INDIAN JOURNAL OF SCIENCE AND TECHNOLOGY

RESEARCH ARTICLE

The Proposed IT-TALB in Infrastructure as a Service Cloud

S Shanmugapriya[1]*, **N Priya**[2]

1 Assistant Professor, PG Department of Information Technology and BCA, Dwaraka Doss Goverdhan Doss Vaishnav College, University of Madras, Chennai, Tamil Nadu, India
2 Associate Professor, Research Department of Computer Science, Shrimathi Devkunvar Nanalal Bhatt Vaishnav College for Women, University of Madras, Chennai, Tamil Nadu, India

OPEN ACCESS

Received: 04-01-2024

Accepted: 19-03-2024

Published: 15-04-2024

Citation: Shanmugapriya S, Priya N (2024) The Proposed IT-TALB in Infrastructure as a Service Cloud. Indian Journal of Science and Technology 17(16): 1654-1662. https://doi.org/10.17485/IJST/v17i16.3274

* **Corresponding author**.

priyadgvc17@gmail.com

Funding: None

Competing Interests: None

Published By Indian Society for Education and Environment (iSee)

ISSN
Print: 0974-6846
Electronic: 0974-5645

Abstract

Objectives: The purpose of the proposed IT-TALB load balancing algorithm is to dynamically allocate the user's workload to the appropriate virtual machine in an Infrastructure as a Service (IaaS) cloud environment. **Methods:** This research work includes several key procedures. The user's workloads are distributed to the data center controller (DCC), which in turn uses the ECO-SBP service broker policy to select the efficient data center (DC) for processing the loads. The DCC forwards the load to the selected DC, and the IT-TALB load balancer picks the best Virtual Machine (VM) using CloudAnalyst simulation tool for load allocations according to metrics such as its size, current number of loads, and load size. IT-TALB partitions the available and busy VMs separately and stores them in the TreeMap structure. This algorithm also incorporates the scalability of the given VM when the load size is not compatible with the existing VMs by extending the resources of underutilized VMs. **Findings:** The research finding demonstrates that the proposed IT-TALB algorithm improves IaaS cloud performance compared to the existing algorithms. It achieves optimum load balancing, reduces the searching time of the VM, avoids the load waiting time, improves throughput, minimizes the response time, and enhances the resource utilization ratio. IT-TALB yields a throughput and resource utilization ratio of 98 to 99 percent. **Novelty:** The novelty of this research is that the IT-TALB algorithm incorporates the scalability of the underutilized VM and also introduces new metrics such as throughput and resource utilization ratio in the CloudAnalyst simulation tool for assessing the performance of the proposed algorithm. This study provides information for analyzing the proposed IT-TALB strategies with the existing two algorithms such as TLB and TALB in order to show its performance.

Keywords: Cloud Computing; Infrastructure as a Service; Load Balancing; Throttled Load Balancing; Virtual Machine

1 Introduction

Cloud computing delivers various computing resources such as virtual servers with processing capabilities, memory units, storage and networking connections to an end user as an Infrastructure as a Service (IaaS) over the internet on a demand basis[1]. Nowadays several users from various geographical locations access the cloud resources for processing their requests. Virtualization is a technique in the IaaS cloud environment that creates multiple virtual servers also called virtual machines (VMs) on top of every single physical server to allow multitenancy and achieve efficient resource utilization and offered to the customers for processing their workloads[2,3].

Load balancing (LB) is a technique used in the IaaS cloud model to evenly distribute the user's workload across various VMs. LB can be categorized as static or dynamic. Static LB needs prior knowledge about the systems, and loads can be allocated to the VM based on its previous system performance. Static LB is simple and easy to implement, but the limitations of static LB are that once a load is assigned to a particular VM, it cannot be moved to any other VM due to failure or overloading of the VM, and it does check the VM's current status while assigning the load to a homogenous environment[4]. In dynamic LB, load is allocated according to the present state of a VM, and it is possible to provide scalability of a VM, i.e., it is possible to scale down and scale up on a demand basis and also perform load migration when necessary[5].

Cloud simulation software called CloudAnalyst was created specifically for use in an IaaS cloud model. CloudAnalyst enables LB simulations to assess various scenarios and determine the most efficient LB strategy. The simulation tool includes a load balancer, service broker policy, data center, virtual machines, and user bases that accept user workloads that can be configured to simulate various scenarios with different parameters like response time, processing time, cost, resource utilization and workload balancing[6].

Due to the heterogeneous VMs i.e., different size VMs in terms of the capacity of the processor, memory, storage and bandwidth it holds and voluminous amount of user loads and the dynamic nature of cloud systems (dynamic cloud system means the ability of cloud systems to adapt and change in real-time to changing circumstances, demands, and requirements in order to achieve high availability, fault tolerance, workload balancing, resource scheduling, and reliability), it is very important to know the current state of the available VMs for efficient resource management. Dynamic load scheduling is best suited for heterogeneous IaaS cloud systems to achieve optimum resource utilization, avoid overloading or under loading of the VMs, and provide efficient resource allocation[7].

It is difficult to choose the best VM for a particular workload because of the rising demand for cloud resources, the diversity of cloud resources, and the exponential growth of user loads. The ineffectiveness of the IaaS cloud system as a whole is exacerbated by improper workload distribution among VMs, which also raises costs and leads to poor resource utilization[8].

To solve these issues, an LB strategy is needed, one that selects the most appropriate VM for effectively processing each load. This could also lead to improved cloud system performance, reducing load response times, processing times, and cost, improving the throughput of a virtual machine, and also assisting in distributing the workload among different VMs[9]. Even though many authors have already developed several LB algorithms, there are still many significant challenges with LB that are not addressed by the current techniques, including VM overloading, availability, migration, scalability, and efficient resource utilization.

The author outlined that the elementary static Round Robin (RR) LB algorithm selects a VM at random, distributes the load to it when it comes, and then circularly distributes the remaining loads among the other VMs. The VM processes the load using the time quantum. That is, when a time slice comes, the virtual machine processes that specific load. Due to its inability to validate VM sizes during allocation and its inability to facilitate load relocation in the event of a VM failure or overload, the RR algorithm is limited to use only in homogeneous IaaS cloud environments. RR uses too few resources because it doesn't check if the VM is suitable[10,11].

The author sketched that the Equally Spread Current Execution (ESCE) procedure is dynamic, i.e., it needs the current status of each VM to check the present allocations of each VM by counting the number of loads and their availability status. When loads arrive from the users, it allocates the loads to the available VM, which has a smaller number of current allocations than other VM's and allocates the remaining loads by following the same procedure. The limitations of ESCE are that it does not check whether the VM can process the load or not, which leads some VMs to be overloaded while others are underutilized[12,13].

The author drew that the throttled load balancing (TLB) strategy is a dynamic strategy that schedules the user's load to the available VMs according to the load size and VM size to improve the selection of VMs to efficiently process each load. TLB stores VM details such as its ID, status (which specifies the available and busy VMs), and size using the HashMap structure. Initially, it keeps all the VMs available. When the load arrives, TLB finds all the available VMs suited for the load. If the proper VM is present for each load, then it's allocated; if not, then it keeps the load waiting in the queue till it finds the proper VM, which increases the load's waiting time and the load's response time. The drawback of the TLB strategy is that it increases the searching time of a particular VM when the number of VMs is increased in the HashMap table[14–16].

We proposed the dynamic TALB algorithm in our previous study to overcome the drawbacks of the TLB strategy. It changes the storage structure of the VMs and allocates the user's load to the accessible virtual machines (VMs) to reduce the searching time of the VM by enhancing the Throttled Load Balancing (TLB) algorithms. It chooses the best VM according to the load size, VM size, and VM existing load to advance the selection of VM to balance the workload among them. The TALB algorithm uses the TreeMap structure, which has two partitions, namely AVM and BVM, to keep all the available and busy VMs, respectively. Initially, all the VMs are available, and once the workload arrives from the users, the TALB chooses the VM from the available list and then considers all the busy VMs. The limitations of the TALB algorithm are that if none of the VMs are suitable for a load while being allocated, then the load has to wait until the appropriate VM is available, which increases the response time and lowers the throughput and resource utilization. The TALB has been compared and examined with three different LB algorithms: RR, TLB, and ESCE, using the CloudAnalyst simulation tool. The TALB algorithm surpasses RR, ESCE, and TLB in terms of improving performance by providing low searching time for the VM, load response time, processing time, and processing cost. Even though the TALB strategy reduces the searching time, response time, processing time, and cost compared to RR, ESCE, and TLB, it does not consider the loads that are not suitable for any of the VMs in both the available and busy TreeMap lists. These loads have to wait till they get the suitable VM, which increases the load waiting time and also the response time[17].

Therefore, the goal of the suggested study is to enhance the TALB algorithm and propose the IT-TALB algorithm, which resolves the problem of TALB by incorporating scalability in the given available VMs to match the waiting loads to avoid loads waiting time, minimize the response time, increase the throughput and resource utilization ratio.

2 Methodology

2.1 The Proposed IT-TALB Strategy

This research focuses on improving the Throttled Adapted Load Balancing (TALB) strategy, which was proposed in our earlier study, and proposes a dynamic Improved Throughput-Throttled Adapted Load Balancing (IT-TALB) algorithm.

The proposed IT-TALB algorithm enhances the TALB strategy and overcomes the limitations of TALB by eliminating the waiting time for assigning the load and incorporating scalability in VMs. IT-TALB implements a TreeMap structure for strong VM details. The TreeMap is divided into an available and busy list for keeping available and busy VMs, respectively. Within each list, the VMs are stored in a sorted order based on their size and present load, i.e., the VMs with the lowest load and high capacity are kept at the top. In this manner, it stores all the VMs in the TreeMap. If the load is not suitable for any VM, then it increases the resource capacity (vCPU, RAM, and storage) of the least utilized VM based on the capacity requirement of the load and allocates the load to avoid delay, thus increasing the throughput, minimizing the response time, and optimizing the resource utilization ratio of 96%. Response time can be obtained from Equation (1).

$$VM\ (i)RT\ =\ (\textstyle\sum_{k=0}^{n} VM(i).LDS(k)/(VM(i)_{BW}\ +\ VM(i)_{TD}))\ +\ VM(i)_{PT} \tag{1}$$

Where VM $(i)_{RT}$ refers to the response time of the VM, VM(i). LDS(k) refers to the total load size of the VM; VM$(i)_{BW}$ refers to the bandwidth capacity of the VM; VM$(i)_{TD}$ refers to the transmission delay; and VM(i)PT refers to the processing time. The utilization of each VM is calculated by Equation (2).

$$VM(i).Util\ =\ VM(i).vCPU.Util + VM(i).RAM.Util +\ VM(i).Storage.Util\ +\ VM(i).BW.Util \tag{2}$$

Where VM(i).Util denotes the current capacity utilization of the i^{th} VM, VM(i).vCPU.Util denotes the virtual CPU utilization, VM(i).RAM.Util denotes the RAM utilization, VM(i).Storage.Util denotes the storage utilization and VM(i).BW.Util denotes the bandwidth utilization of the i^{th} VM. Total VM utilization is represented in Equation (3).

$$TotVM_{Util} = \textstyle\sum_{k=0}^{n} (VM\,(k)\,.Util)) \tag{3}$$

Where VM(K).Util denotes all the available VMs. Throughput of a VM is obtained by Equation (4)

$$VM_{TP}\ =\ VM(i)_C\ /VM(i)_A \tag{4}$$

Where VM$_{TP}$ describes the throughput of the VM, VM$(i)_C$ describes the group of loads completed by the VM, and VM$(i)_A$ denotes the total loads allotted to the VM. The total load completion time of the VM is acquired in Equation (5).

$$VM(i)_C\ =\ T_{Loads}\ + \textstyle\sum_{k=0}^{n} VM(i).LoadSize(k) \tag{5}$$

Where T_{Loads} refers to the total loads allocated to the VM, I refers to the VM ID, and k refers to the load ID.

2.2 Steps of the proposed IT-TALB Strategy

Input: Number of Loads L1,L2,L3…Ln, Number of VMs M1,VM2,VM3,…VMn,VMStateList VM(n) =1 where n is the VM id.

Steps:

1. Loads arriving in the UserBase have been forwarded to the Data Centre Controller (DCC).
2. The size of each load is calculated according to the needed processing capacity.
3. DCC uses a service brokering policy to forward each load to various datacenters (DC).
4. The loads arriving at each DC are forwarded to the TALB load balancer.
5. The IT-TALB load balancer maintains each VM's unique id, its status in the VMStatus field, i.e., VMid->VMStatelist = available (1) or busy (0), the size of each VM, and the current number of loads allocated to each VM. Initially, the status of the VMs is assigned as 1 in its VMStatus field.
6. IT-TALB creates the TreeMap structure and splits the structure into two substructures, namely AVL and BSY, to hold all the available and busy VMs separately, where AVL has all the VMs that are not yet assigned any loads and BSY has all the VMs that have at least one load for processing.
7. IT-TALB finds the best VM for each load based on its size and the present number of loads in the AVL list, i.e., the size of the VM is larger than the size of the load (VMSize>LSize).
8. If a suitable VM is present in the AVL list, then the load is assigned to that VM, which updates its VMStatus field to 0 and moves the VM from AVL to BSY in the TreeMap table, calculates the remaining size of that VM (VMrs), and returns its VM ID to the DCC.
9. If the suitable VM is not present in the AVL list, then it checks the BSY VMs and chooses the VM that is appropriate for processing the load. If the suitable VM is present in the BSY list, then the load is assigned to that VM, which calculates the remaining size of that VM (VMrs) and returns its VM ID to the DCC.
10. If none of the VMs are suitable, then IT-TALB finds the underutilized VM and scales up its size by increasing the resource capabilities such as vCPU, memory units, and storage capacity based on the load size, assigns the loads to that VM, and returns its VM ID to the DCC.
11. IT-TALB allocates the remaining load to the VMs in this manner.
12. If the VM finishes processing the loads, it calculates the VMrs and updates the TreeMap.

2.3 Pseudo Code of IT-TALB for implementing scalability in VM

```
For each VM in the chosen DC
    Assign VM_ID[i].VMStateList = 1 //Initially set the status to 1 for all VMs
    Calculate VM_Size (vCPU+vRAM+vStorage) // Find the size each VM End For
    Construct Map<Integer,String, Integer,Integer > TM = new TreeMap<>() //Create TreeMap structure
    Construct TM.AVL, TM.BSY = new TreeMap<>() // Construct 2 sub structures under the TreeMap
    //Load Allocation to VMs
    If (UB_Load == "Yes") then
    Calculate Load_Size, Load.Assign=0
    For each VM in TM.AVL
    If (VM_SIZE>Load_SIZE) then //Search the appropriate VM
    Assign VM_ID <- Load_ID , Load.Assign = 1, return VM_ID
    Find VM_ID.VMrs and move VM_ID to TM.BSY
    End If End For
    If (Load.Assign == 0) then // Search VM in BSY list
    For each VM in TM.BSY
    If (VM_SIZE>Load_SIZE) then //Search the appropriate VM
    Assign VM_ID <- Load_ID, Load.Assign = 1, VM_ID.VMStateLIst = 0
    Find VM_ID.VMrs and update VM_ID to TM.BSY
    End If End For
    If (Load.Assign == 0) then // Scale up the VM
    Choose the underutilized VM
    VM.ID.VM_SIZE.Utilities >Expand (vCPU+vRAM+vStorage) // expand the resource capacity
```

Assign VM_ID <- Load_ID, Load.Assign = 1, VM_ID.VMStateLIst = 0
Update VM_ID in TM.BSY End If

2.4 Performance Comparison of IT-TALB with TLB and TALB

Table 1. Performance Assessment of TLB, TALB and IT-TALB

TLB	TALB	IT-TALB
Only one load is assigned to each VM simultaneously, which lowers resource utilization.	More than one load is assigned to each VM to improve resource utilization.	More than one load is assigned to each VM to improve resource utilization.
Each VM is selected based on its current size.	Each VM is selected based on its current size and present load.	Each VM is selected based on its load size, current size, and present load.
The HashMap structure is used for storing VM details.	The TreeMap structure is used for storing VM details with two partitions.	The TreeMap structure is used for storing VM details with three partitions.
VMs are not sorted in a HashMap structure, so finding the proper VM takes a long time.	VMs are sorted in a TreeMap structure.	VMs are sorted in a TreeMap structure.
Load has to wait if the proper VM is not found.	Load has to wait if the proper VM is not found.	Load need not wait if the proper VM is not found; instead, Load can be assigned to the least utilized VM by incorporating scalability to improve resource utilization, throughput, and response time.

2.5 A Flow Diagram of IT-TALB

Figure 1 depicts the pictorial representation of the procedure of the IT-TALB strategy: loads are forwarded to the DCC, which in turn arranges the load in the load queue based on the severity value using the ECO-SBP service broker policy. The service broker selects the proper DC and distributes the load to that DC. The IT-TALB algorithm keeps all the VM details in the VM, chooses the appropriate VM based on its size and present allocations, and allocates the load to that VM.

3 Results and Discussion

3.1 Simulation Setup

The experiment is conducted on the IT-TALB strategy with the existing TLB and TALB policies in terms of performance metrics such as throughput (TP), response time (RT), and resource utilization ratio (RUR) using the CloudAnalyst simulation tool in the IaaS cloud system. We have developed the proposed IT-TALB algorithm in the ITTALB.java file and included it as an IT-Throttled Adapted Load Balancer in the load balancing policy of the CloudAnalyst simulation tool under the Cloudsim.ext.Datacenter package, as shown in Figure 2.

In these experiments, we have built ten DCS (DC1 to DC10) with two physical machines in each DC, and 80 VMs have been built with varying physical configurations. 25 user bases (UB1 to UB25) have been created in six regions (R0 to R5) with varying user loads of different sizes, as depicted in Figure 3. The VM allocation policy used is time-shared, and the service broker policy is ECO-SBP, which was proposed in our previous research and is used to choose the appropriate DC for load allocations. The simulation duration has been set at 60 minutes.

3.2 Experimental Results and Analysis

For experiments, four testing scenarios have been included with DCs, VMs, UBs, and varying numbers of loads, as described in Table 2.

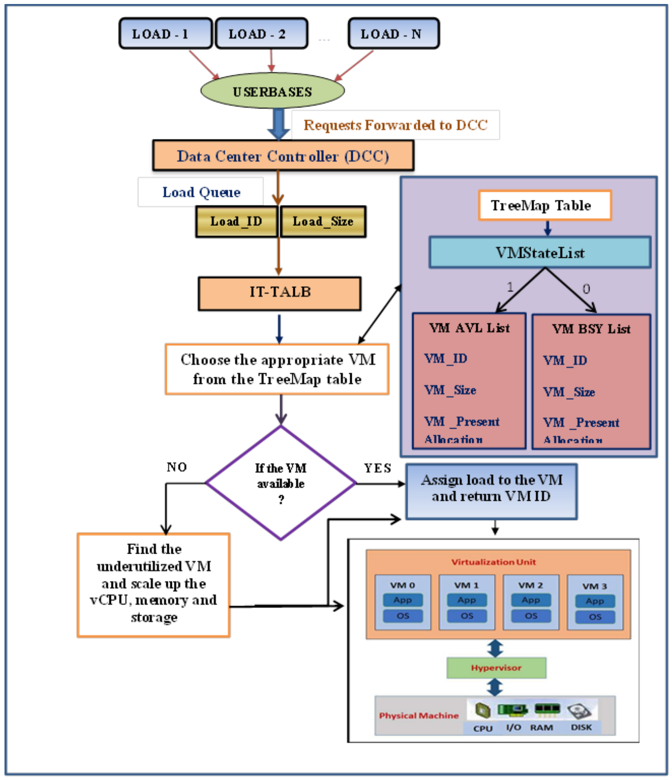

Fig 1. Load allocations and VM selections of IT-TALB

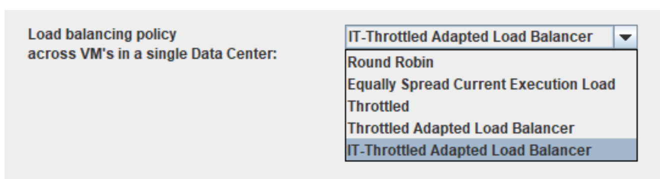

Fig 2. Proposed IT-TALB in Cloud Analyst

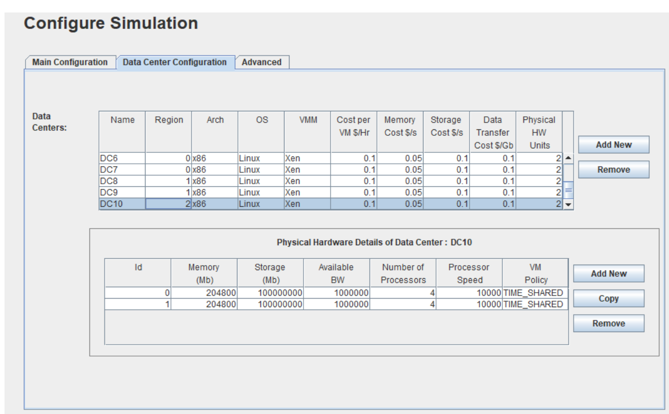

Fig 3. Configuration simulation panels of DC and Physical Machine

Table 2. Scenario Description

Scenario	Number of DCs	Number of VMs in each DC	Number of UBs	Number of Loads in each UB
1	10	80	25	10,000
2	10	80	25	15,000
3	10	80	25	20,000

3.2.1 Results of Response Time of TLB, TALB and IT-TALB

Shows the minimum, average, and maximum values of the RT of the TLB, TALB, and IT-TALB algorithms acquired in the Cloud Analyst tool based on each scenario. The values show that IT-TALB gives minimum RT values for other algorithms with the same number of loads and load sizes.

Table 3. Experimental Comparison of TLB, TALB and IT-TALB with RT

LB Algorithms	Scenario -1			Scenario -2			Scenario -3		
	RT in Milliseconds			RT in Milliseconds			RT in Milliseconds		
	Avg.	Min.	Max.	Avg.	Min.	Max.	Avg.	Min.	Max.
TLB	188.6	38.7	783.1	375.5	106.3	1053	545.5	226.2	1290
TALB	158.9	33.4	735.1	330.1	90.0	825.3	400.1	189.0	1032
IT-TALB	120.6	30.2	693.3	256.5	66.3	653.1	345.5	106.2	990.1

3.2.2 Results of Throughput of TLB, TALB and IT-TALB

Table 4 shows the TP in percentage for the TLB, TALB, and IT-TALB algorithms acquired in the Cloud Analyst tool based on each scenario. The values show that IT-TALB gives higher TP values than other algorithms with the same number of loads and load sizes, i.e., it allocates maximum loads for the available VMs in such a way that the maximum loads are completed within 60 minutes.

Table 4. Experimental Comparison of TLB, TALB and IT-TALB with TP

LB Algorithms	Scenario - 1	Scenario - 2	Scenario - 3
	TP in %	TP in %	TP in %
TLB	75	64	55
TALB	90	84	78
IT-TALB	99	98	97.5

3.2.3 Results of Resource Utilization Ratio of TLB, TALB and IT-TALB

Table 5 shows the RUR in percentage for the TLB, TALB, and IT-TALB algorithms acquired in the Cloud Analyst tool based on each scenario. The values show that IT-TALB gives higher RUR values than other algorithms with the same number of loads and load sizes, i.e., its maximum resource utilization ratio for all the VMs.

Table 5. Experimental Comparison of TLB, TALB and IT-TALB with RUR

LB Algorithms	Scenario - 1	Scenario - 2	Scenario - 3
	RUR (%)	RUR (%)	RUR (%)
TLB	47	40	35
TALB	79	75	70
IT-TALB	98	95	96

The chart displayed in Chart 1**a.** visualizes the results that compare the RT, **b.** shows the comparison of TP, and **c.** compares the RUR of each LB strategy. The suggested IT-TALB strategy appears to show improvements in throughput and optimizes resource utilization within a stipulated period.

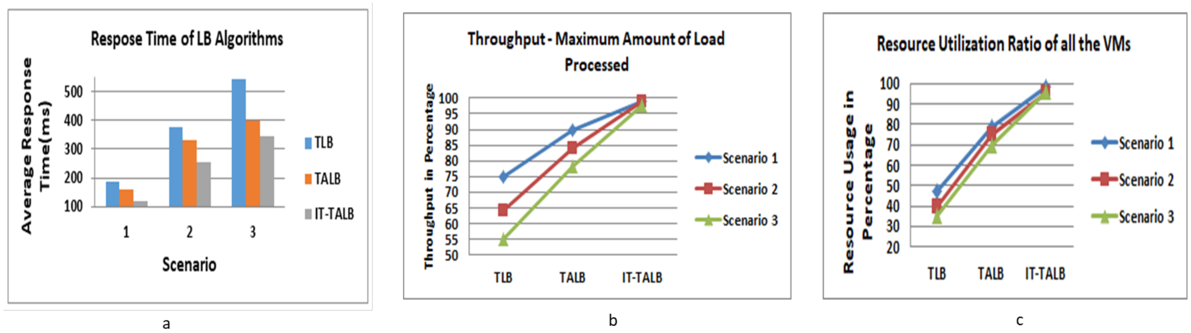

Chart 1: **Comparison Chart of a. Average Response Time, b. Throughput, and c. Resource Utilization Ratio**

By observing the results, it is ensured that the proposed IT-TALB strategy consistently produces the minimum response time, high throughput, and high resource utilization ratio as compared to other TLB and TALB strategies since it avoids the waiting time of the loads.

While analyzing the previously proposed TALB strategy with RR, ESCE, and TLB, the TALB strategy yields the minimum searching time of each VM, load response time and processing time, and also the cost. I.e., The results of the TALB strategy yields the average searching time of 163.11 milliseconds which is 48.23 percent faster than RR, 47.12 percent faster than ESCE and 53.64 percent faster than TLB. With respect to response time, the TALB yields 300.91 milliseconds which is 22.74 percent faster than RR, ESCE and TLB. With respect to data center processing time, the TALB yields 322.47 which are 24 percent faster than RR, ESCE and TLB. The cost of VM and DC for TALB is 50.76$ which reduces 50 percent than the cost used by the RR, ESCE and TLB. However, it makes some loads wait in the queue due to the unavailability of a suitable VM. Even though some VMs are available, their capacity is not fitted for these loads, which increases the response time of such loads and leads to low resource utilization. The IT-TALB strategy allocates all the loads to the VMs by incorporating scalability into the VMs to make it suitable to process all the loads without making them wait; thus, the results obtained in this research work yield a lower response time value, higher throughput and resource utilization ratio than the results in each scenario with the previous results obtained in the TALB strategy, regardless of the size of the loads, even during peak hours[17].

4 Conclusion

The cloud environment has adopted load balancing strategies to ensure that user workloads are distributed fairly among the multiple virtual machines (VMs). Load balancing techniques have been implemented into the cloud environment in order to achieve the equitable distribution of user workloads among the numerous VMs. One of the biggest challenges in load balancing is distributing the user's workload evenly across the relevant virtual machine (VM), particularly when the cloud's burden has dramatically increased and managing heavy website traffic becomes difficult. In our prior research, we introduced the TALB approach, which minimizes the TLB method's searching time by selecting the optimal virtual machine (VM) for each load. But a load must wait in the queue if it is not appropriate for any of the available virtual machines (VMs), which lengthen the load's waiting and response times.

The objective of this research work is to propose a new dynamic IT-TALB strategy is to allocate the user's workloads to the appropriate VM in an IaaS cloud platform to enhance the performance and achieves user's satisfaction. The novelty of this research is to overcome the shortcomings of the TALB strategy by incorporating scalability to the underutilized VM in the TALB strategy i.e., the VM is scaled up with its resources to match the loads which are not suitable to any of the available VMs. The suggested IT-TALB strategy is analyzed with the current load balancing procedures such as TLB and TALB in the CloudAnalyst simulation tool in different simulated experimental setups. In each scenario, the number of workloads has increased, and the throughput, response time, and resource utilization ratio are experimentally measured and compared with the TLB and TALB algorithms. By observing the results, the proposed IT-TALB achieves an optimal resource utilization ratio and higher throughput in all scenarios while reducing waiting times and response times.

The IT-TALB approach's experimental comparison results in a response time value of 345.5 milliseconds, which is 22.40% faster than the TLB strategy and 7.30% faster than the TALB strategy. A throughput value of 97.5% is obtained with IT-TALB, which is 19.5% greater than TALB and 42.5% higher than TLB. The resource utilization ratio obtained by IT-TALB is 96%, which is 26% and 61% greater than that of the TALB and TLB strategies, respectively. This suggests that the IT-TALB is operating at a very high level of efficiency, with minimal wasted resources and high throughput.

References

1) Ramzan MS, Asghar A, Ullah A, Alsolami F, Ahmad I. A Bee Colony-Based Optimized Searching Mechanism in the Internet of Things. *Future Internet*. 2024;16(1):1–15. Available from: https://dx.doi.org/10.3390/fi16010035.
2) Tasneem R, Jabbar MA. An Insight into Load Balancing in Cloud Computing. In: Proceeding of 2021 International Conference on Wireless Communications, Networking and Applications. Lecture Notes in Electrical Engineering;Singapore. Springer. 2022;p. 1125–1140. Available from: https://doi.org/10.1007/978-981-19-2456-9_113.
3) Sharma M, Kumar R, Jain A. Load Balancing in Cloud Computing Environment: A Broad Perspective. In: Intelligent Data Communication Technologies and Internet of Things;vol. 57 of Lecture Notes on Data Engineering and Communications Technologies. Singapore. Springer. 2021;p. 535–551. Available from: https://doi.org/10.1007/978-981-15-9509-7_44.
4) Rajak R, Choudhary A, Sajid M. Load balancing techniques in cloud platform: A systematic study. *International Journal of Experimental Research and Review*. 2023;30:15–24. Available from: https://dx.doi.org/10.52756/ijerr.2023.v30.002.
5) Reshan MSA, Syed D, Islam N, Shaikh A, Hamdi M, Elmagzoub MA, et al. A Fast Converging and Globally Optimized Approach for Load Balancing in Cloud Computing. *IEEE Access*. 2023;11:11390–11404. Available from: https://dx.doi.org/10.1109/access.2023.3241279.
6) Rajpoot NK, Singh P, Pant B. Load Balancing Strategies for Cloud Computing: A Simulation-Based Study. *Journal of Nano- and Electronic Physics*. 2023;15(3):1–4. Available from: https://doi.org/10.21272/jnep.15(3).03023.
7) Sansanwal S, Jain N. An Improved Approach for Load Balancing among Virtual Machines in Cloud Environment. *Procedia Computer Science*. 2022;215:556–566. Available from: https://dx.doi.org/10.1016/j.procs.2022.12.058.
8) Udayasankaran P, Thangaraj SJJ. Energy efficient resource utilization and load balancing in virtual machines using prediction algorithms. *International Journal of Cognitive Computing in Engineering*. 2023;4:127–134. Available from: https://dx.doi.org/10.1016/j.ijcce.2023.02.005.
9) Khair Y, Benlabbes H. Opportunistic Load Balancing for Virtual Machines Scheduling in a Cloud Environment. *Engineering Proceedings*. 2023;29(1):1–6. Available from: https://doi.org/10.3390/engproc2023029001.
10) Zhou J, Lilhore UK, Poongodi M, Hai T, Simaiya S, Jawawi DNA, et al. Comparative analysis of metaheuristic load balancing algorithms for efficient load balancing in cloud computing. *Journal of Cloud Computing*. 2023;12(1):1–21. Available from: https://dx.doi.org/10.1186/s13677-023-00453-3.
11) Pakhrudin NSM, Kassim M, Idris A. Cloud service analysis using round-robin algorithm for quality-of-service aware task placement for internet of things services. *International Journal of Electrical and Computer Engineering (IJECE)*. 2023;13(3):3464–3473. Available from: https://dx.doi.org/10.11591/ijece.v13i3.pp3464-3473.
12) Sharma A, Sharma KK. A Novel approach for Load Balancing distribution and storage by using Cloud Computing. *E3S Web of Conferences*. 2023;399:1–6. Available from: https://dx.doi.org/10.1051/e3sconf/202339904009.
13) Zamri AH, Pakhrudin NSM, Saaidin S, Kassim M. Equally Spread Current Execution Load Modelling with Optimize Response Time Brokerage Policy for Cloud Computing. *International Journal of Advanced Computer Science and Applications*. 2023;14(2):482–491. Available from: https://dx.doi.org/10.14569/ijacsa.2023.0140257.
14) Kumar VD, Praveenchandar J, Arif M, Brezulianu A, Geman O, Ikram A. Efficient Cloud Resource Scheduling with an Optimized Throttled Load Balancing Approach. *Computers, Materials & Continua*. 2023;77(2):2179–2188. Available from: https://dx.doi.org/10.32604/cmc.2023.034764.
15) Le HN, Tran HC. ITA: The Improved Throttled Algorithm of Load Balancing on Cloud Computing. *International Journal of Computer Networks & Communications*. 2022;14(1):25–39. Available from: https://aircconline.com/ijcnc/V14N1/14122cnc02.pdf.
16) Sivaraj V, Kangaiammal A. Load Allocation of Virtual Machines based on Enhanced Throttled Load Balancing Algorithm. *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*. 2021;12(7):1841–1848. Available from: https://turcomat.org/index.php/turkbilmat/article/view/3084.
17) Priya N, Shanmugapriya S. A New Throttled Adapted Load Balancing (TALB) Strategy for Dynamic VM Allocations in Cloud Datacenters. *International Journal of Cloud Computing*. 2024;13(3). Available from: https://doi.org/10.1504/ijcc.2024.10060349.