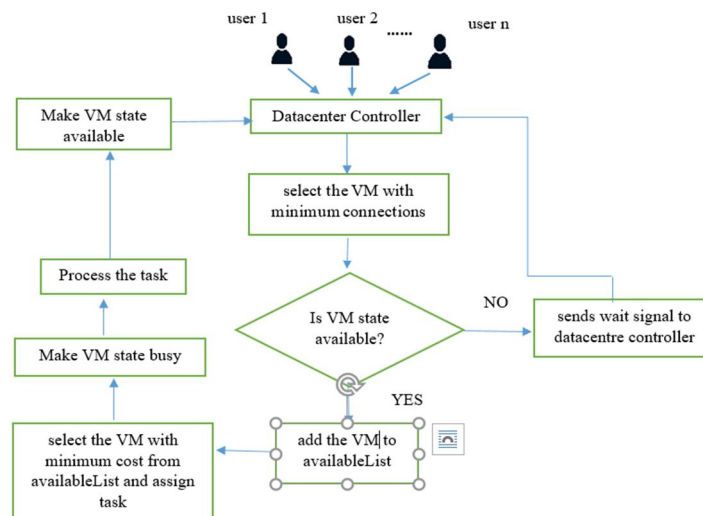

Fig. 2. Workflow Schedule

As soon as the user request is received, the datacenter controller adds it to the queue. allocates the request to that VM once it has chosen it using the COHDLBC algorithm Fig.3. The Primary objective of this algorithm is to reduce customer costs and response times while still providing effective services. once after the request is processed, the VM state changes to "AVAILABLE" and the datacenter removes the request from queue.

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue III Mar 2024- Available at www.ijraset.com

Algorithm: Cost Optimized Hybrid Distributed Load Balancing in Cloud (COHDLBC)

Input: User requests, Selected region, Service demand for the tasks

Output: Select VM for allocation of task/tasks

Initialize all VM allocation status to AVAILABLE in the VM State list

While (New request is received by the data centre controller)

    do

    Data Centre Controller queues the request.

  if (CurrentAllocation.size () < VmStatesList.size ())

   if(state.equals(VirtualMachinestate.Available))

        AvailableList.add(Key)

   end if

    for each Key in AvailableList

     if cost(Key) < mincost then

        mincost =cost

       VM_{id} =Key

    end if

   end for

    Allocatevm(VM_{id})

   end do

  VMstatelist.put(VM_{id}.VirtualMachineState.BUSY)

   Data Centre Controller removes a request from   beginning of queue.

VMstatelist.put (VMid.VirtualMachineState .AVALABLE)

  end

Fig.3. Cost Optimized Hybrid Distributed Load Balancing in Cloud(COHDLBC)

IV.   SIMULATION ENVIONMENT

The CloudAnalyst tool, a graphical user interface built on the Cloud Sim Architecture, is used to run the simulation. CloudAnalyst uses the CloudSim architecture to model Data Centers. Due to the distributed nature of cloud infrastructure, requests will originate from all over the world and need to be handled carefully. The real-time scenario with six different geographic locations is provided by the CloudAnalyst simulator. In other words, the number of users from various areas can be detected based on the application. Fig. 4 displays the CloudAnalyst simulation output.

The simulator is incredibly adaptable and offers various resources for testing, including virtual machines, data centres, bandwidth, and more. There are six different geographic locations (six separate continents) taken into account. For all user locations, a single time zone is taken into account.one-hundredth of the total users from each continent is considered and it is assumed that only 5% of all users are online during peak hours and that off-peak hours have a tenth of peak hour usage. Each data centre is capable of hosting a certain number of virtual machines that are required for specific applications.

Fig. 4. Snapshot of simulation output

The configuration simulation is shown below Fig.5 where 5 userbases representing set of users from different regions of the world request are received by the datacenter and scheduled tasks based on optimize response time datacenter broker policy.
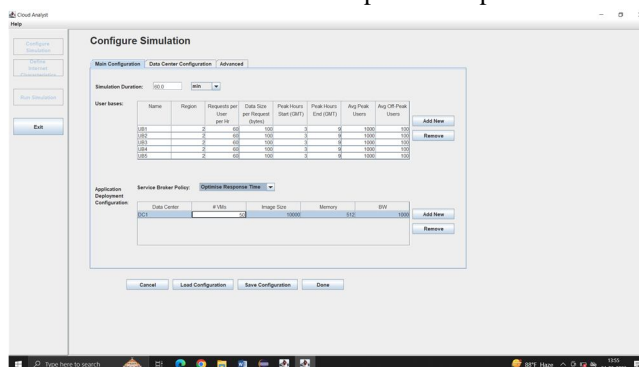

Fig. 5. Configuration Simulation

By customising the cloudanalyst tool's components, the proposed algorithm has been subjected to analysis and simulation and is compared to other available load balancing techniques. The parameters for the application deployment have been set.

TABLE I.   application deployment configuration

No. of Data centers	1-6
No. of VMs	5-500
Image Size	10000 MIPS
Memory	512 MB
Bandwidth	1000 Mbps
User Bases	1-6
Service Broker Policy	Optimize Cost

V.   RESULTS AND DISCUSSION

The proposed methodology (COHLBC) is contrasted with well-known heuristic load balancing algorithms in order to assess its efficacy. The proposed algorithm is implemented in cloud analyst simulator by varying the number of datacenters, virtual machines and userbases with different test cases. Results of four algorithms are examined, and the outcomes of simulations of the various scenarios taken into account for the performance analysis.

Each test case involves changing and testing datacenter broker policies, such as optimizing cost, optimizing response time, and optimising the closest datacenter with the proposed load balancing algorithm. Following test cases were considered and compared proposed algorithm with existing algorithms in cloud analyst.

CASE I

Experiment Name	ActiveMonitoring
Number of Datacenters	2
Number of Virtual Machines	50
Userbases	5
Service Broker Policy	Optimize Cost

CASE II

Experiment Name	COHDLBC
Number of Datacenters	2
Number of Virtual Machines	50
Userbases	5
Service Broker Policy	Optimized Cost

CASE III

Experiment Name	Round Robin
Number of Datacenters	2
Number of Virtual Machines	50
Userbases	5
Service Broker Policy	Optimize Cost

CASE IV

Experiment Name	Throttled
Number of Datacenters	2
Number of Virtual Machines	50
Userbases	5
Service Broker Policy	Optimized Cost

The Results of various test cases were tabulated in Table 2.

TABLE II.   Comparison Of Load Balancing Algorithms

Load Balancing Algorithm	Overall Response Time	Data transfer cost
Round Robin	310.20	0.32
Active	309.01	0.32
Throttled	308.72	0.32
COHDLBC	48.21	0.03

The performance of the proposed algorithm is analyzed using the following metrics.

1) *Overall Response Time:* This is the time required for a load-balancing algorithm to complete each of the assigned tasks or the moment the system finishes its final task.

  Response time = Fin_Time --- Arr_Time +TDelay-----(1)

         where Fin_Time is the finish time of user request and Arr_Time is the arrival time of user request and TDelay is the transmission delay.

TDelay can be determined as

$$TDelay = TLatency + Ttransfer------------------(2)$$

Where TLatency is the network latency and T transfer is the time taken to transfer the size of single request from source location to destination.

2) *Data transfer Cost:* Data transfer cost is calculated based on the amount of data tranfererred (measured in gigabytes,terabytes etc.) using network capacity ie. bandwidth cost and The cost of VM selected based on the algorithm.

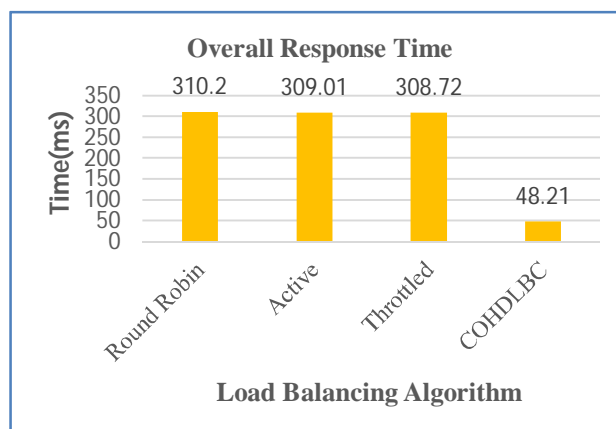   data transfer cost← VM cost+ bandwidth cost

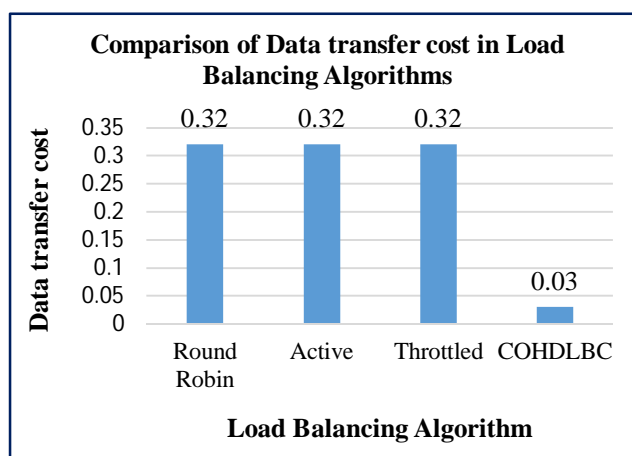Fig.6. Comparison of Overall Response Time

Fig.7. Comparison of Cost

The metrics overall response time and cost were compared in various load balancing algorithms like Round Robin, Throttled, Active monitoring along with proposed Cost Optimized Hybrid Distributed Load Balancing in Cloud (COHDLBC). The results shows that, the overall response time is reduced 84% approximately and data transfer cost is reduced 90% in proposed COHDLBC compared to existing load balancing algorithms round robin, active monitoring and throttled algorithms as shown in Fig.6. and Fig.7.

The datacenter broker schedules task/tasks to selected virtual machine based on datacenter broker policy. three datacenter broker policies optimize response time, closest datacenter and reconfigure dynamically with load were available in cloud analyst. A new broker policy 'optimize cost' is designed to schedule task/tasks based on the cost so that cloud services can be provided to the customers with minimum cost.
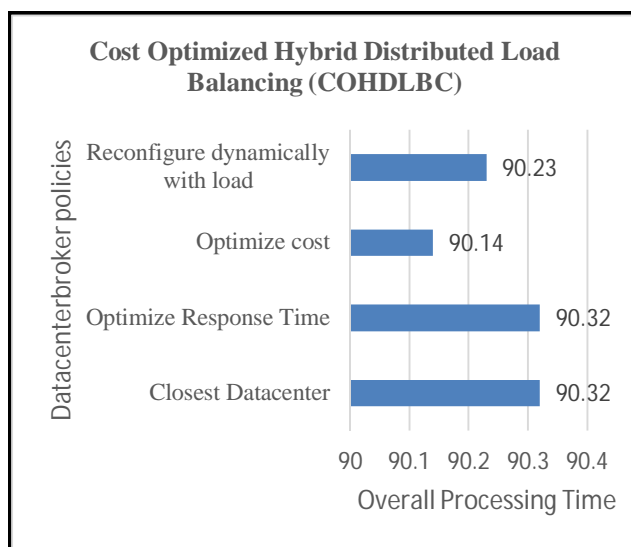

Fig.8. COHDLBC with datacenter broker policies

The proposed algorithm COHDLBC gave almost same processing time with all types of datacenter broker policies as it is observed in Fig.8., This proves that the proposed algorithm works well with any type of broker policy i.e. any datacenter chosen based on the distance, response time or cost.

VI. CONCLUSION

The proposed load balancing algorithm COHDLBC improved the overall response time and datacenter processing time to process requests in different test cases with varying number of datacenters and virtual machines. The experimental results shows better results in case of response time and cost and also proved that resource utilization is improved as cloudlet loads are shared equally on different virtual machines. The data transfer cost reduced much in proposed algorithm compared to three baseline load balancing algorithms in cloud analyst simulator.

The improvement of COHDLBC is because of selecting virtual machines based on minimum connections and cost and also selecting only available virtual machines at that moment so that overload can be prevented by selecting only minimum connections and readily available virtual machine to process the task given. It also reduced cost as the algorithm finds minimum cost virtual machine in every iteration of task allocation.

REFERENCES

[1] Katangur, S. Akkaladevi and S. Vivekanandhan, "Priority weighted round robin algorithm for load balancing in the cloud," *2022 ieee 7th international conference on smart cloud (smartcloud)*, shanghai, china, 2022, pp. 230-235, doi: 10.1109/smartcloud55982.2022.00044.

[2] Shieu n. le1 and hung c. tran2 , "ITA: the improved throttled algorithm of load balancing on cloud computing", international journal of computer networks and communications(ijcnc) vol.14 no.1 january 2022

[3] B.kruekaew, w. kimpan "Multi-objective task scheduling optimization for load balancing in cloud computing environment using hybrid artificial bee colony algorithm with reinforcement learning", doi 10.1109/access.2022.3149955, february 17, 2022

[4] Neetu Sharma, Sonal and Puneet garg"Ant colony based optimization model for QoS-based task scheduling in cloud computing environment", https://www.sciencedirect.com/journal/measurement-sensors,2022

[5] Fatema tuj johora, iftakher ahmed, md. ashiqul islam shajal, rony chowdhory, "a load balancing strategy for reducing data loss risk on cloud using remodified throttled algorithm", international journal of electrical and computer engineering (ijece), vol. 12, no. 3, june 2022, pp. 3217~3225 issn: 2088-8708, doi: 10.11591/ijece.v12i3.pp3217-3225

[6] Dalia Abdul Kareem Shafiq, N.Z. Jhanjhi and Azween Abdullah,"Load balancing techniques in cloud computing environment: A review" https://doi.org/10.1016/j.jksuci.2021.02.007

[7] Biswajit nayak, bhubaneshwari bisoyi, prashanth kumar pattnaik, "data center selection through service broker policy in cloud computing environment" ,https://doi.org/10.1016/j.matpr.2021.06.185

[8] Soumen swarnakar;ritik kumar; saurabh krishn;chandan banerjee; (2020). improved dynamic load balancing approach in cloud computing . 2020 IEEE 1st international conference for convergence inengineering(icce),doi:10.1109/icce50343.2020.9290602

[9] Aggarwal, V. B.; Bhatnagar, Vasudha; Mishra, Durgesh Kumar (2018). [Advances in Intelligent Systems and Computing] Big Data Analytics Volume 654 || WAMLB: Weighted Active Monitoring Load Balancing in Cloud Computing. , 10.1007/978-981-10-6620-7(Chapter 65), 677–685.

[10] Shyam singh rajput and virendra singh kushwah,"A genetic based improved load balanced min-min task scheduling algorithm for load balancing in cloud computing", 2016 8th international conference on computational intelligence and communication networks (cicn)

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)