INDIAN JOURNAL OF SCIENCE AND TECHNOLOGY

RESEARCH ARTICLE

Examination of Cloud Simulation Platforms and Implementation of Load Balancing in CloudAnalyst

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

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

OPEN ACCESS

Received: 14-06-2024

Accepted: 24-07-2024

Published: 20-08-2024

Citation: Shanmugapriya S, Priya N (2024) Examination of Cloud Simulation Platforms and Implementation of Load Balancing in CloudAnalyst. Indian Journal of Science and Technology 17(33): 3424-3436. https://doi.org/10.17485/IJST/v17i33.1751

* **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: In light of the growing number of open-source simulation tools for load balancing in cloud computing, the objective of this research is to assess some of the most significant tools before selecting the best one depending on the requirements and goals of the research and also implementing the load balancing approach in the CloudAnalyst tool for dynamic workload (HTTP request) allocation and virtual machine selection in the Infrastructure as a Service (IaaS) heterogeneous cloud environment, with the aim of minimizing the searching time, improving the response time and datacenter processing time and also adds a new performance metrics such as makespan. **Methods:** We conducted a comparative examination of a few cloud simulators using a variety of criteria, including their core components, working models, scalability, performance metrics, simulation technique, and structural design, it incorporated and experimented with the load balancing approach with significant performance metrics such as searching time, response time, datacenter processing time and makespan in CloudAnalyst simulation tool. **Findings:** This research finding shows that, after reviewing, examining, and comparing some of the significant cloud simulation tools and their features in the Infrastructure as a Service cloud computing environment, the CloudAnalyst tool was best suited for implementing and experimenting with the load balancing strategies in a cloud environment with respect to significant performance metrics such as response time, datacenter processing time, cost, searching time and, throughput. Moreover, it evaluates the performance of the load balancing approach by implementing it in the CloudAnalyst tool and proves that there is an improvement in average response time of 270.91milliseconds, Datacenter processing time by 212.74 milliseconds and there is a reduction in searching time of 127.21 milliseconds and makespan value of 192.89 compared to existing approaches in a heterogeneous IaaS cloud environment. **Novelty:** The novelty of this research provides recommendations for researchers that the CloudAnalyst simulation tool is the ideal tool for the implementation of the load balancing

approach and also incorporates the new metrics termed makespan in a dynamic IaaS cloud environment.

Keywords: Cloud Computing; Load Balancing; Simulation Platforms; Virtual Machine; Infrastructure as a Service; Datacenter

1 Introduction

The Infrastructure as a Service model in cloud computing delivers on-demand services to its users for processing their business with greater flexibility via the internet. These services incorporate millions of computing resources, including virtual servers, processors, storage capabilities, databases, and network connections, which are pooled in various locations known as data centers, and the user's loads are transported to multiple servers in various data centers that are dispersed geographically. Cloud computing technology offers a variety of resources, such as hardware, software, memory, networks, databases, etc., as services to numerous users according to their demand for finishing their tasks via the internet at a reduced cost with less maintenance. Cloud computing models its services as "Infrastructure as a Service" (IaaS), "Platform as a Service" (PaaS), "Software as a Service" (SaaS), "Database as a Service" (DBaaS), "Communication as a Service" (CaaS), etc., depending on the kind of resources it delivers [(1)]. All the services are accessible using a pay-per-usage subscription approach with minimal management effort and interventions. IaaS is the core service model that offers virtual servers with computing resources, including processors, memory, storage, and network connections, to users for processing their applications [(2)].

Due to the flexibility of the IaaS cloud, more and more users are approaching its computing facilities, which require an efficient technique for the allocation of cloud resources such as virtual machines (VM) and data centers to the user's loads (HTTP requests). Load Balancing (LB) stands as a resource allocation policy in the IaaS cloud that helps in evenly distributing user requests among the pool of resources in such a way that it improves the system efficiency utilization of resources, reduces the response time, minimizes the cost by forwarding requests to a proper resource, and avoids overloading or under loading of loads in any resources [(3)].

Various LB algorithms have been employed for scheduling a user's load among innumerable resources based on the user's needs and parameters. LB needs a cloud computing environment for testing the scheduling algorithms, which can be done in two ways: in a real cloud environment or a simulation environment. Undertaking experiments in a real cloud environment like Amazon EC2 is expensive and may not be repeatable, scalable, or possible for some scenarios due to security reasons. Thus, the performance of various LB techniques can be experimented with by using any one of the cloud simulation tools that imitate a real environment, and the performance testing can be done repeatedly with variable workloads and is easily accessible and helpful for the researcher [(4)] [(5)].

There are several open source and proprietary cloud computing tools available, namely CloudSim, CloudAnalyst, CloudReports, CloudShed, GreenCloud, WorkflowSim, GridSim, TeachSim, iCanCloud, Ucloud, and CloudExp, etc., and there are numerous performance metrics used in LB for measuring the performance of the LB strategies. The major challenge in choosing the appropriate tool depends on the metrics and applications used.

Many authors have examined several cloud simulation tools, but the reviews that are currently available only provide a general overview of all the cloud computing tools; they do not address whether the tools are appropriate in terms of performance metrics in the particular context for load balancing, also they did not apply the load balancing approach while comparing the simulation tools. While comparing the existing studies, the main research gap of the existing studies is that they mentioned the tools that are

used for load balancing, but they did not show whether the tools are suitable for load balancing in a dynamic IaaS heterogeneous cloud environment and its implementations.

Due to the dynamic arrival of the user's loads and the heterogeneous nature of the computing resources, and in recent days many research studies have been involved in load balancing, it is important to portray the best simulation tool that allocates the workloads of the users using load balancing in a dynamic IaaS heterogeneous cloud model. The paper which was written in our previous study made a comparison of three popularly used significant load balancing algorithms as Round Robin (RR), Equally Spread Current Execution (ESCE), and Throttled Load Balancing (TALB) to select the appropriate VM and allocation of the incoming user's workloads among several VMs. RR is a basic static load balancing approach that randomly selects any one VM and allocates the first workload to that machine and the remaining workloads are distributed next to the selected VMs in a circular manner, this strategy is simple to implement and equally distributes the workloads but it does not check the current workloads and capacity of the VM leads certain VMs to be overloaded. The ESCE is a dynamic strategy that chooses the least workload VMs for allocations to avoid overloading of resources but does not check the previous usage of VMs leading certain VMs to be always busy while others are idle. The TLB is a dynamic strategy that selects the suitable VM with a capacity greater than the workload capacity and allocates the workload to avoid overloading of VMs. The analysis was presented with the primary metrics such as response time, datacenter processing time, and cost. These strategies are implemented in the CloudAnalyst simulation tool to assess their performance with metrics such as response time, processing time, and cost. Comparing the results, the TLB approach outperforms RR and ESCE [6].

The objective of our research is to study and examine the performance of a few popular cloud simulation tools according to their graphical support, programming model used, physical configuration, power consumption, simulation time, platform used, and elasticity support, and also implement our proposed approach in the CloudAnalyst tool to prove that, this tool is suitable for examining load balancing techniques in terms of some significant performance metrics in a dynamic IaaS heterogeneous cloud environment and implements the new metrics termed makespan in the CloudAnalyst tool.

2 Methodology

This section is divided into two phases, in phase one, we provide an outline of the various simulation tools used to examine the LB algorithms. In phase two, we have implemented our proposed load-balancing approach in CloudAnalyst.

2.1. Analysis of Prominent IaaS Cloud Simulators

There are a multitude of open-source tools available for simulating the LB strategies in an IaaS cloud model. Phase one of this paper focuses on the assessment of five prominent simulation tools, more specifically CloudSim, CloudAnalyst, CloudReport, CloudExp, and GreenCloud, for simulating IaaS cloud computing models, which are compared with parameters like structural design, components, simulation process, scalability, and performance metrics. The rationale behind selecting these specific tools is their unique application in load balancing and dynamic workload distribution among diverse resources in an IaaS cloud heterogeneous environment, as well as they are practically implemented and expanded their features when required.

2.1.1. CloudSim

This simulator is an open-source event-based simulation framework written in the Java programming language. It supports several platforms, such as Windows, Linux, and MAC operating systems, and makes it possible to simulate various cloud services, such as SaaS, PaaS, and IaaS. The main services are to offer an IaaS cloud computing virtual environment, support the provisioning of the needed resources, and give extremely accurate simulation results. The main drawback of this simulator is that it provides limited support for the Graphical User Interface (GUI). [7].

Features of CloudSim:

- The user can easily develop scenarios and extend them with little effort.
- Enables large-scale cloud application simulation.
- Using CloudSim, a user can model the virtualization environment with a varying number of datacenters and virtual machines.

Architecture of CloudSim: As shown in Figure 1, the CloudSim architecture is subdivided into two layers, such as the CloudSim layer and the User Code Layer, and each component in the layers can communicate with each other by passing messages. The user can alter the features of the stimulating environment using their own code in the User Code layer, a custom layer, in accordance with their most recent study findings. The CloudSim layer manages the datacenters, virtual machines, memory and storage requirements, bandwidth, provisioning of physical machines to virtual machines, etc. [8].

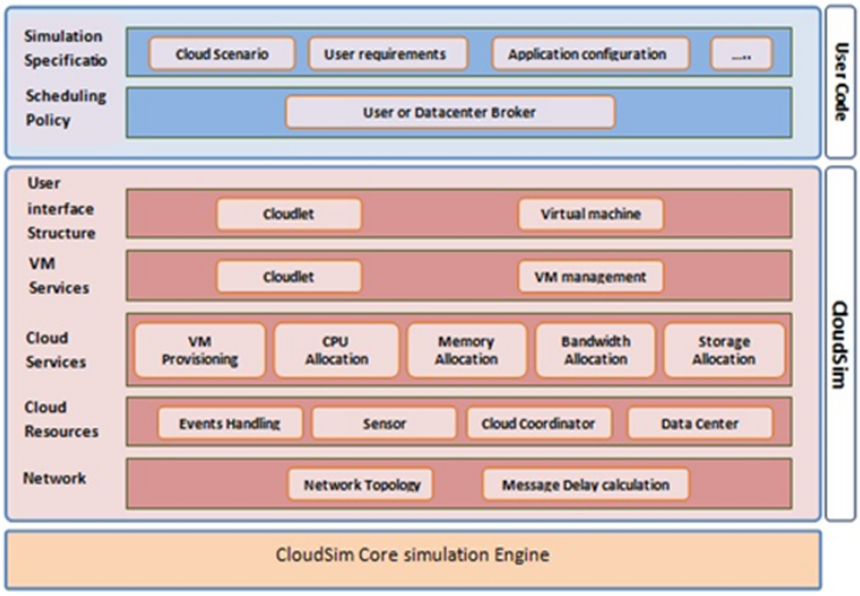

Fig 1. Overall Architecture of CloudSim tool in Cloud Computing

Datacenter: It consists of several physical hosts and it builds numerous virtual servers on top of each physical host.

Physical Host: Original physical hardware.

Virtual Machine: It processes all the incoming user's workload arriving from the UserBase.

Cloudlet: This contains the specification of the user's workload, such as size and computational requirements (number of CPUs, memory, and bandwidth).

Broker: distributes the user's workload among different cloud service providers in accordance with the quality of service (QOS).

Cloud Coordinator: coordinates all the cloud services and monitors the activities of the data center [9].

2.1.2. CloudAnalyst

CloudAnalyst is a cloud computing simulation tool for simulating large-scale cloud applications in an IaaS heterogeneous cloud environment. It is an extension of the CloudSim tool that supports examining the performance of both static and dynamic LB policies under different scenarios, supports the provision of cloud resources based on the requirements of the user, and balances the workload among various data centers and virtual machines. This tool distributes cloud resources such as virtual machines among various data centers scattered in different geographic locations based on various parameters like the geographic location of the user and the data center, network traffic based on the delay to reach the data center, capacity of the virtual machines in each datacenter, response time, processing time, task priority, etc.

Features of CloudAnalyst:

- Easy to use and most versatile
- Repeatability of experiments
- Easy to use interactive Graphical User Interface
- Results are presented in graphical format such as charts and Graphs
- High level of Scalability [10].

The main highlight is the full support of a Graphical User Interface (GUI), where a user can set up experiments easily and quickly, save their experiments in XML format, and view the results as tables and charts. Figure 2 shows the pictorial view of the CloudAnalyst simulation tool. The CloudAnalyst structure is the same as in CloudSim, with the addition of a graphical user interface. [11].

The primary parts of cloudAnalyst are the Internet, UserBase (pool of end users), Regions, Datacenter Controller, VM Load Balancer, and Service Brokers, as shown in Figure 3 [12]. There are seven core components present in the CloudAnalyst tool as follows [13].

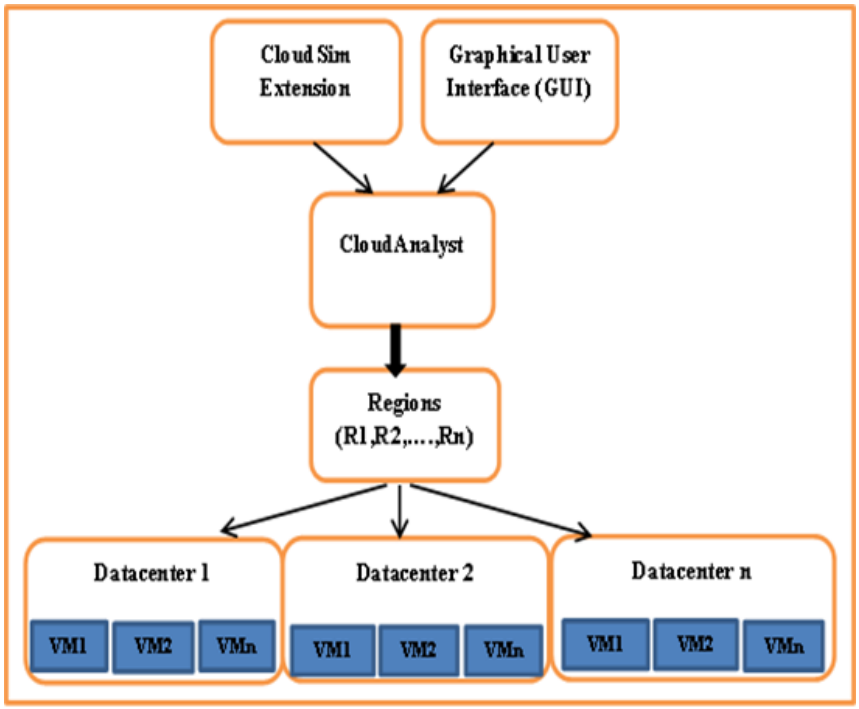

Fig 2. OverallView of CloudAnalyst Tool in Cloud Computing

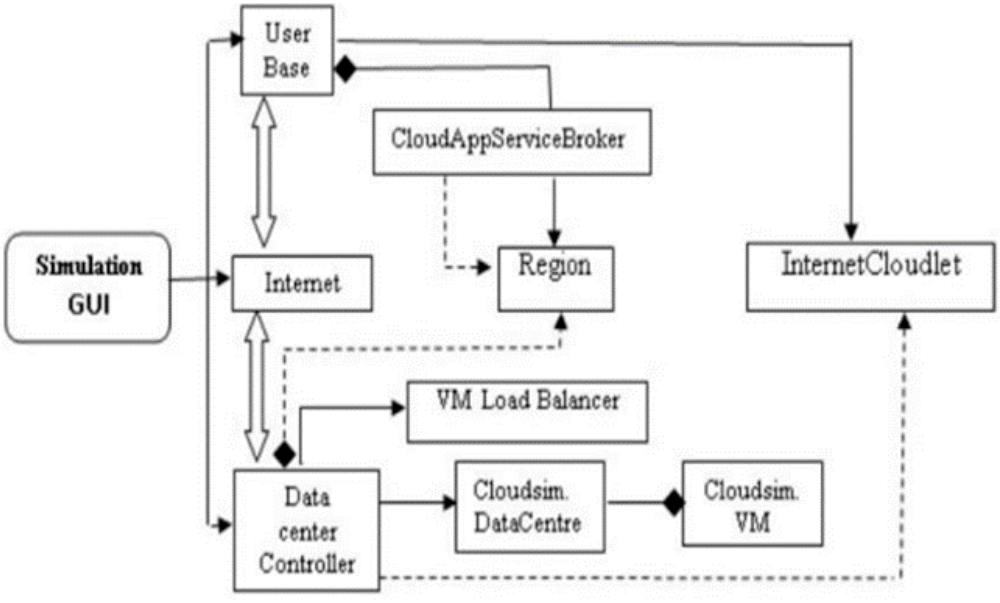

Fig 3. Main Components of CloudAnalyst Simulation Tool

Region: In CloudAnalyst, the world is geographically distributed into six regions (continents). All the UserBases and Datacenters associated with any one of these regions. User Base: Each UserBase consists of a collection of users who need to execute loads (HTTP requests).

Internet: It represents the real world internet, which routes web traffic by the UserBase to the Data Center Controller. The Internet also specifies the traffic delay among the UserBases and the Datacenters.

Datacenter Controller: It initiates the datacenter creation process, obtains the application given by the UserBase through the Internet, and routes the user's workload to the datacenter and the load balancer based on the criteria given in the load balancing algorithms.

Service Brokering: Service Brokering selects the datacenter to which the user's requests should be forwarded. On the basis of the Service Broker Policies, it selects the appropriate datacenter.

Internet Cloudlet: Each user's load has been grouped into a single Cloudlet.

VM Load Balancer: It assigns each Cloudlet to the proper virtual machine for processing the Cloudlet.

2.1.3. CloudReport

CloudReport offers a framework that simulates the cloud environment using a GUI with a large amount of data without the need for programming knowledge and makes it simpler to understand the simulation results by displaying charts and graphs showing information about resource usage cost and execution time. It provides an energy consumption model, resource utilization policies, virtual machine allocation policies, and service brokering policies.

Features of CloudReport:

- Provides an easy-to-use interface and report generation.
- Simulation data can be exported in the form of graphs and charts.
- Using APIs allows researchers to develop their own policies by making extensions to the existing rules[14].

Figure 4 shows the architecture of CloudReport which is entirely based on the CloudSim tool with a GUI environment and persistence layer to store the data permanently and provides the simulation report which shows the total time in seconds taken by each workload to finish its execution[15].

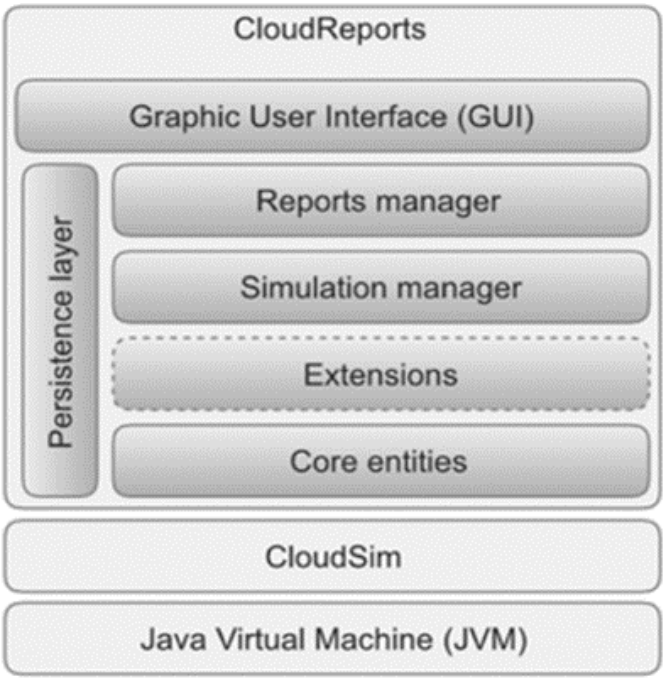

Fig 4. Architecture of CloudReport Tool

2.1.4 GreenCloud

It is an open-source packet-level simulator tool and an extension of the NS2 simulator primarily used to calculate the energy consumption of cloud datacenters and also concentrates on workload distribution and resource allocation, but the limitation is that it does not support Graphical User Interface and it can only scale to small datacenters due to its extremely long simulation time and high memory requirements and optimize the datacenter.

Features of GreenCloud:

- Focus on energy awareness and cloud networking performance.
- Support a TCP/IP reference model that enables the integration of TCP/IP with simulations.
- Protocol communication is possible [website address13] [16].

Figure 5 depicts the architecture of the Green Cloud Simulator. The construction of the Green Cloud Simulator is based on a three-tier datacenter design. The servers, which perform task execution, are the most crucial parts of a datacenter. The Green Cloud's server components are single-core nodes with a defined processing power limit expressed in FLOPS or MIPS. Access Network, Aggregation Network, and Core Network are a few of the simulator's sub modules. Workloads can be sent to any of the computing servers in time for execution thanks to the switches and cables that make up these networks' connectivity material. The Green Cloud Simulator employs an energy model of switches and links with the goal of preserving the characteristics of the DNS (Dynamic Shutdown) and DVFS (Dynamic Voltage and Frequency Scaling) techniques [17].

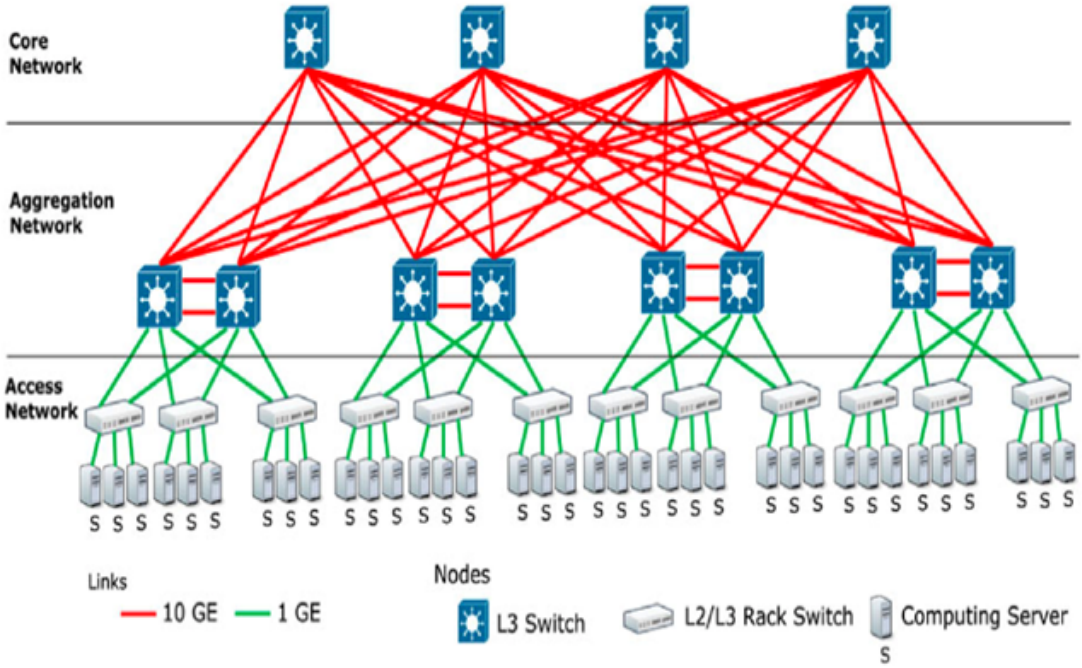

Fig 5. Architecture of GreenCloud Simulator

2.1.5. CloudExp

The CloudExp simulation tool is a unified structure for large-scale mobile cloud computing and is used mainly for demonstrating and modeling the mobile cloud computing framework. It allows users to build their own user-friendly mobile cloud infrastructure with GUI features and experiment with various mobility cases for mobile devices. The GUI aids in constructing the various modules of the cloud, like datacenters, processing components, storage, network connections, etc. The scenario results are exhibited in the form of an Excel sheet, which can be easily examined and inferred. CloudExp also has a MapReduce model to handle scenarios correlated to big data and to produce real physical workloads in simulated settings [18].

2.1.6. Performance Assessment of different Cloud Simulator

This section compares five different simulators: CloudSim, CloudAnalyst, CloudReports, CloudExp, and GreenCloud, according to a variety of criteria, including development platforms, programming languages, availability, graphical support, cost models, adoption of migration, scalability, energy consumption models, simulation time, and support of graphical output, as shown in Table 1 [19] [20].

Table 1. Parameter Comparison of different Cloud based Simulator

Parameters	CloudSim	CloudAnalyst	GreenCloud	CloudReport	CloudExp
Availability	Open Source	Open Source	Open Source	Open Source	Proprietary
Simulation Time	Minutes	Seconds	Minutes	Seconds	Minutes
Speed	Low	Fast	Low	Fast	Slow
Programming Language	Java	Java	C++/OTel	Java	C++
Graphical User Interface Support	No	Yes. Full	Limited	Yes. Full	Yes. Full
Platform used	SimJava	CloudSim/SimJava	NS2	CloudSim	CloudSim
Migration Support	Limited	Full	Full	No	Limited
Load Balancing	Yes	Yes	Yes	No	
Extension	Not Possible	Possible	Limited to only small data-centers	Not Possible	Possible
Graphical Output	Partial	Yes	Partial (HTML report with graphics)	Yes	No
Energy Model	No	Yes	Precise	Yes	Yes
Cost Model	Yes	Yes	No	Yes	Yes
Load Balancing	Yes	Yes	Yes	No	Yes

2.2. The TALB Strategy

The Throttled Adapted Load Balancing (TALB) model that has been proposed in our previous research is used for the distribution of all incoming user workloads to multiple virtual machines (VMs) in the IaaS cloud platform, It enhances the current TLB algorithm by changing the data structure from the HashMap to the TreeMap structure, with the key objective of reducing the time spent searching for a suitable VM and increase the makespan value. Makespan refers to the total time required for the resources to complete processing of set of workloads.

In order to store virtual machine (VM) details efficiently as key-value pairs in sorted form, it creates a red-black tree structure called a TreeMap. This structure allows for multiple partitions to hold varying levels of VM, such as the busy VM List and the available VM List, to keep the busy and available VMs in separate partitions. This process makes it easier to quickly identify the appropriate VM. When scheduling loads, this facilitates and speeds up the process of identifying the available and busy virtual machines.

TALB also assigns multiple loads to each VM, whereas TLB assigns a single load to each VM simultaneously, preventing overloading or underloading of VMs and improving the IaaS cloud's performance [21].

To assign loads to a VM, a few prerequisites must be satisfied, including determining which VMs are appropriate for the load allocation and depending on the VM's size, availability, and current allocations. As soon as the load arrives, TALB examines the available TreeMap partition, and if the VM is suitable for that specific load, the load is assigned to it. After the load is assigned to that VM, it is moved to the list of busy VMs once the load has been assigned to it, and its remaining capacity is computed. If the required VM is not in the available list, then the TALB searches the busy list for acceptable virtual machines (VMs) based on load size and VM size. If it is not on this busy list, the load should wait until it obtains the correct VM.

2.2.1. TALB Workflow Diagram

Figure 6 depicts the pictorial representation of the procedure of the TALB strategy: loads are forwarded to the DCC from the userbases, which in turn arranges the load in the load queue. The TALB selects the proper VM and distributes the load. The TALB chooses the appropriate VM based on its size and present allocations, and allocates the load to that VM. If the appropriate VM is not available, then TALB makes the loads wait until it finds the proper one.

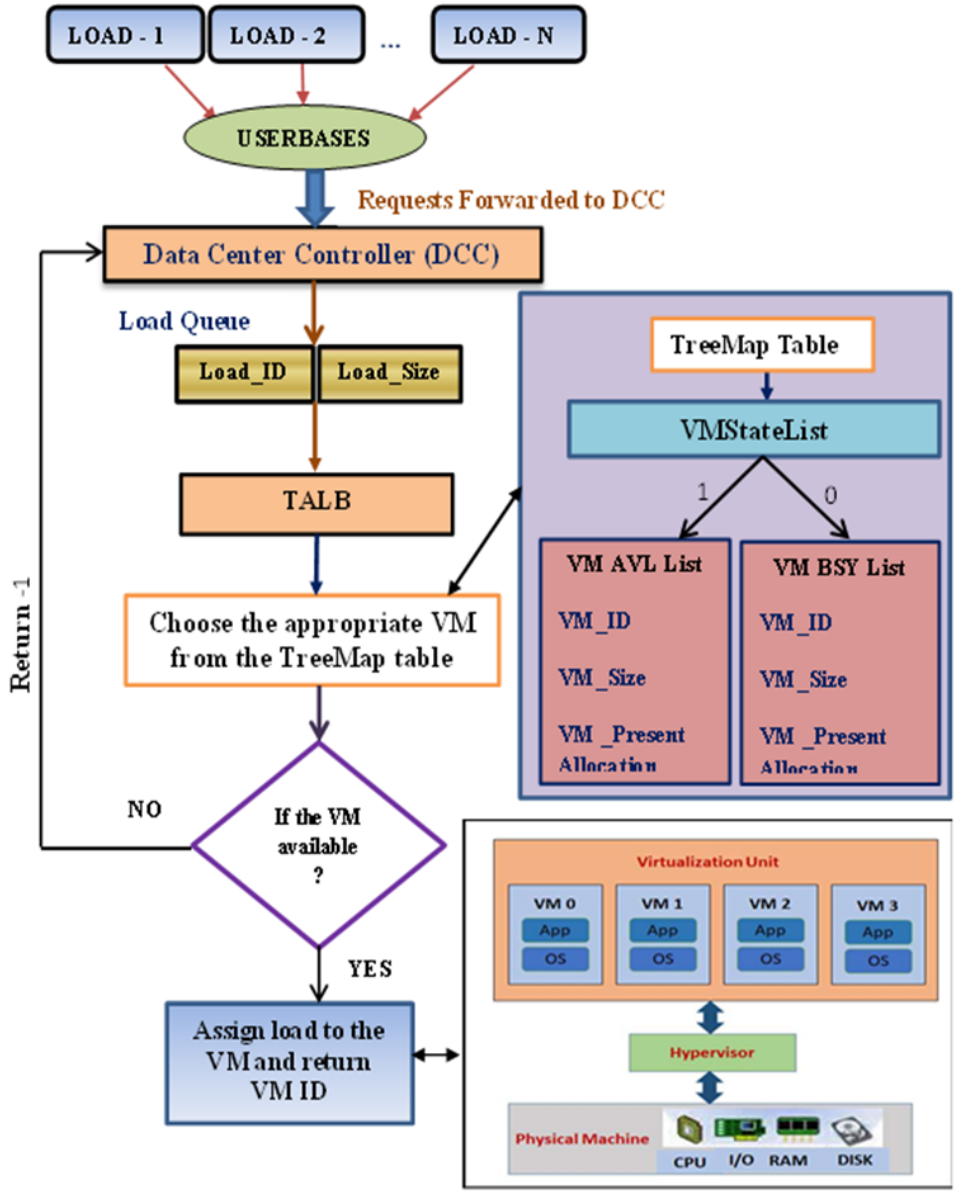

Fig 6. Workflow Diagram of TALB

2.2.2. TALB Procedure

Input: Number of loads submitted. L1, L2, L3,...Ln.

Total VMs: VM1, VM2, VM3, VMn.

VM state = VMs (i) € {Busy, Available), where i is the VM ID and s is the state of each VM.

VMs (i) = Busy (0), If any load Li is assigned to that VM,

VMs (i) = Available (1), then. The VM is idle, and no load has been assigned yet.

I. A Datacenter Controller (DCC) receives all the loads arriving from various UserBases (UB), and creates the Cloudlets for each load and each Cloudlet contains the name of the load, size, UB, region, and DC, keeping them in the ready queue based on the order in which the load arrived from the UB.

II. The amount of RAM capacity and CPU power needed for each Cloudlet is determined by calculating its size based on the amount of RAM, CPU capacity, and bandwidth requirements.

III. Unique VM ID is assigned to all the VMs present and determines each VM's size by accumulating the capacity of each virtual processor (VCPU), amount of memory and storage capacity and the available bandwidth present in that VM.

V. Submits the loads to the TALB load balancer for VM allocation

V. The TreeMap structure holds all the VM details. A unique node is created in the TreeMap for every VM, which is composed of three fields: VM ID, size, and its status information such as Available (1) or Busy (0). Initially, all VMs are given the status field value 1 i.e., all VMs are available.

VI. The TreeMap structure is subdivided into two separate partitions: the available partition (AvailVM) and the busy partition (BusyVM), where the AvailVM holds all the VMs that have no loads assigned and the BusyVM has all the VMs that have at least one load or some loads already assigned to them.

VII. At first, each node in the AvailVM is arranged in a sorted order based on the key (VM size). That is, the highest-capacity VM is listed first in the list, followed by the lowest-capacity VMs.

VIII. Each node in the BusyVM is arranged in a sorted order based on multiple keys (VM size and VM present number of loads). That is, the highest capacity VM with the lowest loads is listed first, and then the lowest capacity with the highest loads.

IX. The TALB Load Balancer looks over the AvailVM and selects the VM based on its size (capacity); that is, the VM's size should exceed the job's size. If the VM is available, the load is then assigned to that VM, and the VM ID is returned to the DCC.

X. After a VM is assigned a load, the VM status is changed to Busy (0), and it is moved to the BsyVMList, where its remaining size is updated to accommodate additional load allocation. Now the BsyVMList updates the position of all of the VMs based on the current load and size.

XI. The specific VM is transferred to the AvlVMList partition of the TreeMap, and its status field is modified to Available (1) if it completes all its loads.

XII. If there is no available VM and all VMs are busy, meaning that no VMs are available for allocation, TALB searches the busy list for the appropriate VM based on the VM's current capacity and load size. If VM_{rc} >L_{is}, where VM_{rc} represents the VM's remaining capacity and L_{is} is the size of the i^{th}load if a suitable VM is found, then the load is assigned; if there are no available VMs for a load, then it must wait until it obtains the proper VM. In this case, it will return -1 to the DCC.

XIII. The TALB alerts the DCC to the availability of the appropriate VM as soon as the loads call for a VM. Depending on the size of each available VM, the TALB distributes the remaining loads to them.

XIV. If additional loads are received on the UB, go back to step 2 and continue the process until all of the loads have been assigned.

RT can be determined by Equation (1)

$$RT = FT - AT + TD \tag{1}$$

Whereas AT stands for Arrival Time, TD for Transmission Delay, RT stands for Response Time of every load, and FT for loads Finishing Time.

Equation (2) yields the DCPT value.

$$DC_{pt} = DC_{rt} - (2 * DC_{np}) \tag{2}$$

Where DC_rt denotes the response time and DC_np denotes the network transfer latency.

3 Results and Discussion

The result section evaluates the proposed TALB approach performance relating to the existing load balancing approaches. As performance measures, we have used the Searching Time (ST) of each virtual machine (VM), the Response Time (RT) of each user load, and the Data Center Processing Time (DCPT) and the makespan value.

3.1. Implementation of the proposed Load Balancing Approach in CloudAnalyst

In this experiment, we have created and evaluated the TALB approach with three distinct LB techniques: Equally Spread Current Execution (ESCE), Throttled LB (TLB), and Round Robin (RR), using the CloudAnalyst tool. The primary performance parameters that are crucial to take into account in this situation are searching time, response time, datacenter processing time and makespan value. The proposed TALB approach is implemented in the Cloudsim.ext.Datacenter package in the CloudAnalyst module, as shown in Figure 7.

The LB techniques have been investigated in this simulation using five data centers, sixty virtual machines, and ten user bases, sixteen thousand user loads in each user bases. Table 2 displays the outcomes of the simulation of the RR, ESCE, TLB and TALB techniques with respect to Response time, processing time, searching time and makespan value.

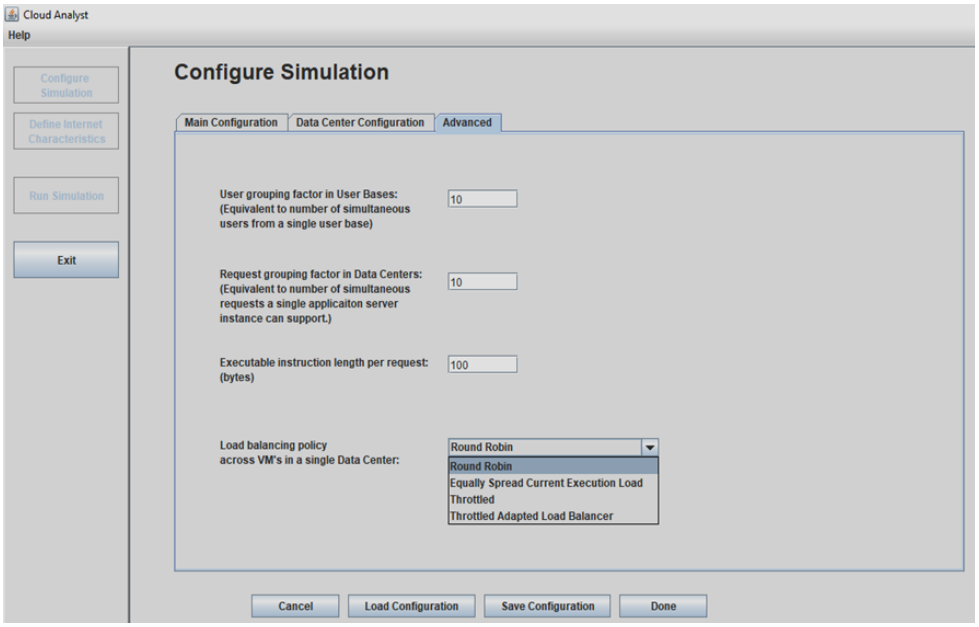

Fig 7. Deployment Simulated Panel for the CloudAnalyst Tool Kit

Table 2. Simulation Results of different Load Balancing Approaches

Load Balancing /Performance Metrics	Response Time (Milliseconds)			Datacenter (DC) Processing Time (Milliseconds)			Overall Average Searching Time	Overall Makespan Value
	Avg	Min	Max	Avg	Min	Max		
RR	299.64	95.43	995.33	340.33	48.33	482.21	340.45	313.73
ESCE	297.94	93.42	974.62	318.89	48.24	482.21	360.65	285.14
TLB	289.95	90.52	940.61	296.65	47.32	489.15	390.02	269.12
TALB	270.91	86.52	899.33	212.74	37.12	470.22	127.21	192.89

Figure 8 display the graph for the simulation results, the searching time, data center processing time, average response time and makespan value for each method are displayed. It can be shown that the TALB strategy has lower searching time, response times and datacenter processing time, than RR, ECSE and TLB and also implements new parameter such as makespan. The results in the graph shown that TALB yields minimum makespan values than RR, ESCE and TLB.

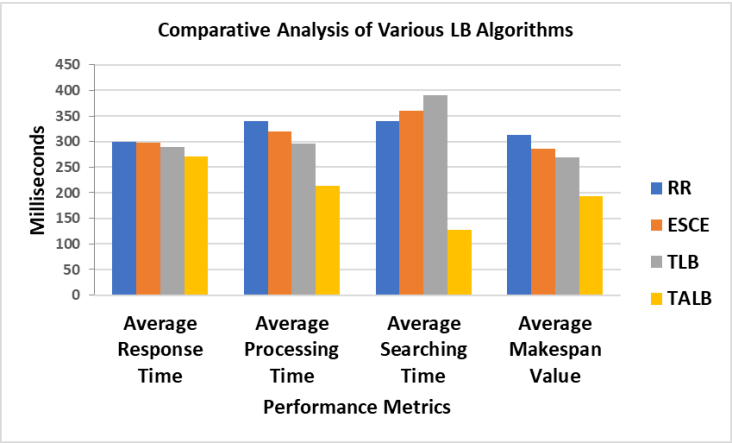

Fig 8. Average Searching, Response and Datacenter Processing Time and Makespan of RR, ESCE, TLB and TALB

While comparing the results of the TALB strategy with the previous results of RR, ESCE and TLB in the literature, the proposed TALB strategy yields the overall average searching time of 127.21 milliseconds which is 47.21 % faster than RR, 44.53 % faster than ESCE and 54.65 % faster than TLB. According to response time, the TALB gives 270.91 milliseconds which is 21.34 % faster than RR, 20.23 % faster than ESCE and 19.98 % faster than TLB. According to the datacenter processing time, TALB reduces into 212.74 which is 26 % faster than RR, 23 % faster than ESCE and 21 % faster than TLB. Hence, the TALB approach presented in this research yields optimum results while comparing with the previous results [(6)] [(21)].

4 Conclusion

Load balancing is an emerging technique for the uniform distribution of various computing resources to end users in a cloud environment. The implementation and utilization of cloud resources grow exponentially, it is imperative to evaluate the efficacy and performance of load-balancing strategies in cloud systems. One of the biggest challenges is choosing the proper simulation tool for load balancing depending on the nature of the resources, size of the workloads and most importantly the performance metrics used and the resultant parameters. Several cloud simulators have been particularly developed for the performance evaluation of load balancing strategies in cloud computing environments. The objective of this research, was to discuss a few well-known cloud simulators, such as CloudSim, GreenCloud, CloudExp, CloudAnalyst, and CloudReport and compared based on the important performance metrics and also revealed that which is suitable for the particular scenarios by implementing load balancing algorithms in CloudAnalyst tool by incorporating the new metric such as makespan. The novelty of this research is that, after the comparison of the simulators, CloudSim is the most effective tool for analyzing complex simulations, while CloudAnalyst is best suited for large-scale cloud performance testing. It is also the best option when calculating response times, resource processing times, costs, resource migration, network proximity, and social media platform usage. CloudReport is the ideal option when it comes to measuring resource utilization and implementation costs. The greatest choice for datacenter efficiency and energy modeling is GreenCloud. Applications in mobile cloud computing are processed using the CloudExp tool. It may be inferred from the comparisons that the choice of cloud simulation tool depends upon the goals and requirements of the researcher. Here, the proposed TALB policy is implemented and obtains the minimum RT which is 22.74% lower than that of RR, ESCE, and TLB and minimal DCPT value which is 24% faster than RR, ESCE, and TLB. TALB fasten the ST which is 48.23% faster than RR, 47.12% faster than ESCE, and 53.64% faster than TLB and also produces low makespan value of 23.85% than RR, 19.29 % than ESCE and 16.5 % than TLB algorithm.

Concerning Future work, In order to get good results, we want to model other techniques in the future and use these simulators to carry out different tests. The plan is designed to analyse and make use of the findings, and it includes fresh ideas and suggestions for putting green cloud computing into practice.

References

1) Omurgonulsen M, Ibis M, Kazancoglu Y, Singla P. Cloud computing: a systematic literature review and future agenda. *Journal of Global Information Management*;1(6):1–25. Available from: https://doi.org/10.4018/jgim.20211101.oa40.
2) Islam R, Patamsetti V, Gadhi A, Gondu RM, Bandaru CM, Kesani SC, et al. The Future of Cloud Computing: Benefits and Challenges. *International Journal of Communications*. 2023;16(4):53–65. Available from: https://doi.org/10.4236/ijcns.2023.164004.
3) Sefati S, Mousavinasab M, Farkhady ZR. Load balancing in cloud computing environment using the Grey wolf optimization algorithm based on the reliability: performance evaluation. . *The Journal of Supercomputing*. 2022;78:18–42. Available from: https://doi.org/10.1007/s11227-021-03810-8.
4) Sahkhar L, Balabantaray BK. Scheduling Cloudlets to improve response time using CloudSim Simulator. *Lecture notes in networks and systems 2021*;p. 483–93. Available from: https://doi.org/10.1007/978-981-33-4084-8_46.
5) Singh H, Tyagi S, Kumar P. Comparative analysis of various simulation tools used in a cloud environment for task-resource mapping. In: Proceedings of the International Conference on Paradigms of Computing, Communication and Data Sciences. Springer. 2020;p. 419–430. Available from: https://doi.org/10.1007/978-981-15-7533-4_32.
6) Priya N, Priya SS. An experimental evaluation of load balancing policies using cloud Analyst. *Lecture notes in networks and systems 2022*;p. 185–98. Available from: https://doi.org/10.1007/978-981-16-7657-4_16.
7) Sahkhar L, Balabantaray BK. Scheduling cloudlets to improve response time using cloudsim simulator. In: Proceedings of the International Conference on Computing and Communication Systems: I3CS 2020. Springer. 2021;p. 483–493. Available from: https://doi.org/10.1007/978-981-33-4084-8_46.
8) Sundas A, Panda SN. An introduction of CloudSim simulation tool for modelling and scheduling. In: and others, editor. Proceedings of the 2020 international conference on emerging smart computing and informatics. ESCI. 2020;p. 263–268. Available from: https://doi.org/10.1109/esci48226.2020.9167549.
9) Bala K, Kumar S, Rani A. Cloud Computing Environment Modelling and simulation as well as resource provisioning algorithm assessment. *International Journal of Mechanical Engineering*. 2021;6(0001). Available from: https://doi.org/10.56452/2021sp-8-010.
10) Qarqur H, Sah M. imulation of algorithms and techniques for green cloud computing using CloudAnalyst. *Learning and analytics in intelligent systems*. 2021;p. 399–407. Available from: https://doi.org/10.1007/978-3-030-65407-8_34.
11) Ahmad AY, Hammo AY. A Comparative Study of the Performance of Load Balancing Algorithms Using Cloud Analyst. *Webology*. 2022;19(1):4898–911. Available from: https://doi.org/10.14704/web/v19i1/web19328.

12) Jena SR, Shanmugam R, Saini K, Kumar S. Cloud computing tools: inside views and analysis. . *Procedia Computer Science*. 2020;173:382–91. Available from: https://doi.org/10.1016/j.procs.2020.06.045.
13) Chafi SE, Balboul Y, Mazer S, Fattah M, Bekkali ME, Bernoussi B. Cloud computing services, models and simulation tools. *International Journal of Cloud Computing*. 2021;10(5/6):533–533. Available from: https://doi.org/10.1504/ijcc.2021.120392.
14) Mehta A, Panda SN. CloudReports tool to implement IaaS framework with location-based authentication in cloud. In: Communication and Computing Systems: Proceedings of the 2nd International Conference on Communication and Computing Systems (ICCCS 2018). CRC Press. 2018;p. 388–388. Available from: https://doi.org/10.1201/9780429444272-62.
15) Shahid MA. A Systematic Survey of Simulation Tools for Cloud and Mobile Cloud Computing Paradigm. *Journal of Independent Studies and Research Computing*. 2022;20(1). Available from: https://doi.org/10.31645/JISRC.22.20.1.10.
16) Qarqur H, Sah M. Simulation of algorithms and techniques for green cloud computing using CloudAnalyst," . *Learning and Analytics in Intelligent Systems*. 2021;p. 399–407. Available from: https://doi.org/10.1007/978-3-030-65407-8_34.
17) Rajpoot NK, Singh P, Pant B. Load Balancing in Cloud Computing: A Simulation-Based Evaluation. In: 2023 International Conference on Computational Intelligence and Sustainable Engineering Solutions (CISES). 2023;p. 564–568. Available from: https://doi.org/10.1109/CISES58720.2023.10183622.
18) Sharmah D, Bora KC, Noorain M. Utilizing dynamic load balancing to improve private cloud paradigm. *Int j inf tecnol*. 2024. Available from: https://doi.org/10.1007/s41870-024-01888-w.
19) Dubey S, Singhal V, Sahu AK. Simulation of Scheduling & Load Balancing Algorithms for a Distributed Data Center by Using Service Broker Policy Over the Cloud. In: and others, editor. Heterogenous Computational Intelligence in Internet of Things. CRC Press. ;p. 131–180. Available from: https://doi.org/10.1201/9781003363606-9.
20) Chafi SE, Balboul Y, Mazer S, Fattah M, Bekkali ME, Bernoussi B. Cloud computing services, models and simulation tools. *International Journal of Cloud Computing*. 2021;10(5-6):533–580. Available from: https://doi.org/10.1504/ijcc.2021.10044185.
21) 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.