ENERGY AWARE LOADBALANCING
in Cloud Computing using CloudSim Simulator

A project work submitted in fulfillment of requirements
for the degree of
BACHELOR OF TECHNOLOGY

in
COMPUTER SCIENCE AND ENGINEERING

By

CHILUKALAPALLI SHIVA PRASANTH (Roll No. 10700120004)
&
MOHAMMED ADEEB (Roll No. 10700120048)
&
ROHIT SHARMA (Roll No. 10700120080)

Under the super vision of
Prof. Lipika Dutta

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

COLLEGE OF ENGINEERING AND MANAGEMENT, KOLAGHAT
(Affiliated to MAKAUT, WB)
Purba Medinipur-721171, West Bengal, India
June' 2024

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

COLLEGE OF ENGINEERING AND MANAGEMENT, KOLAGHAT
(Affiliated to MAKAUT, WB)
Purba Medinipur-721171, West Bengal, India

CERTIFICATE OF APPROVAL

This is to certify that the work embodied in this project entitled **Energy-aware Load Balancing in Cloud Computing** submitted by Chilukalapalli Shiva Prasanth, Mohammad Adeeb and Rohit Sharma, to the Department of Computer Science & Engineering, is carried out under my direct supervision and guidance.

The project work has been prepared as per the regulations of Maulana Abul Kalam Azad University of Technology and I strongly recommend that this project work be accepted in partial fulfilment of the requirement for the degree of B.Tech.

Supervisor
Prof. Lipika Dutta

Countersigned by

Prof. (Dr.) Alok Ranjan Pal
Head of the department of
Computer Science and Engineering

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

COLLEGE OF ENGINEERING AND MANAGEMENT, KOLAGHAT
(Affiliated to MAKAUT, WB)
Purba Medinipur-721171, West Bengal, India

CERTIFICATE BY BOARD MEMBERS

This is to certify that the project work entitled by **Energy-aware Load Balancing in Cloud Computing** submitted by Chilukalapalli Shiva Prasanth, Mohammad Adeeb, Rohit Sharma, to the Department of Computer Science and Engineering of the College of Engineering and Management, Kolaghat has been examined and evaluated .

The project work has been prepared as per the regulations of the Maulana Abul Kalam Azad University of Technology and qualifies to be accepted in partial fulfilment of the required for the degree of Bachelor of Technology(*B.Tech*).

Project Coordinator **Board of Examiners**

ACKNOWLEDGEMENT

We take this opportunity to express our profound and sincere gratitude to our project guide, Prof. Lipika Dutta. We respectfully acknowledge her constant encouragement, attentive guidance and keen interest throughout the duration of the work. We are deeply indebted to her for her invaluable tips and suggestions that have significantly shaped and structured our object.

We are also extremely grateful to the Department of Computer Science and Engineering at College of engineering and Management, Kolaghat for providing us with all the necessary departmental facilities.

We extend our heartfelt thanks to the other faculty members, laboratory staff, library staff and administration of this institute for their valuable assistance, time, and for fostering a conductive work environment.

Finally, we wish to express our deepest thanks to our classmates, whose timely suggestions and support have greatly contributed to the success of the project.

Date : 06th June' 2024

Name: Chilukalapalli Shiva Prasanth
University Roll No. : 10700120004
University Registration No. : 201070100110123

Name: Mohammad Adeeb
University Roll No. : 10700120048
University Registration No.:201070100110079

Name: Rohit Sharma
University Roll No. : 10700120080
University Registration No.:201070100110047

ABSTRACT

This project investigates the strategic application of Virtual Machine (VM) Migration based on resource utilization to achieve Load Balancing and optimise energy consumption in Cloud computing environments. By setting predefined thresholds for resource utilization metrics such as CPU and memory, the study aims to dynamically redistribute VMs among servers. To enhance this methodology, a Machine Learning algorithm, specifically Linear Regression, was utilized to analyse historical datasets and accurately determine the optimal threshold values that minimize energy consumption. The primary objective is to maintain a balanced workload across the cloud infrastructure, ensuring efficient resource utilization while reducing energy consumption. Experimental results demonstrate the effectiveness of this approach in achieving Load Balancing; however, the observed increase in execution time with more frequent migrations highlights the need for a nuanced consideration of trade-offs. These findings offer valuable insights into the delicate balance between Load Balancing, VM migration, and energy efficiency, providing a foundation for refining strategies to enhance the performance and sustainability of cloud computing environments.

CONTENTS

1. OBJECTIVE

The primary objective of this project is to enhance load balancing and optimize energy consumption within cloud computing environments through the strategic application of Virtual Machine (VM) migration. This involves setting predefined thresholds for resource utilization metrics such as CPU and memory to dynamically redistribute VMs among servers. Additionally, the project aims to refine this methodology by employing a Machine Learning algorithm, specifically Linear Regression, to analyse historical datasets and determine optimal threshold values. The ultimate goal is to maintain a balanced workload across the cloud infrastructure, ensuring efficient resource utilization while minimizing energy consumption, thereby improving overall system performance and sustainability.

2. INTRODUCTION

Cloud Computing has emerged as the transformative paradigm in information technology, redefining how resources are accessed, managed, and utilized. This report delves into the crucial aspect of load balancing within cloud environments, with a specific focus on leveraging the cloudSim simulator for comprehensive analysis and simulation.

As organisations increasingly migrate to cloud infrastructures, the dynamic nature of workloads poses significant challenges in resource allocation and optimization. Load balancing becomes paramount in ensuring equitable distribution of computational tasks, thereby maximising efficiency and minimising response times. This report navigates through the fundamental components of cloud computing, introduces the cloudSim simulator, and elucidates the significance of load balancing in addressing the evolving demands of cloud-based applications.[1]

Through a structured exploration of CloudSim's installation process, architecture, and main components, this report establishes the groundwork for a detailed examination of balancing models. The ensuing sections delves into the necessity of load balancing in context of computing, acknowledging the complexities arising from varying workloads and the needs for adaptive and efficient resource management.

Furthermore, the report ventures into the realm of migration, specifically focusing on virtual machine migration and its live counterpart. It explores the impact migration of the broader spectrum of green cloud computing, recognizing the environmental implications of dynamic resource allocation. By incorporating a Machine Learning algorithm --- Linear Regression --- to analyse historical datasets, the study identifies optimal threshold values for energy-efficient resource management, thereby enhancing traditional load balancing approaches.

The practical implementation of virtual machine migration within the CloudSim framework serves as a bridge between theoretical concepts and real-world application. The subsequent sections critically analyse results, providing insights into the effectiveness of load balancing, examining scenarios with user-input, and scrutinizing resource utilization to identify instances of both over-utilization and under-utilizations.

In conclusion, this report synthesizes the findings, emphasizing the pivotal role of load balancing in optimizing cloud resources. Additionally, it sets the stage for future exploration by proposing energy-aware scheduling using CloudSim, signalling the ongoing evolution of strategies for resource management in cloud computing environments.

2.a. OVERVIEW OF CLOUD-SIMULATOR (CloudSim)

CloudSim is a sophisticated simulation tool developed at the CLOUDS Laboratory at the University of Melbourne, designed specifically for cloud computing environments. It enables users to evaluate specific system issues without delving into the low-level details associated with cloud-based infrastructures and services [2]. This capability allows for seamless modelling, simulation, and implementation in the design of cloud computing frameworks. CloudSim serves as an autonomous platform that supports large-scale cloud environments, encompassing users, applications, data centres, brokers, scheduling, and provisioning policies.

As an open-source software, CloudSim provides users with the ability to test applications in a controlled and repeatable environment, identify system bottlenecks without requiring access to real cloud infrastructures, and experiment with various configurations to develop adaptive provisioning techniques.

Key reasons for adopting CloudSim for modelling and simulation include:

- Support for diverse cloud computing data centres.
- Virtualization of server hosts, including customized policies for provisioning host resources to virtual machines (VMs).
- Specialized software containers.
- Power-aware computational resources.
- A variety of data center network topologies and message-passing applications.
- Dynamic addition or removal of simulation components.
- The ability to pause and resume simulations.
- User-defined policies for the allocation of hosts to VMs and the allocation of host resources to VMs.

2.a.i. Installation

Installation process of CloudSim can be broadly classified into **4-steps.**

i. Java Installation.
ii. Download CloudSim and Common-math-jar file.
iii. Eclipse IDE installation.
iv. Run CloudSim setup in Eclipse.

Java Installation

- Check Java in your system.
- If Java is not installed then download Java.
- Install Java.
- Set path variable for java

Download CloudSim and additional jar file.

- Download CloudSim 3.0.3
- Download Common-math-jar file.

Eclipse IDE installation

- For Java 64-bit you need to install 64-but eclipse.
- For Java 32-bit, you need to install 32-bit eclipse.
- Install Eclipse IDE.

Run CloudSim in Eclipse

- Put common-math-3.6.1 jar file into jar folder of CloudSim 3.0.3

2.a.ii. CloudSim Architecture

The structure of CloudSim simulator, as illustrated in Fig 1.1, is organized into three layers:

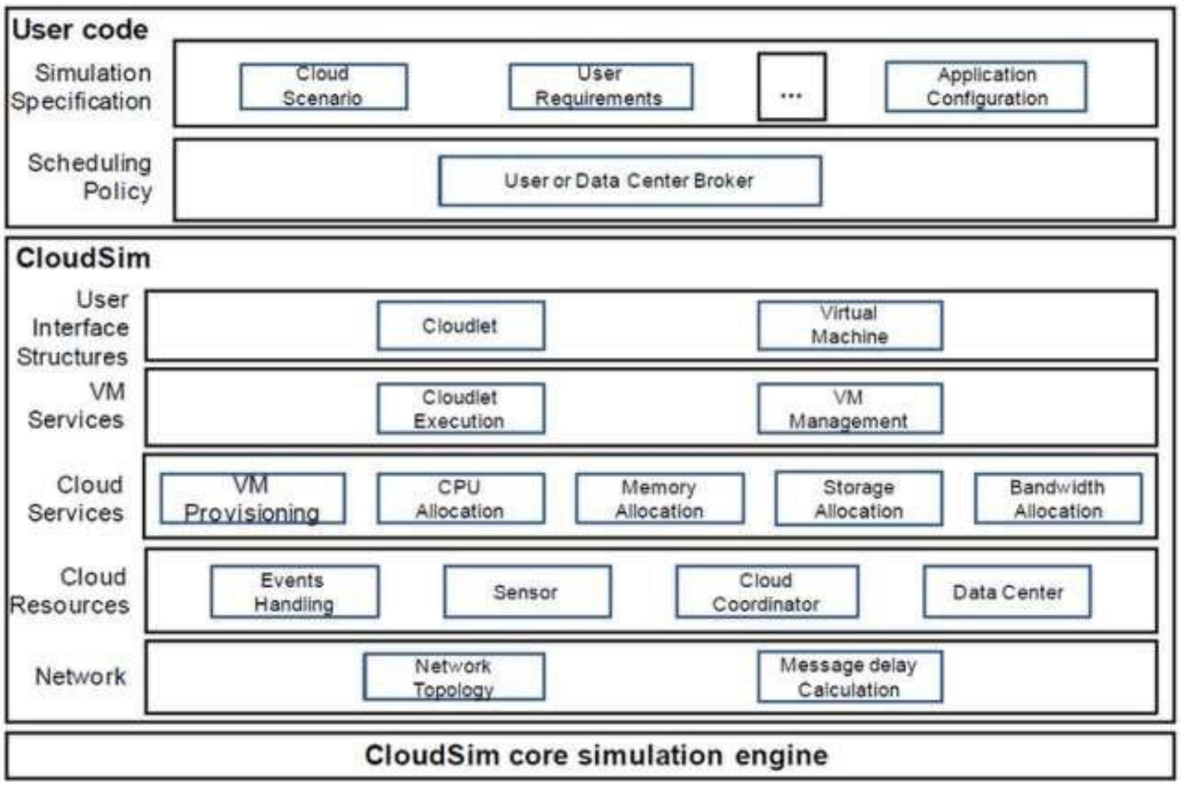


Fig 1.1

CloudSim Core Layer:

This layer is responsible for managing events and creating CloudSim objects such as hosts, virtual machines, data centres, and brokers.

CloudSim Simulation Layer:

i. Handles the modelling and simulation of cloud-based centres.
ii. Includes dedicated interfaces for resource allocation, covering CPU, RAM memory, storage, and network bandwidth.
iii. Manages the implementation of applications and monitors system status.
iv. Comprises five hierarchical layers:

   a. **Network:** Deals with network topology and determines network delays.
   b. **Cloud Resources:** Models infrastructure-level services (Datacentre*),* monitors resource states, and performs load balancing (*Cloud Coordinator)*.
   c. **Cloud Services:** Provides hosts with virtual machines, allowing for resource allocation (CPU*, memory, bandwidth)* customisation by developers.
   d. **VM Services:** Manages virtual machines and cloudlets
   e. **User Interface Structures:** Offers an interface for configured virtual machines and tasks (Cloudlets*)*

User Code Layer:

i. Empowers developers to modify parameters of key CloudSim objects, such as hosts, virtual machines, users, and resource planning policies.
ii. Allows users to generate new application and configurations, conduct tests in a cloud-based environment with custom settings, and compare and evaluate resource allocation techniques for clouds and cloud federations.

Within the CloudSim simulation layer, Infrastructure as a Service (IaaS) services are simulated by the Datacentre objects. The object oversees hosts, which are assigned to one or more virtual machines (VM) based on resource allocation policies. In CloudSim, object represents components, such as classes for the CloudSim model (data centre, hosts). A host, as a CloudSim component, represents a physical server (computing models) in a cloud, implementing interfaces for simulated distributed systems.[3]

Resource allocation, specifically VM allocation, involves creating VM instances on hosts that meet the characteristics and configuration requirements of cloud service providers. The component responsible for resource allocation VmAllocationPolicy, with the default allocation method being First-Come, First-Serve (FCFS).

2.a.iii. Main Components

Datacentre:

The datacentre class is utilized to model the essential hardware components of any cloud environment. It provides methods to define the functional requirements of the Datacentre and set allocation policies for virtual machines.

Hosts:

The host class carries out actions related to management of virtual machines, establishing policies for provisioning memory and bandwidth to VMs, and allocating CPU cores to them.

Virtual Machine (VM):

The VM class represents a virtual machine, offering data members that define parameters such as bandwidth, RAM, MIPS (million instruction per second) and size. It also provides setter and getter methods for these parameters.

DatacentreBroker:

The DatacentreBroker serves as an entity acting on behalf of the user or customer. It is responsible of VM-related functions, including creation, management, destruction and the submission of the cloudlets to VMs.

Cloudlet:

The Cloudlet class represents tasks run on a VM, such as processing, memory access, or file updating tasks. It stores parameters defining task characteristics and provides methods similar to the VM class. Additionally, it offers methods that define a task's execution time, status, cost, and history.

CloudletScheduler:

This component determines how the available CPU resources of a virtual machine are allocated among cloudlets. It offers two types of policies: Space Shared and Time Shared.

CloudletSchedulerSpaceShared:

This class implements a scheduling policy for virtual machines to execute one cloudlet at a time in a space-shared environment. It resembles batch processing, where cloudlets share the same queue and are processed one at a time per computing core.

CloudletSchedulerTimeShared:

This class implements a cloudlet scheduling policy for virtual machines to execute multiple cloudlets in a time-shared environment. Multiple cloudlets are submitted to the virtual machine, each getting its specified share of time. This influences the completion time of cloudlets in CloudSim.

VMScheduler:

The VmScheduler is an abstract class defining and implementing the policy for sharing processing power among virtual machines on a specified host. It offers two types of policies: VmSchedulerTimeShared and VmSchedulerSpaceShared.

VmSchedulerTimeShared:

This class implements the VM scheduling policy that allocates one or more processing elements to a single virtual machine, allowing the sharing of processing elements by multiple VMs with a specified time slice.

2.b. LOAD BALANCING IN CLOUDSIM

Load balancing is critical method for optimising the utilization of virtual machines (VM) resources in cloud computing environments [4]. It ensures the equitable and dynamic distribution of workloads, leading to efficient resource utilization. Effective load balancing enhances user satisfaction and improves resource utilization. Effective load balancing enhances user satisfaction and improves resource allocation. In cloud systems, implementing load balancing reduces data transmission delays (Kaur and Luthra, 2014) and prevents node overloads that can negatively impact the Quality of Service (QoS) in cloud data centres. Therefore, addressing load balancing issues and enhancing the performance of cloud-based applications is crucial, as discussed in more detail in this section.

2.b.i. Load Balancing Model

The depicted model outlines the operational flow of a load balancer in cloud computing, emphasizing the equitable distribution of user requests to selected data centres based on resource availability. It is crucial to prevent both overloading and underloading of servers (VMs) to maintain an even distribution, highlighting the importance of load balancing. An effective load balancer is essential for sustaining the performance of cloud applications, as unequal load distribution can result from various factors, including task scheduling. Without efficient task scheduling, resources may not be utilized effectively [5]. Load balancing activities primarily occur in the backend of the cloud infrastructure. The articles reviewed in this paper focus specifically on server-side aspects of cloud computing.

In the realm of cloud computing, load balancing serves two pivotal objectives, as highlighted by Shafiq et al. (2019): resource allocation and task scheduling. Initially, tasks must be mapped to appropriate VMs to prevent overloaded or underutilized nodes. Subsequently, task scheduling techniques are applied post-allocation to optimize the task completion process, aligning with user requirements and meeting the specified deadlines outlined in the Service Level Agreement (SLA).

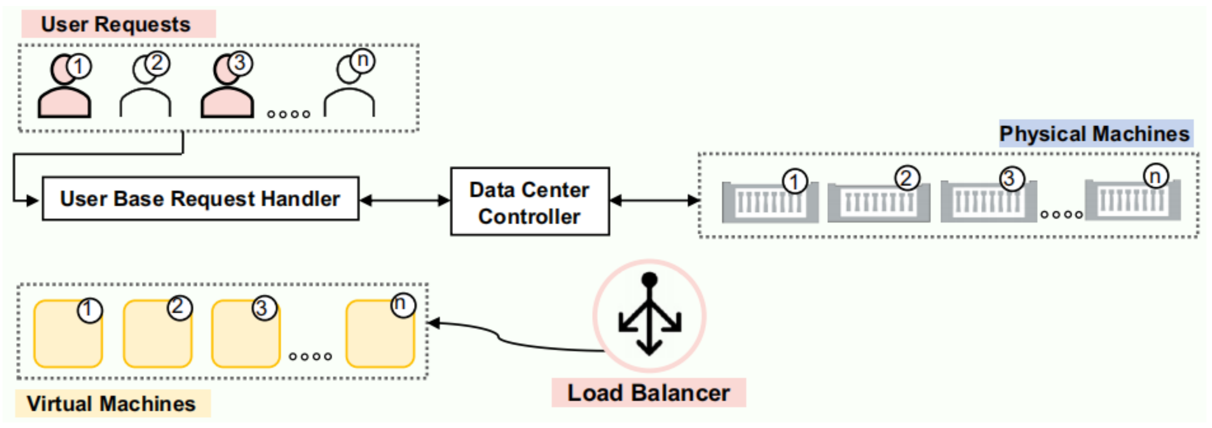


Fig 1.2

2.b.ii. Necessity of Load Balancing

Load balancing is a critical strategy in computer networks that involves distributing workloads across a variety of computing resources such as computers, clusters, network links, CPUs, or disk drives. The primary goals of load balancing are to optimize resource utilization, maximize throughput, minimize response time, and prevent any single resource from becoming overloaded. Utilizing multiple components with load balancing, rather than relying on a single component, enhances reliability through redundancy.

In cloud environments, load balancing differs from traditional architectures by employing commodity servers. This approach is chosen due to the inherent difficulty in accurately predicting the number of requests a server may encounter. While it offers new opportunities and economies of scale, it also introduces unique challenges. Load balancing is essential in cloud computing for ensuring the equitable distribution of dynamic local workloads across all nodes, thereby preventing imbalances and maintaining high customer satisfaction and optimal resource utilization ratios.

This mechanism significantly enhances overall system performance and resource efficiency by ensuring the fair and effective distribution of computing resources. It addresses and mitigates bottlenecks caused by load imbalances, ensuring service continuity. In cases of service component failures, load balancing enables seamless failover by managing the provisioning and de-provisioning of application instances. Figure 1 demonstrates the critical role of load balancing in the cloud, particularly in handling requests from multiple clients.

Current load balancing techniques in the cloud consider various parameters, including performance, response time, scalability, throughput, resource utilization, fault tolerance, migration time, and associated overhead. As cloud computing evolves to accommodate the proliferation of web-connected devices and vast data volumes, the primary focus remains on effectively balancing the ever-increasing load.[6]

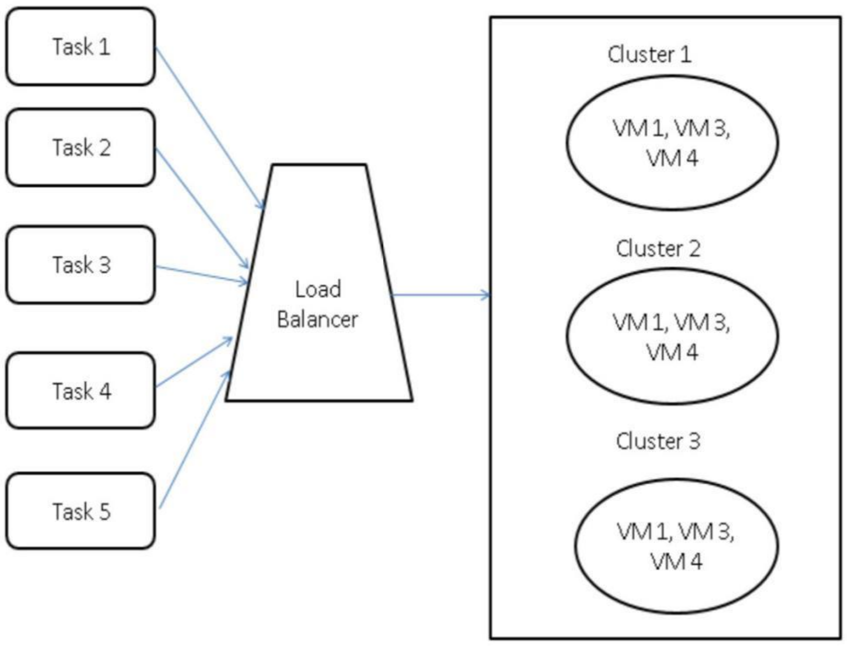


Fig 1.3

2.c. MIGRATION IN CLOUDSIM

The implementation of migration techniques in CloudSim is crucial for optimizing resource utilization and enhancing the efficiency of cloud computing environments. Specifically, virtual machine (VM) migration within CloudSim involves the seamless relocation of running VMs between physical hosts. This process is essential for achieving load balancing, resource consolidation, and adapting to dynamic changes in demand. CloudSim enables the simulation of various migration scenarios, allowing researchers and practitioners to experiment with and analyse different migration policies and algorithms.[7]

Live migration, a subset of VM migration, ensures uninterrupted service availability during the migration process. Additionally, CloudSim supports energy-aware migration strategies, which contribute to environmentally sustainable practices by minimizing energy consumption and reducing the overall carbon footprint. The toolkit's support for dynamic resource allocation allows for the modelling and evaluation of strategies to adapt cloud infrastructures to fluctuating workloads. Thus, CloudSim is an invaluable tool for studying and optimizing migration techniques in cloud computing environments.

2.c.i. Virtual Machine Migration

Virtual machine (VM) migration is a critical process in cloud computing, involving the seamless transfer of a running VM from one physical host to another. This dynamic relocation facilitates load balancing, ensuring optimal resource utilization and preventing server overloads. It enhances fault tolerance by enabling the evacuation of VMs from failing hosts, thereby contributing to uninterrupted service availability. VM migration also aids in resource consolidation, optimizing the use of computing resources.

In cloud simulations like CloudSim, researchers' model and analyse various migration strategies, including live migration, to optimize system performance, energy efficiency, and overall cloud infrastructure management.

2.c.ii. Live Migration

Live migration is a pivotal technique in cloud computing that enables the seamless relocation of a running virtual machine (VM) from one physical host to another without interrupting its operation. This process ensures continuous service availability and minimizes downtime, making it essential for maintaining a responsive and resilient cloud infrastructure. Live migration is particularly valuable during scenarios such as load balancing, hardware maintenance, and addressing performance issues.

By transferring a VM while it remains active, live migration optimizes resource utilization, enhances fault tolerance, and increases overall system flexibility. This dynamic approach allows cloud providers to efficiently manage their resources, adapt to changing workloads, and ensure a smooth, uninterrupted experience for users accessing applications and services hosted on virtual machines.

2.c.iii. Impact of migration towards Green Computing

The influence of migration on green computing is profound, as it fosters more sustainable and environmentally conscious practices within the domain of information technology. Here are key ways in which migration strategies, particularly in cloud computing, impact green computing:

Enhanced Energy Efficiency:

Migration facilitates the consolidation of workloads and efficient allocation of resources, resulting in a reduction in the number of active servers. This leads to decreased energy consumption, aligning with the objectives of green computing.

Optimized Resource Utilization:

Through migration, cloud providers can optimize resource utilization, ensuring that servers operate at peak efficiency. This reduces the need for additional hardware and subsequently lowers energy requirements.

Dynamic Resource Scaling:

Migration enables dynamic scaling of resources based on demand. During periods of low demand, resources can be consolidated and unnecessary servers powered down, conserving energy and promoting environmentally responsible practices.

Reduced Environmental Impact:

Green computing emphasizes minimizing the environmental footprint of IT operations. Efficient resource management through migration helps organizations lower their carbon footprint and foster a more sustainable computing environment.

Efficient Hardware Management:

Migration facilitates effective management of hardware life-cycles. By relocating workloads from aging or under-performing hardware, organizations can extend the lifespan of existing equipment, reducing electronic waste and promoting responsible disposal practices.

Integration with Renewable Energy Sources:

Cloud providers can strategically migrate workloads to data centres powered by renewable energy, thereby minimizing carbon emissions associated with data center operations.

Cost-Efficiency:

Migration-enabled green computing practices not only benefit the environment but also lead to cost savings. Reduced energy consumption and optimal resource utilization contribute to lower operational costs for organizations.

In summary, migration strategies play a pivotal role in advancing green computing objectives by enhancing energy efficiency, optimizing resource utilization, and aligning IT operations with environmentally sustainable practices. This not only results in cost savings but also contributes to building a more ecologically responsible and resilient IT infrastructure.[8]

2.d. MACHINE LEARNING

Machine Learning (ML) plays a transformative role in enhancing load balancing strategies within cloud computing environments. By leveraging ML algorithms, cloud systems can predict, adapt, and dynamically optimize resource allocation, thereby significantly improving overall system performance and efficiency. In this project, we have integrated a linear regression algorithm to determine optimal threshold values for virtual machine (VM) migration within the CloudSim framework. This approach aims to minimize energy consumption while maintaining balanced workloads, ensuring that cloud resources are utilized effectively and sustainably.[8]

2.d.i. Importance of Machine Learning

Machine Learning (ML) has emerged as a pivotal technology in the modern digital landscape, driving innovation and efficiency across various industries. Its ability to process vast amounts of data, identify patterns, and make predictions without explicit programming makes it an indispensable tool for businesses, researchers, and technologists. Here are key reasons highlighting the importance of ML:

Enhanced Decision-Making

Machine Learning algorithms can analyse complex datasets and uncover insights that might be overlooked by traditional data analysis methods. By identifying patterns and trends, ML aids in making informed decisions, reducing the reliance on intuition and enhancing strategic planning. For instance, in finance, ML models predict market trends and manage risks, while in healthcare, they assist in diagnosing diseases and personalizing treatment plans.

Automation and Efficiency

ML enables the automation of repetitive and mundane tasks, allowing human resources to focus on more strategic activities. In manufacturing, ML-driven robots and systems optimize production processes, ensuring high-quality output with minimal human intervention. In customer service, chatbots powered by ML provide immediate responses to common inquiries, improving customer satisfaction and operational efficiency.

Predictive Maintenance

In industries such as manufacturing, energy, and transportation, ML is crucial for predictive maintenance. By analysing data from equipment sensors, ML models predict potential failures before they occur, reducing downtime and maintenance costs. This proactive approach ensures the reliability and longevity of machinery and infrastructure.

Personalization

ML algorithms excel at providing personalized experiences to users. In e-commerce, ML analyses user behaviour and preferences to recommend products tailored to individual tastes. Streaming services like Netflix and Spotify use ML to suggest movies and music based on past viewing and listening habits, enhancing user engagement and satisfaction.

Improved Healthcare

ML significantly impacts healthcare by enhancing diagnostic accuracy, predicting patient outcomes, and personalizing treatment plans. For example, ML models can analyse medical images to detect early signs of diseases such as cancer, often with greater accuracy than human experts. Predictive models assess patient data to foresee complications and suggest preventive measures, improving patient care and outcomes.

Fraud Detection

In banking and finance, ML algorithms detect fraudulent activities by analysing transaction patterns and identifying anomalies that deviate from typical behaviour. These systems provide real-time alerts, helping institutions prevent fraud and protect customers' assets. Similarly, in cybersecurity, ML detects and responds to threats by recognizing patterns of malicious activity.

Enhanced User Experience

ML enhances user experience across various applications by providing intuitive and responsive interfaces. Voice-activated assistants like Siri and Alexa leverage ML for natural language processing, enabling seamless interactions with users. ML-driven recommendation engines in social media platforms curate content feeds based on user preferences, keeping users engaged and satisfied.

Scalability and Adaptability

ML systems can handle large-scale data and adapt to changing conditions, making them suitable for dynamic and complex environments. In cloud computing, ML optimizes resource allocation, ensures efficient load balancing, and improves energy efficiency. This scalability and adaptability are critical for businesses looking to remain competitive in rapidly evolving markets.

Driving Innovation

ML is at the forefront of technological innovation, enabling breakthroughs in fields such as artificial intelligence, autonomous systems, and data science. Autonomous vehicles, for example, rely heavily on ML for navigation, object detection, and decision-making processes. In research, ML accelerates scientific discovery by analysing vast datasets and identifying potential areas for exploration.

Economic Impact

The economic impact of ML is profound, with industries adopting ML technologies experiencing increased productivity and growth. Companies leveraging ML gain a competitive edge by optimizing operations, reducing costs, and creating new business models. This transformative technology fosters economic development and drives progress in various sectors.[11]

2.d.ii. Role of Machine Learning in CloudSim

In the dynamic and complex landscape of cloud computing, traditional load balancing techniques often fall short in managing resources efficiently due to the unpredictable and fluctuating nature of workloads. ML algorithms address these challenges by analysing historical data and current system states to make informed predictions about future resource demands. This predictive capability enables proactive management of resources, reducing the likelihood of overloading or underutilizing servers, and enhancing the overall stability and reliability of cloud services.

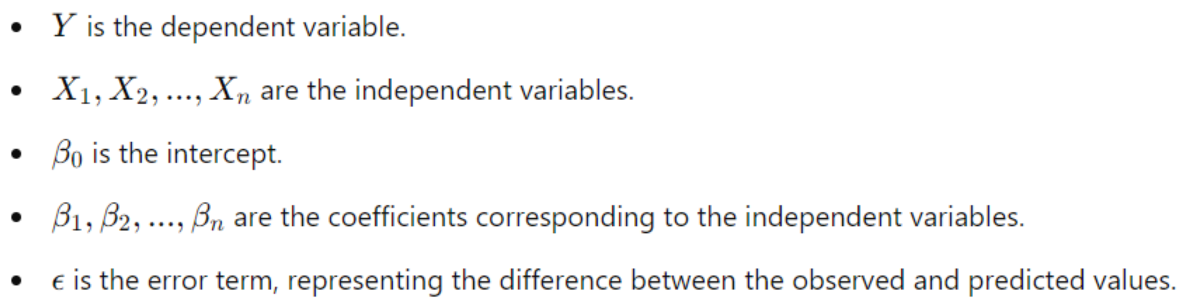
2.d.iii. Linear Regression

Linear regression is one of the fundamental algorithms in machine learning and statistics, primarily used for predictive modelling. Its simplicity, interpretability, and effectiveness in modelling linear relationships between variables make it a widely utilized tool in various fields such as economics, finance, healthcare, and engineering.[9]

Concepts and Principles

Linear regression aims to model the relationship between a dependent variable (also called the response or target) and one or more independent variables (also known as predictors or features) by fitting a linear equation to the observed data. The simplest form, known as simple linear regression, involves a single predictor, whereas multiple linear regression involves multiple predictors.
The general form of the linear regression equation is:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X2_2 + \ldots + \beta_n X_n + \epsilon$$



- Y is the dependent variable.
- $X_1, X_2, \ldots, X_n$ are the independent variables.
- β_0 is the intercept.
- $\beta_1, \beta_2, \ldots, \beta_n$ are the coefficients corresponding to the independent variables.
- ϵ is the error term, representing the difference between the observed and predicted values.

Role of Linear Regression in CloudSim :

Linear regression can be applied in CloudSim to optimize resource utilization and enhance the efficiency of cloud computing environments.[10] CloudSim is a simulation tool used for modelling and simulating cloud computing infrastructures and services. By integrating linear regression within CloudSim, various aspects of cloud management can be improved, including resource allocation, load balancing, and energy optimization.

Here's how linear regression can be utilized in CloudSim:

1. **Resource Allocation:** Linear regression can analyse historical resource usage data to predict future resource demands. By understanding the relationship between resource utilization metrics (such as CPU usage, memory usage) and workload patterns, linear regression models can forecast resource requirements for different VMs or applications. This predictive capability enables cloud providers to allocate resources more efficiently, ensuring that VMs have the necessary resources to meet performance requirements without over-provisioning.

2. **Load Balancing:** Linear regression can aid in load balancing by predicting the optimal distribution of workloads across cloud servers. By analysing historical workload data and system performance metrics, linear regression models can identify trends and patterns that indicate imbalances in resource utilization. Based on these insights, load balancing algorithms can dynamically adjust VM placements to evenly distribute workloads and optimize resource usage. This ensures that no server is overloaded while others remain underutilized, maximizing overall system performance.

3. **Energy Optimization:** Linear regression can help optimize energy consumption in cloud data centres by predicting energy usage based on resource utilization patterns. By understanding the relationship between resource usage and energy consumption, linear regression models can identify opportunities for energy savings. For example, by predicting periods of low resource demand, cloud providers can dynamically adjust server power states or migrate VMs to fewer servers, reducing energy consumption during off-peak hours.

4. **Performance Monitoring and Prediction**: Linear regression can be used for performance monitoring and prediction in CloudSim environments. By analysing historical performance data, such as response times, throughput, and latency, linear regression models can identify performance bottlenecks and predict future performance trends. This information enables cloud providers to proactively address performance issues, optimize resource allocation, and ensure high levels of service quality for cloud users.[10]

Overall, integrating linear regression into CloudSim enables cloud providers to make data-driven decisions, optimize resource utilization, and enhance the efficiency of cloud computing environments. By leveraging predictive analytics capabilities, cloud providers can improve system performance, reduce costs, and deliver better quality of service to users.

3. ALGORITHMS

3.a. Virtual Machine Migration based on Threshold value:

Input:

- List of VMs on each host
- Threshold values for CPU, memory, or other relevant metrics
- Migration policy (e.g., load balancing strategy)

Output:

- Updated VM distribution across hosts

Procedure:

1. *Initialize variables and parameters:*

    a. Set threshold values for resource utilization metrics (e.g., CPU, memory).
    b. Define the migration policy (e.g., least loaded, round-robin).

2. *Periodically or when triggered, repeat the following steps:*

    a. Monitor resource utilization metrics (e.g., CPU, memory) for each host and VM.
    b. Identify hosts that exceed predefined threshold values for resource utilization.

3. *For each host exceeding threshold values:*

    a. Select VMs on the overloaded host based on the migration policy (e.g., least loaded, round-robin).
    b. Evaluate candidate destination hosts for migration based on resource availability and load.

4. *Migrate selected VMs to the chosen destination hosts:*

    a. Initiate live migration for seamless transitions.
    b. Update the VM distribution across hosts.

5. Repeat the monitoring and migration process to maintain load balance and prevent resource saturation.

6. End Algorithm

3.b. Machine Learning – Linear Regression for finding Threshold value:

Step-1: Import Necessary Libraries

i. *pandas* for data manipulation.
ii. *StandardScaler* from sklear.preprocessing for data manipulation.
iii. *train_test_split* from sklearn.model_selection for splitting the data.
iv. *LinearRegression* from sklearn.linear_model for building the regression model.
v. *mean_absolte_error* from sklearn.metrics for model evaluation.

Step-2: Load the CSV file:

i. Use pd.read_csv file to read the data from the CSV file.

Step-3: Convert specific columns into numeric values:

i. Define a list column_to_convert containing the names of the columns to be converted.
ii. For each column, remove specific text (e.g., ‘sec’, ‘kWh’) and convert the column to numeric values using pd.to_numeric. Handle any errors by coercing invalid values to NaN. Drop rows with NaN using dropna().

Step-4 : Split the dataset into features (x) and target (y)

i. Set x to contain the columns : ‘Number of hosts:’, ‘Number of VMs:’, ‘Threshold values’.
ii. Set y to contain the ‘energy consumption’ column.

Step-5 : Normalise the feature data (if needed) :

i. Initialise a StandardScaler.
ii. Fit and transform X using the scaler to obtain X_scaled.

Step-6 : Split the data into training and test sets

i. Use train_test_split to split x and y into training and test sets with a test size of 40%.

Step-7 : Train the Linear Regression model

i. Initialise a LinearRegression model.
ii. Fit the model using the training data(x_train, y_train).

Step-8: Predict on the test set

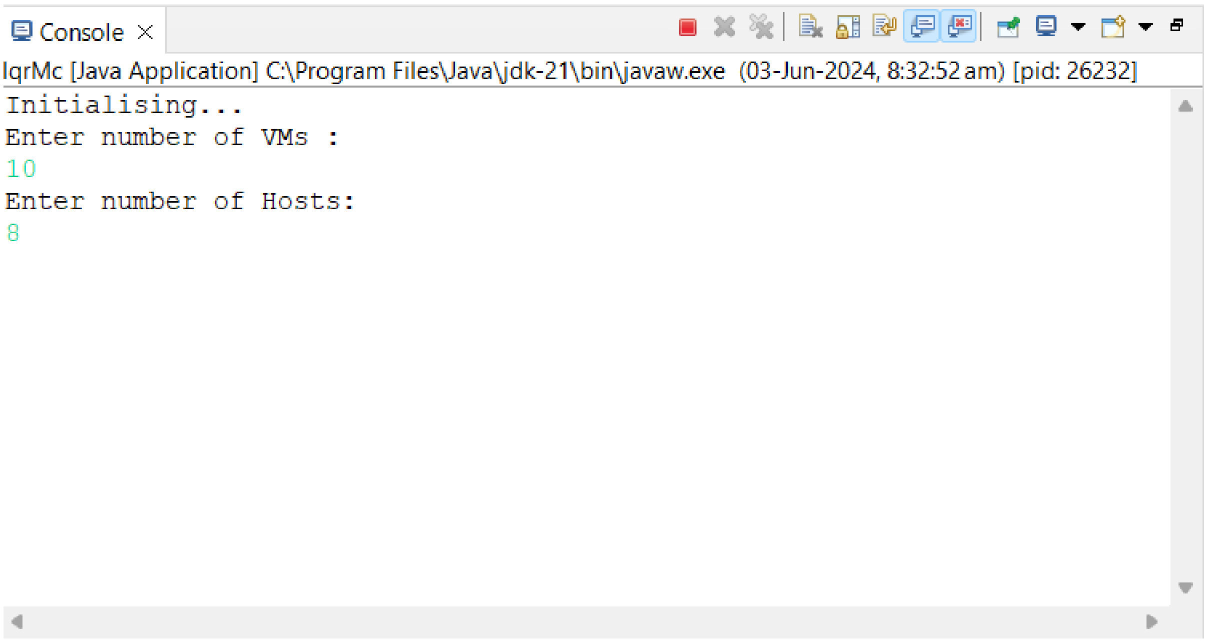
i. Use the trained model to make predictions on the test set (x_test).

Step-9: Evaluate the model

i. Calculate the Mean Absolute Error(MAE) between the actual values (y_test) and the predicted values(y_pred).
ii. Print the MAE.

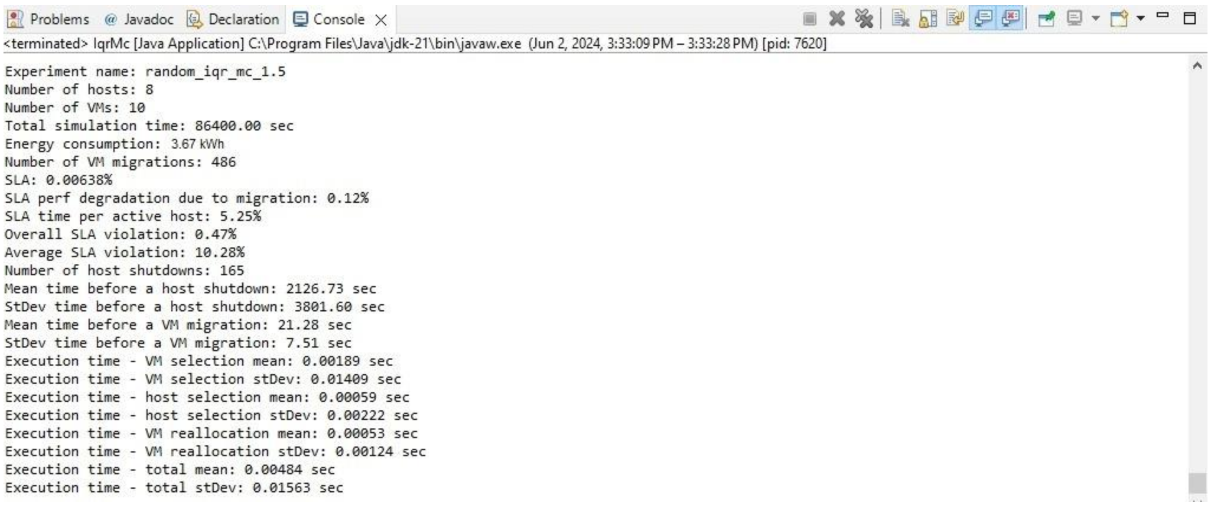
4. RESULT & ANALYSIS

OUTPUT-1: Setting up values for VMs and Hosts.



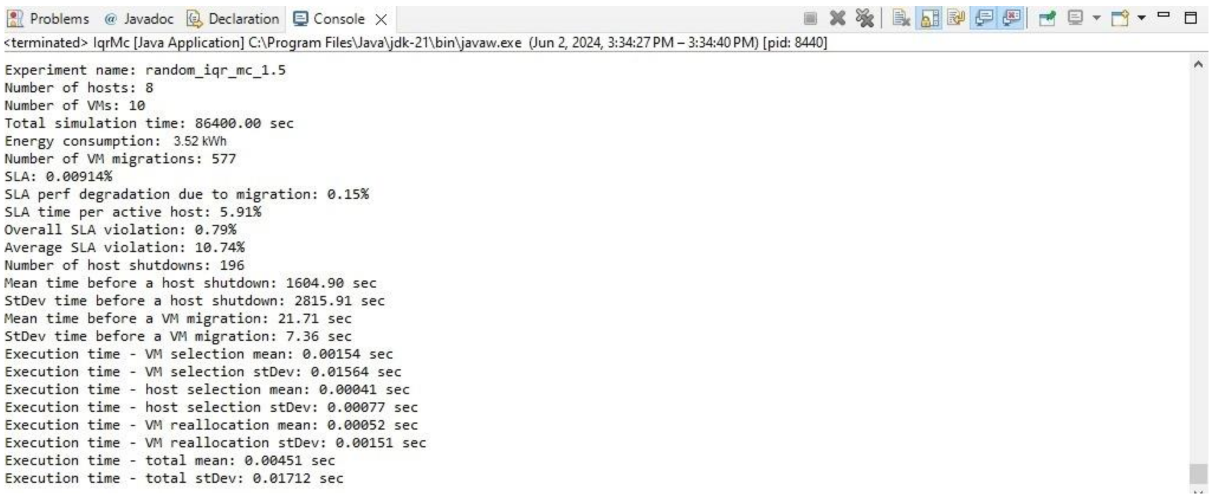
```
IqrMc [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe  (03-Jun-2024, 8:32:52 am) [pid: 26232]
Initialising...
Enter number of VMs :
10
Enter number of Hosts:
8
```

OUTPUT-2: Simulation of Virtual Machine Migration and setting up threshold value = 0.2



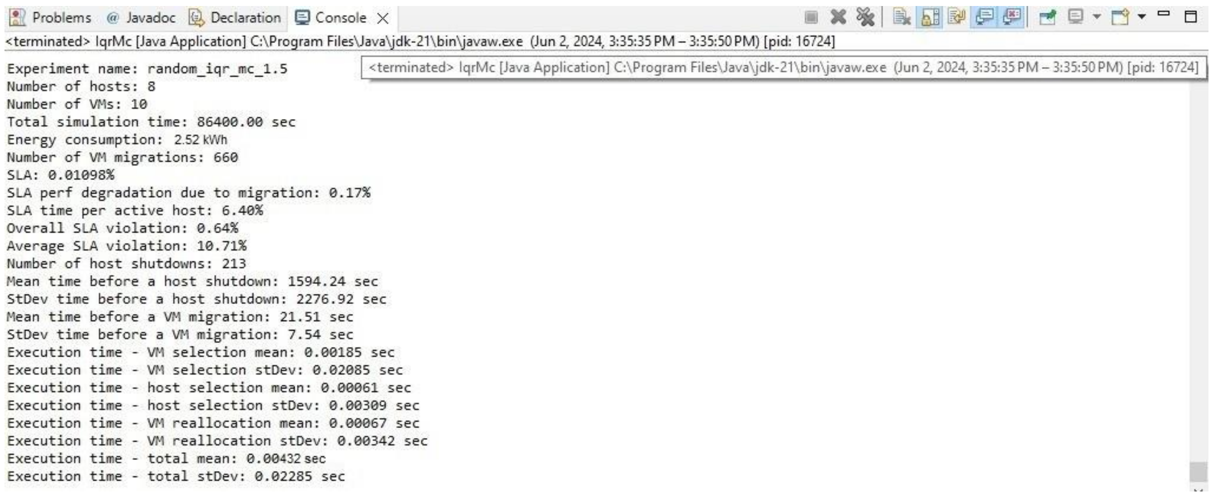
```
<terminated> IqrMc [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe  (Jun 2, 2024, 3:33:09 PM – 3:33:28 PM) [pid: 7620]
Experiment name: random_iqr_mc_1.5
Number of hosts: 8
Number of VMs: 10
Total simulation time: 86400.00 sec
Energy consumption: 3.67 kWh
Number of VM migrations: 486
SLA: 0.00638%
SLA perf degradation due to migration: 0.12%
SLA time per active host: 5.25%
Overall SLA violation: 0.47%
Average SLA violation: 10.28%
Number of host shutdowns: 165
Mean time before a host shutdown: 2126.73 sec
StDev time before a host shutdown: 3801.60 sec
Mean time before a VM migration: 21.28 sec
StDev time before a VM migration: 7.51 sec
Execution time - VM selection mean: 0.00189 sec
Execution time - VM selection stDev: 0.01409 sec
Execution time - host selection mean: 0.00059 sec
Execution time - host selection stDev: 0.00222 sec
Execution time - VM reallocation mean: 0.00053 sec
Execution time - VM reallocation stDev: 0.00124 sec
Execution time - total mean: 0.00484 sec
Execution time - total stDev: 0.01563 sec
```

OUTPUT-3: Simulation of Virtual Machine Migration and setting up threshold value = 0.4


Problems @ Javadoc Declaration Console

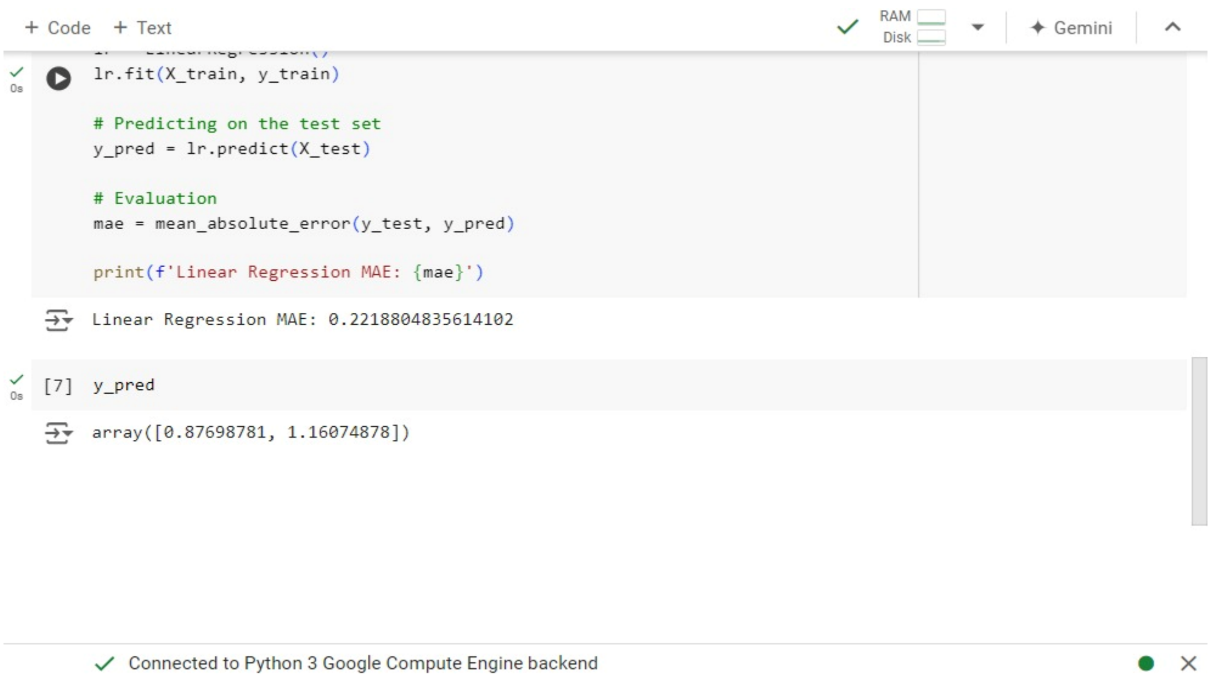
```
<terminated> IqrMc [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (Jun 2, 2024, 3:34:27 PM – 3:34:40 PM) [pid: 8440]
Experiment name: random_iqr_mc_1.5
Number of hosts: 8
Number of VMs: 10
Total simulation time: 86400.00 sec
Energy consumption: 3.52 kWh
Number of VM migrations: 577
SLA: 0.00914%
SLA perf degradation due to migration: 0.15%
SLA time per active host: 5.91%
Overall SLA violation: 0.79%
Average SLA violation: 10.74%
Number of host shutdowns: 196
Mean time before a host shutdown: 1604.90 sec
StDev time before a host shutdown: 2815.91 sec
Mean time before a VM migration: 21.71 sec
StDev time before a VM migration: 7.36 sec
Execution time - VM selection mean: 0.00154 sec
Execution time - VM selection stDev: 0.01564 sec
Execution time - host selection mean: 0.00041 sec
Execution time - host selection stDev: 0.00077 sec
Execution time - VM reallocation mean: 0.00052 sec
Execution time - VM reallocation stDev: 0.00151 sec
Execution time - total mean: 0.00451 sec
Execution time - total stDev: 0.01712 sec
```

OUTPUT-4: Simulation of Virtual Machine Migration and setting up threshold value = 0.7


Problems @ Javadoc Declaration Console

```
<terminated> IqrMc [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (Jun 2, 2024, 3:35:35 PM – 3:35:50 PM) [pid: 16724]
Experiment name: random_iqr_mc_1.5     <terminated> IqrMc [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (Jun 2, 2024, 3:35:35 PM – 3:35:50 PM) [pid: 16724]
Number of hosts: 8
Number of VMs: 10
Total simulation time: 86400.00 sec
Energy consumption: 2.52 kWh
Number of VM migrations: 660
SLA: 0.01098%
SLA perf degradation due to migration: 0.17%
SLA time per active host: 6.40%
Overall SLA violation: 0.64%
Average SLA violation: 10.71%
Number of host shutdowns: 213
Mean time before a host shutdown: 1594.24 sec
StDev time before a host shutdown: 2276.92 sec
Mean time before a VM migration: 21.51 sec
StDev time before a VM migration: 7.54 sec
Execution time - VM selection mean: 0.00185 sec
Execution time - VM selection stDev: 0.02085 sec
Execution time - host selection mean: 0.00061 sec
Execution time - host selection stDev: 0.00309 sec
Execution time - VM reallocation mean: 0.00067 sec
Execution time - VM reallocation stDev: 0.00342 sec
Execution time - total mean: 0.00432 sec
Execution time - total stDev: 0.02285 sec
```

OUTPUT-5: The results of machine learning algorithm obtained. Platform used: GoogleCollab.



```
lr.fit(X_train, y_train)

# Predicting on the test set
y_pred = lr.predict(X_test)

# Evaluation
mae = mean_absolute_error(y_test, y_pred)

print(f'Linear Regression MAE: {mae}')
```

```
Linear Regression MAE: 0.2218804835614102
```

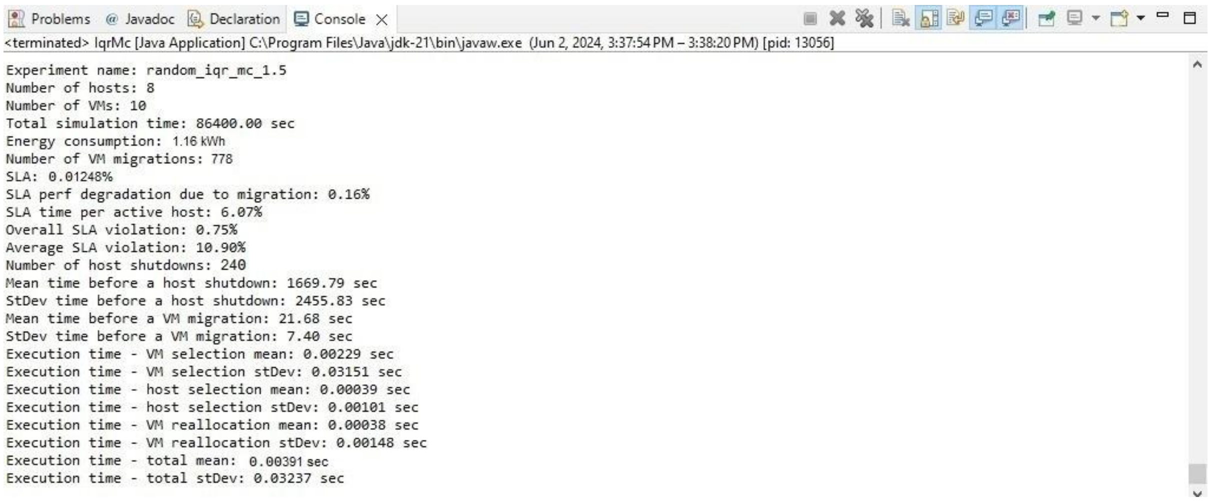
```
[7] y_pred
```

```
array([0.87698781, 1.16074878])
```

Connected to Python 3 Google Compute Engine backend

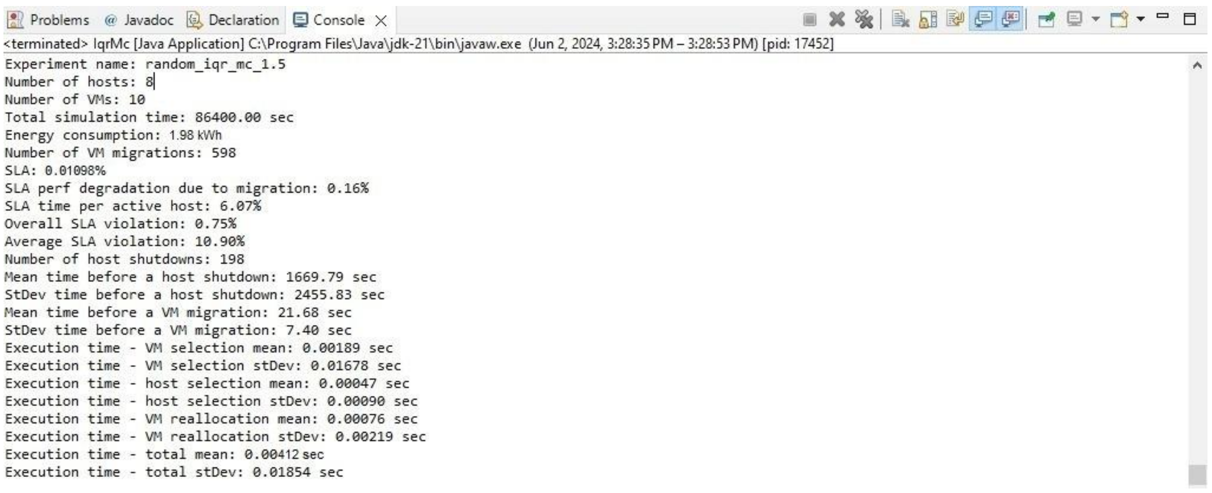
Here, we have obtained the accuracy as 80% (approx.) and the threshold value = 0.87 in which we get the lowest energy value.

OUTPUT-6: Simulation of Virtual Machine Migration and setting up threshold value = 0.87 obtained from the LinearRegression Algorithm.



```
<terminated> IqrMc [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (Jun 2, 2024, 3:37:54 PM – 3:38:20 PM) [pid: 13056]
Experiment name: random_iqr_mc_1.5
Number of hosts: 8
Number of VMs: 10
Total simulation time: 86400.00 sec
Energy consumption: 1.16 kWh
Number of VM migrations: 778
SLA: 0.01248%
SLA perf degradation due to migration: 0.16%
SLA time per active host: 6.07%
Overall SLA violation: 0.75%
Average SLA violation: 10.90%
Number of host shutdowns: 240
Mean time before a host shutdown: 1669.79 sec
StDev time before a host shutdown: 2455.83 sec
Mean time before a VM migration: 21.68 sec
StDev time before a VM migration: 7.40 sec
Execution time - VM selection mean: 0.00229 sec
Execution time - VM selection stDev: 0.03151 sec
Execution time - host selection mean: 0.00039 sec
Execution time - host selection stDev: 0.00101 sec
Execution time - VM reallocation mean: 0.00038 sec
Execution time - VM reallocation stDev: 0.00148 sec
Execution time - total mean: 0.00391 sec
Execution time - total stDev: 0.03237 sec
```

OUTPUT-7: Simulation of Virtual Machine Migration and setting up threshold value = 0.9



```
<terminated> IqrMc [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (Jun 2, 2024, 3:28:35 PM – 3:28:53 PM) [pid: 17452]
Experiment name: random_iqr_mc_1.5
Number of hosts: 8
Number of VMs: 10
Total simulation time: 86400.00 sec
Energy consumption: 1.98 kWh
Number of VM migrations: 598
SLA: 0.01098%
SLA perf degradation due to migration: 0.16%
SLA time per active host: 6.07%
Overall SLA violation: 0.75%
Average SLA violation: 10.90%
Number of host shutdowns: 198
Mean time before a host shutdown: 1669.79 sec
StDev time before a host shutdown: 2455.83 sec
Mean time before a VM migration: 21.68 sec
StDev time before a VM migration: 7.40 sec
Execution time - VM selection mean: 0.00189 sec
Execution time - VM selection stDev: 0.01678 sec
Execution time - host selection mean: 0.00047 sec
Execution time - host selection stDev: 0.00090 sec
Execution time - VM reallocation mean: 0.00076 sec
Execution time - VM reallocation stDev: 0.00219 sec
Execution time - total mean: 0.00412 sec
Execution time - total stDev: 0.01854 sec
```

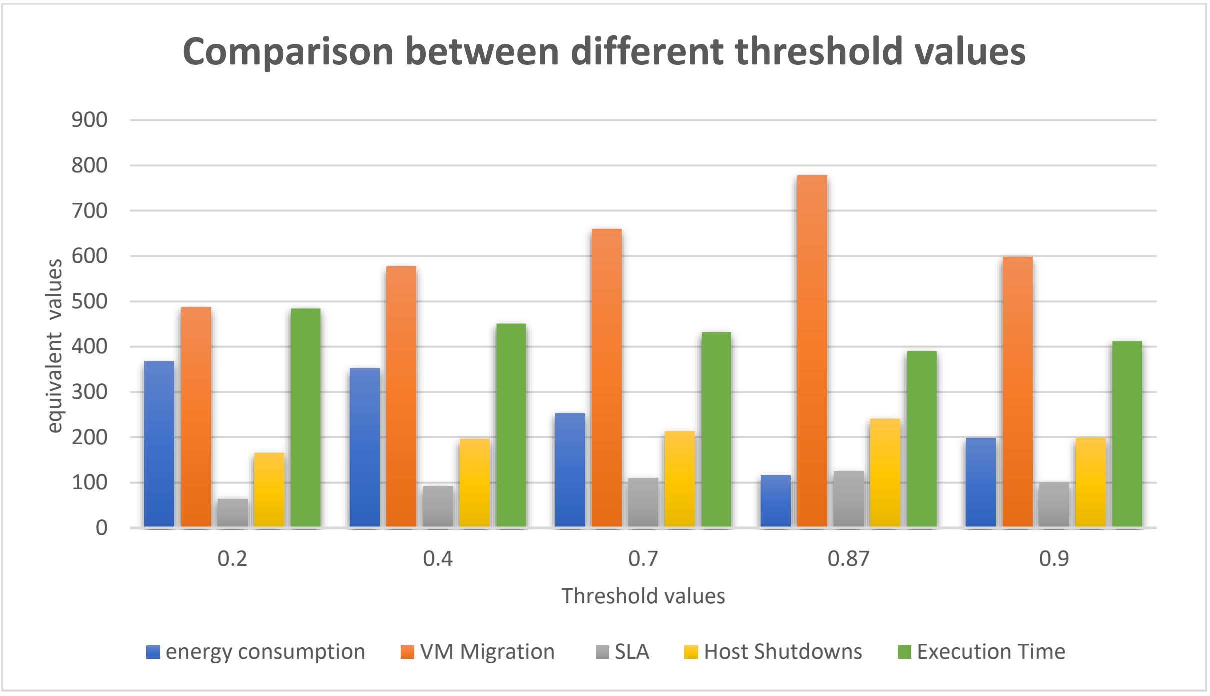
<u>Result Analysis:</u>

Utilisation	Energy Consumption	VM Migrations	SLA	Host Shutdowns	Execution Time
0.2	3.67kWh	486	0.00638	165	0.00484sec
0.4	3.52kWh	577	0.00914	196	0.00451sec
0.7	2.52kWh	660	0.01098	213	0.00432sec
0.87	1.16kWh	778	0.01248	240	0.00391sec
0.9	1.98kWh	598	0.00998	198	0.00412sec

We will construct a graphical representation of the above table. To enhance clarity, we have adjusted the values by converting them into equivalent units.

i. 1kWh = 100 dagWh (deca-gram Watt hour)
ii. SLA = x*10^{-4} units.
iii. Execution time : 1sec = 1000,00 * 10 microsecs

Equivalent graph to the result can be constructed as :



The graph presents a comparison between different threshold values (0.2, 0.4, 0.7, 0.87, and 0.9) for several metrics: energy consumption, VM migration, SLA, host shutdowns, and execution time.

Overall, the threshold value of 0.87 obtained from the machine learning algorithm provides optimized energy consumption, achieving lower values compared to manually entered thresholds. However, increasing the threshold to 0.9 results in a significant spike in energy consumption, demonstrating that the 0.87 threshold is critical for maintaining energy efficiency.

5. CONCLUSION

This project has provided a comprehensive examination of load balancing in cloud computing environments, utilizing the CloudSim simulator to model and simulate various scenarios. By implementing virtual machine (VM) migration strategies based on resource utilization and integrating a Machine Learning algorithm for threshold optimization, this study has demonstrated significant improvements in both load balancing and energy efficiency.

The findings underscore the critical role of dynamic resource allocation in enhancing cloud infrastructure performance. The experimental results validate the effectiveness of the proposed approach, particularly highlighting the balance achieved between load distribution and energy consumption. However, the increase in execution time with more frequent migrations indicates a need for careful consideration of the trade-offs involved.

The application of Linear Regression for analysing historical datasets and setting optimal threshold values represents a novel contribution to the field, offering a data-driven method for resource management. This approach not only improves load balancing but also contributes to the broader goals of green cloud computing by minimizing energy usage.

In summary, this project reinforces the importance of adaptive and intelligent resource management strategies in cloud computing. The insights gained pave the way for future research, particularly in exploring more refined algorithms and expanding the scope of energy-aware scheduling. The ongoing evolution of cloud computing demands continuous innovation, and this study serves as a foundational step towards more efficient and sustainable cloud resource management.

6. SCOPE OF FUTURE WORK

The findings from this project open several avenues for future exploration to further enhance the efficiency and effectiveness of load balancing and energy management in cloud computing environments.

- ***Refinement of Machine Learning Algorithms***: While the linear regression algorithm has proven effective in determining optimal threshold values for VM migration, there is significant potential to explore more sophisticated machine learning models. Algorithms such as neural networks, decision trees, or reinforcement learning could offer improved accuracy and adaptability in dynamic cloud environments.

- ***User-Centric Load Balancing:*** Incorporating user behaviour and preferences into load balancing decisions could improve user satisfaction and service quality. This approach would require developing models that account for user patterns and dynamically adjust resources to meet their specific needs.

By addressing these areas, future research can build on the insights gained from this project, leading to more refined, efficient, and sustainable cloud computing practices. These efforts will contribute to the ongoing evolution of cloud infrastructure management, ensuring that it meets the growing demands of modern applications and services

7.BIBLIOGRAPHY

[1] A. Beloglazov, J. Abawajy, and R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for Cloud computing," Future Generation Computer Systems, vol. 28, no. 5, pp. 755-768, 2012.

[2] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," Software: Practice and Experience, vol. 41, no. 1, pp. 23-50, 2011.

[3] H. Liu, "A new approach to VM live migration using CPU scheduling," in Proceedings of the 2013 International Conference on Cloud and Service Computing, 2013, pp. 1-8.

[4] J. Hu, J. Gu, G. Sun, and T. Zhao, "A scheduling strategy on load balancing of virtual machine resources in cloud computing environment," in 2010 3rd International Symposium on Parallel Architectures, Algorithms and Programming, 2010, pp. 89-96.

[5] M. Masdari, S. ValiKardan, Z. Shahi, and S. Azar, "Towards workflow scheduling in cloud computing: A comprehensive analysis," Journal of Network and Computer Applications, vol. 66, pp. 64-82, 2016.

[6] G. Aceto, A. Botta, W. De Donato, and A. Pescape, "Cloud monitoring: A survey," Computer Networks, vol. 57, no. 9, pp. 2093-2115, 2013.

[7] Z. Xiao, W. Song, and Q. Chen, "Dynamic resource allocation using virtual machines for cloud computing environment," IEEE Transactions on Parallel and Distributed Systems, vol. 24, no. 6, pp. 1107-1117, 2013.

[8] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553, pp. 436-444, 2015.

[9] G. A. F. Seber and A. J. Lee, Linear Regression Analysis, 2nd ed. Wiley, 2003.

[10] D. C. Montgomery, E. A. Peck, and G. G. Vining, Introduction to Linear Regression Analysis, 5th ed. Wiley, 2012.

[11] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups," IEEE Signal Processing Magazine, vol. 29, no. 6, pp. 82-97, 2012.