A

Project Report

On

A COMPREHENSIVE SURVEY OF LOAD BALANCING TECHNIQUES: FROM CLASSIC METHODS TO MODERN ALGORITHMS

Submitted to

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE AND TECHNOLOGIES
RK VALLEY

in partial fulfillment of the requirement for the award of the Degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE & ENGINEERING

Submitted by

MALA SINDHU(R180114)

KAMASANI JAYASREE(R180885)

GOVINDHU REDDY LATHA(R180004)

Under the Guidance of

Mr. K.VINOD KUMAR, Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES

(catering the Educational Needs of Gifted Rural Youth of AP)

R.K Valley, Vempalli(M), Kadapa(Dist) – 516330

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES

A.P.Government Act 18 of 2008

RGUKT-RK Valley

Vempalli, Kadapa, Andhra Pradesh - 516330.

CERTIFICATE OF PROJECT COMPLETION

This is to certify that I have examined the thesis entitled "**A Comprehensive Survey of Load Balancing Techniques: From Calssic Methods to Modern Algorithms**" submitted by **GOVINDHU REDDYLATHA(R180004)**, **KAMASANI JAYASREE(R180885), MALA SINDHU(R180114)** under our guidance and supervision for the partial fulfillment for the degree of Bachelor of Technology in computer Science and Engineering during the academic session February 2023 – July 2023 at RGUKT-RK VALLEY.

Project Guide

Mr K.Vinod Kumar,
Asst.Prof. in Dept of CSE,
RGUKT-RK Valley.

Head of the Department

Mr. N.Satyanandaram,
Lecturer in Dept of CSE,
RGUKT-RK Valley.

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of the people who made it possible and who's constant guidance and encouragement crown all the efforts success.

We would like to express our sincere gratitude to **Mr. K.Vinod Kumar**, my project guide for valuable suggestions and keen interest throughout the progress of our project.

We are grateful **Mr. N. Satyanandaram HOD CSE**, for providing excellent computing facilities and congenial atmosphere for progressing our project.

At the outset, we would like to thank **Rajiv Gandhi University of Knowledge Technologies(RGUKT RK VALLEY)** , for providing all the necessary resources and support for the successful completion of my course work.

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES

(A.P.Government Act 18 of 2008) RGUKT-RK Valley

Vempalli, Kadapa, Andhra Pradesh-516330.

DECLARATION

We, **Govindhu ReddyLatha(R180004), Mala Sindhu(R180114), Kamasani Jayasree(R180885)** hereby declare that the project report entitled **"A Comprehensive Survey of Load Balancing Techniques: From Calssic Methods to Modern Algorithms"** done under guidance of **Mr. K.Vinod Kumar** is submitted in partial fulfillment for the degree of Bachelor of Technology in Computer Science and Engineering during the academic session February 2023 – July 2023 at RGUKT-RK Valley. I also declare that this project is a result of our own effort and has not been copied or imitated from any source. Citations from any websites are mentioned in the references. To the best of my knowledge, the results embodied in this dissertation work have not been submitted to any university or institute for the award of any degree or diploma.

Date :
Place : RK Valley

Kamasani Jayasree – R180885
Mala Sindhu-R180114
Govindhu Reddy Latha-R180004

INDEX

ABSTRACT

Now-a-days, Cloud Computing has become a cornerstone of modern technology, driving innovation, efficiency and accessibility across various industries and application. Distributed computing solves the difficulty of using distributed autonomous machines and communicating with each other over a network. Cloud computing provides clients with a range of services and capabilities that enhances productivity ,accessibility and scalability while reducing the need for extensive hardware and infrastructure investments. when interest in distributed computing rises, it implies that more individuals, businesses, and organizations are exploring, adopting, and implementing distributed computing solutions. This surge in interest leads to increase in data traffic. There are two solutions for the issue of increase in data traffic,one is to server optimization (or) server performance enhancement(upgrade single server to a high performance server) but upgraded server may exceed its capacity(overload).Another one is multi servers. Multi-server configurations are common in scenarios where the demands of an application or service exceed the capabilities of a single server, and the distribution of tasks across multiple servers is necessary for optimal performance and reliability. When there are multiple servers,there is an issue i.e. Load Adjusting. Load Balancing is one of the critical issues in cloud computing. In a cloud environment, where resources are often dynamically allocated and distributed, load balancing plays a central role in managing workloads efficiently. Load balancing in cloud computing is a technique used to distribute computing workloads and network traffic across multiple servers or resources within a cloud environment. The primary goal of load balancing is to optimize resource utilization, prevent individual servers from becoming overloaded, and ensure that the overall system can handle varying levels of demand efficiently. Here we also discussed about the hybrid of Cat and Mouse Optimization and Grey Wolf Optimization algorithms.

OBJECTIVES

The objective of load balancing, in simple words, is to distribute work or tasks evenly among a group of resources (such as servers, processors, or network links) to ensure optimal and efficient utilization. This helps prevent some resources from becoming overwhelmed with too much work while others remain underutilized. The goal is to achieve better performance,

minimize response time and more reliable operation of a system or network.

The ultimate objective of Cat and Mouse Based Optimization and Grey Wolf Optimization is to find the best set of parameters or values for a given optimization problem. These algorithms leverage nature-inspired behaviors to guide the search process, helping to efficiently navigate the solution space and converge towards optimal solutions. The specific mechanisms and behaviors mimicked from nature contribute to the algorithms' ability to handle different types of optimization problems.

SCOPE

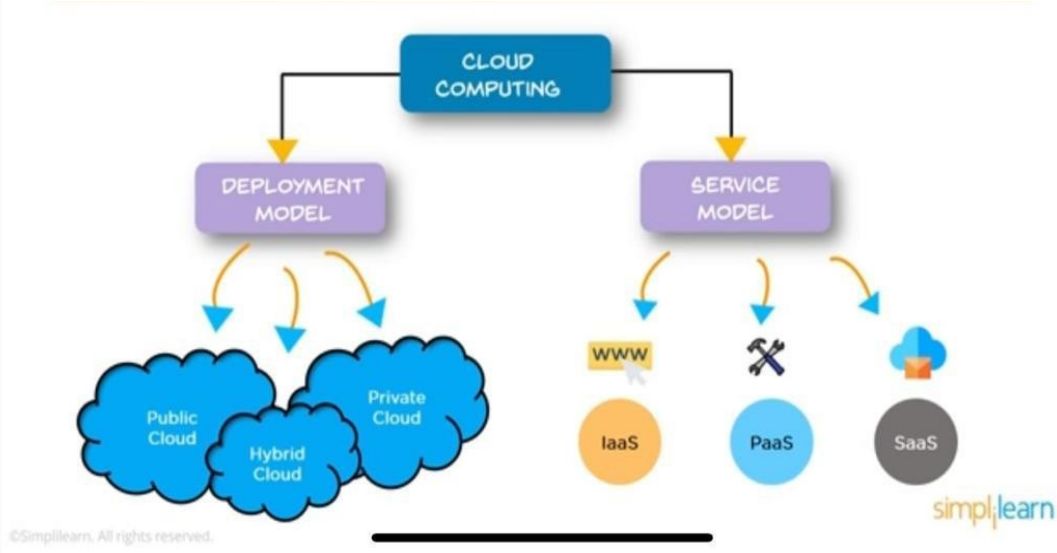
The scope of load balancing involves making sure that tasks or work are evenly spread across different parts of a system, like servers or computers. This helps in ensuring that no single part gets overwhelmed, leading to better performance, faster response times, and a more reliable operation of websites, applications, and computer systems. Load balancing is used in various areas, from websites and cloud computing to networks and large-scale data processing, to make sure resources are used efficiently and things run smoothly.

The scope of the Cat and Mouse Optimization Algorithm (CMOA) and the Grey Wolf Optimization Algorithm (GWO) is to find the best solutions to optimization problems. These algorithms are like smart search strategies inspired by nature.

INTRODUCTION

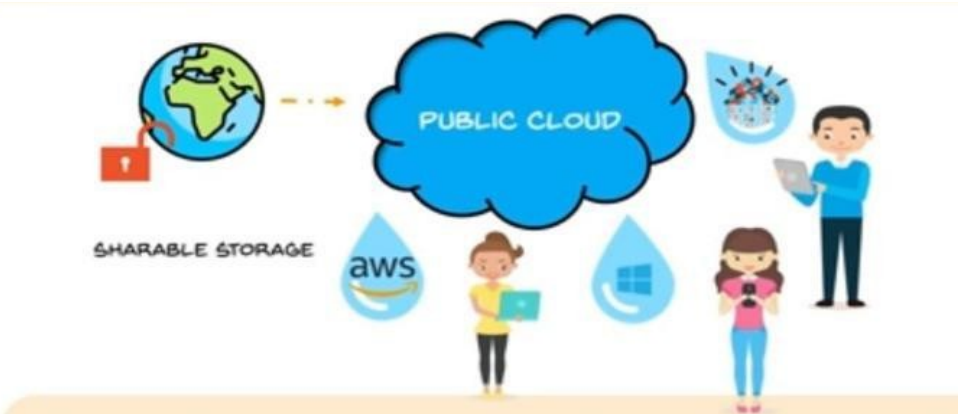
Cloud is nothing but internet. Cloud computing means performing any task/operation over the internet.According to definition proposed by Gartner in 2008 : A style of computing where massively scalable IT related capabilities are provided “as a service” using internet technologies to connect multiple external customers. Distributed computing is an attractive technology within the realm of software engineering. It encompasses a set of powerful capabilities that include online data storage, versatile frameworks, and advanced applications. Cloud computing is defined as “on demand resource pulling” and it works on pay-as-you-go pricing model .The central objective of cloud computing is to enhance the capabilities of a data center by providing dynamic and adaptable services to users in the cloud.

TYPES OF CLOUD COMPUTING



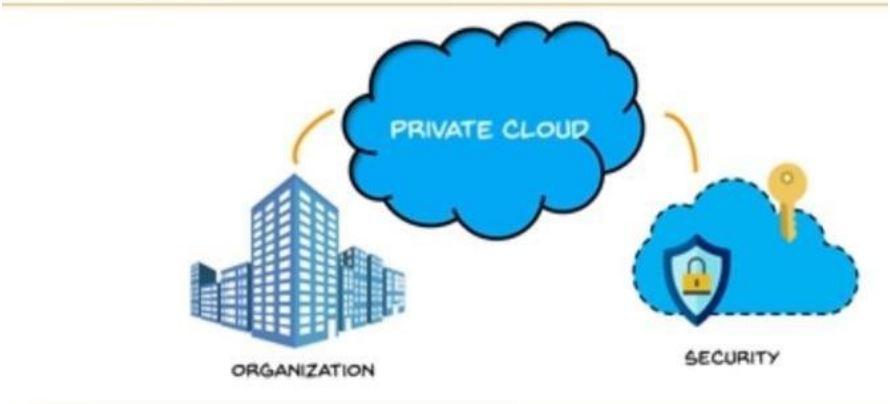
DEPLOYMENT MODELS:

Public Cloud:



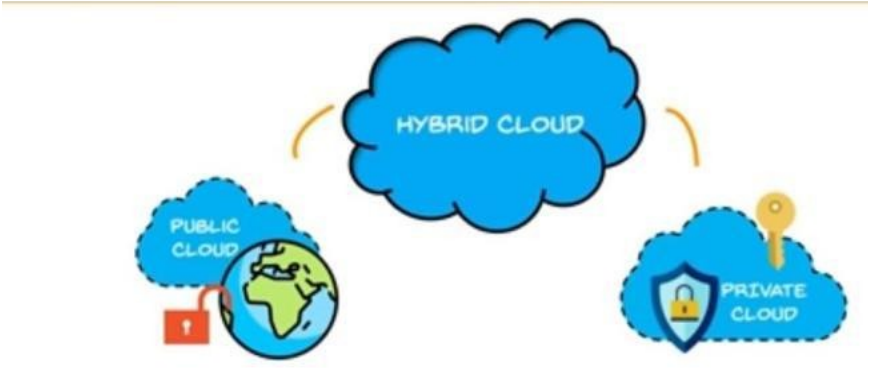
- The public cloud infrastructure is made available to general public over the internet and it is owned by a cloud provider.
- Example: AWS, Microsoft Azure, IBM’s Blue Cloud and sun cloud

Private Cloud:



- The Private Cloud infrastructure is exclusively operated by a single organization.
- It can be managed by the organization or a third party and may exist in on-premise or off-premise.
- Example:AWS, VMware

Hybrid Cloud:



- It consists the functionalities of both public and private cloud.
- For Example: Federal agencies opt for private clouds when sensitive information is involved. Also, they use the public cloud to share datasets with general public or other government departments.

COMMUNITY CLOUD

- A community cloud is part of private cloud.
- A collaborative,multi-tenant platforms used by several distinct organizations to share the same applications.
- Users are typically operating within the same industry or field.

Service Model:

The three primary service models in cloud computing are Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS), each catering to different levels of control, customization, and management responsibilities for users.

Infrastructure as a service(IaaS): Infrastructure as a Service (IaaS) provides virtualized computing resources, including servers, storage, and networking, allowing users to scale and manage infrastructure .

Platform as a service(PaaS): PaaS is a cloud service that provides platforms and runtime environments for developing ,testing and managing applications.

Software as a service(SaaS): Cloud providers host and manage the software applications on a pay as you go pricing model.

This paper addresses the process of distributing workloads efficiently among resources in the cloud. As the number of clients in the cloud grows, the responsibility of handling requests increases. In certain situations, some servers may be overwhelmed with requests, while others remain underutilized. This imbalance can result in server overloading or underloading. To tackle this issue, load balancing is introduced, aiming to evenly distribute incoming requests among servers to optimize resource usage and prevent performance disparities.

LOAD BALANCING

Load balancing is a process of equitably distributing workloads across all available nodes to achieve optimal resource utilization and enhance user satisfaction. This method involves dividing tasks among virtual machines to ensure an equal distribution of

work, promoting efficient use of resources, increasing throughput, and reducing response times. The primary goals include maximizing throughput, minimizing costs and energy consumption, optimizing resource utilization, and enhancing overall system performance. In the context of cloud computing, load balancing becomes a crucial strategy to manage application demands effectively by distributing resources among various nodes, ensuring system consistency, resilience, and protection against failures.

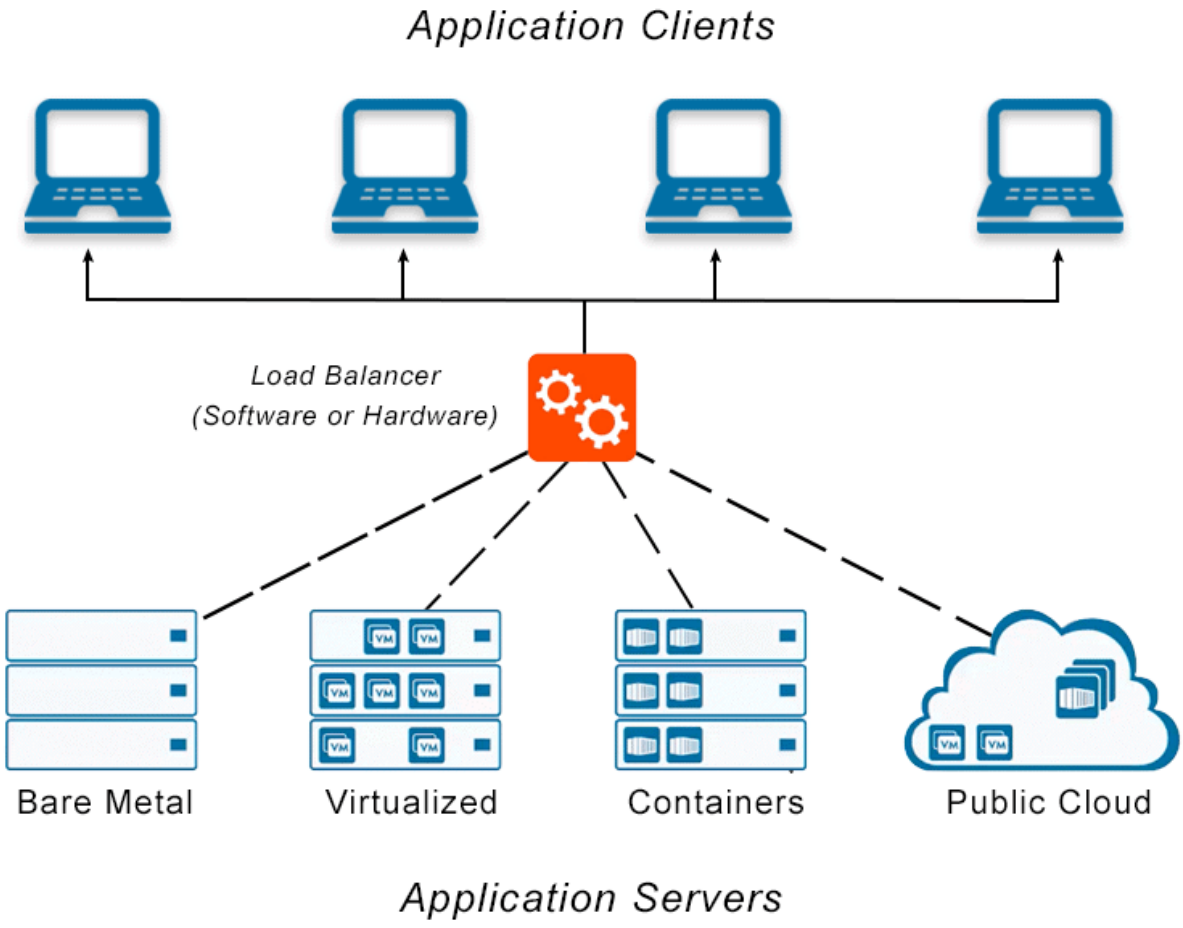


Figure1. Load Balancing

CHALLENGES OF LOAD BALANCING:

In the context of cloud computing, load balancing faces several challenges due to the dynamic and distributed nature of the environment. Here are the main challenges:

Dynamic Workload Variation: Cloud environments experience unpredictable changes in workload demand. Load balancers must dynamically adapt to varying traffic patterns and distribute workloads efficiently to handle fluctuations in demand.

Scalability: Load balancers need to scale seamlessly with the number of resources and adapt to the changing size of the infrastructure.

Security Concerns: Ensuring the security of data and applications while load balancing is critical. Load balancers need to implement secure communication protocols and avoid becoming a potential point of vulnerability in the system.

Cost Management: Efficient load balancing involves minimizing costs associated with data transfer, server usage, and overall infrastructure. Striking a balance between optimal resource utilization and cost-effectiveness is a significant challenge.

Complexity of algorithms: Algorithms in cloud computing ought to be simple to achieve.

GOALS OF LOAD BALANCING ALGORITHMS

1.Optimize Resource Utilization

2.Improve in system performance

3.Handle Workload Fluctuations

4.Ensure High Availability

5.Minimize Response Time

6.Cost Optimization

7.Adapt to Changing Conditions

8.Least reaction time

9.Improvement in unwavering quality

Table 1. CLASSIFICATION OF LOAD BALANCING ALGORITHMS

Algorithm	Objective	Performance	Advantages	Disadvantages
1.Round Robin Algorithm	Round Robin scheduling algorithm is to provide fair and equal allocation of CPU time among competing tasks in a cyclic manner.	1. Round Robin ensures that all machines are treated fairly by giving them equal opportunities to execute tasks in a cyclic manner. 2. This helps in preventing one machine from hogging all the resources and promotes a balanced workload.	1.Round Robin algorithm achieves fairness in workload distribution by cyclically assigning tasks to multiple servers. 2.preventing overload on a single server and ensuring balanced resource utilization. 3.The algorithm is simple to implement, making it easy to deploy in cloud computing environments	1.The algorithm does not consider the current load or capacity of each server, which can lead to situations where some servers are underloaded while others are overloaded. 2.In situations where tasks have dependencies, Round Robin may not be optimal. 3.The algorithm may not efficiently handle situations where tasks have uneven resource requirements,

				leading to potential inefficiencies in resource allocation.
Least Connections Algorithm	The primary goal of the Least Connections algorithm is to direct new requests to the server with the fewest active connections at the moment.	1. It is designed to distribute incoming network traffic or requests among a set of servers in a way that minimizes the number of active connections on each server. 2. This algorithm plays a crucial role in optimizing the usage of server resources and improving the overall performance and reliability of the system.	1. Least Connections ensures that requests are directed to servers based on their current load, leading to dynamic load distribution. 2. Servers with fewer active connections are assumed to have more available resources, promoting efficient resource utilization. 3. Adapts quickly to changes in server loads, redistributing requests to maintain a	1. Assumption of equal server capacities leads to uneven distribution and underutilization of more capable servers. 2. Least Connections does not consider the health or performance metrics of individual servers. 3. Server with few active connections may still be overwhelmed if those connections

			balanced distribution as the number of active connections fluctuates.	are resource-intensive.
3. Ant colony Optimization Algorithm Based Load Balancing Algorithm	Ant colony optimization is an optimization algorithm which employs the probabilistic technique and is used for solving computational problems and finding the optimal path with the help of graphs.	1. Ant Colony Optimization Based Load Balancing Algorithm is one of the Dynamic load balancing calculation. 2. Ant Colony Optimization (ACO) is to solve computational problems through the simulation of the foraging behavior of ants. 3. It is performed to track down the shortest and ideal path using graphs.	1. ACO is inherently decentralized, allowing for autonomous decision-making by individual agents (ants). This self-organizing nature makes it well-suited for distributed systems. 2. ACO can adapt to dynamic changes in the environment, such as varying workloads, server failures, or additions/removals of resources. 3.The	1. Finding optimal parameter values for different problem instances can be challenging. 2. ACO provides solutions that are generally good but does not guarantee optimality. 3. This algorithm doesn't explicitly incorporate measures to handle or recover from faults or failures in the system.

			pheromone updating mechanism helps the algorithm respond to changes.	
4.HoneyBee Foraging Algorithm	The main goal of the Honey Bee Foraging Load Balancing Algorithm is to make sure that tasks or jobs are shared efficiently across different parts of a computer system.	1.This algorithm balances the load of virtual machines when all the systems become unbalanced. 2.This algorithm helps distribute the workload, so no part of the system gets too busy, and everything gets done smoothly. 3.Prevent the overloading of specific computational nodes by distributing tasks .	1.This algorithm can handle shifts in the workload or the computer system without causing problems. 2.If one part stops working, the algorithm can quickly adapt and move tasks to other working parts, preventing the whole system from breaking down.	1.It might take a bit longer to figure out the best way to share tasks, unlike some faster methods. 2.It might not be the best choice for every computer setup; some systems might need different ways to share tasks effectively.
5.Machine learning based Load Balancing Algorithm	1. The primary objective of a machine learning-based load balancing	1. This algorithm is to achieve efficient utilization of resources, improve system performance.	1.This algorithm learns from past experiences to make wise decisions on	1.The effectiveness of machine learning algorithms heavily relies on

	algorithm in cloud computing is to optimize the distribution of tasks or workloads among various resources dynamically and intelligently.	2.This algorithm adapts to changing conditions within the cloud environment. 3.Essentially, it leverages machine learning techniques to learn patterns from historical data and make informed decisions on how to allocate tasks, ensuring a balanced and responsive cloud infrastructure.	how to share tasks, ensuring each part of the cloud does its fair share of work. 2.This algorithm makes sure each computer does just the right amount of work, preventing some from getting too busy and others from being idle.	the quality and representativeness of the training data. 2.If the training data is not diverse or does not accurately represent the real-world scenarios, the algorithm's performance may suffer.

CAT AND MOUSE BASED OPTIMIZATION ALGORITHM

The paper introduces a new optimization method called the Cat and Mouse-Based Optimizer(CMBO), inspired by how cats chase mice. CMBO simulates the movement of cats towards mice and the escape of mice to safety. The paper presents the mathematical model and implementation of CMBO for solving optimization problems. Evaluation on various types of objective functions demonstrates that CMBO performs well in solving different optimization problems. When compared to nine other popular algorithms, including Genetic Algorithm and Particle Swarm Optimization, CMBO stands out by providing more effective solutions closer to the best possible outcomes.

Working Model:

In this section, the theory of the Cat and Mouse Optimization Algorithm (CMBO) is stated, Then its mathematical model is presented in order to optimize various problems. The Cat and Mouse-Based Optimizer (CMBO) is a population-based algorithm inspired by how cats hunt mice in nature. In this algorithm, search agents are divided into two groups: cats and mice. These groups explore the problem search space using random movements. The algorithm updates its population in two phases. In the first phase, it models the movement of cats towards mice, simulating predatory behavior. In the second phase, models the escape of mice to havens to ensure their safety. This dual-phase approach is designed to enhance the algorithm's efficiency in solving optimization problems.

From a mathematical point of view,think of the population as a group of potential solutions to a problem. Each member in this group represents a proposed solution, like points on a map. The population matrix is like a table organizing these solutions: each row is an individual, and each column is a variable in the problem. Equation (1) mathematically describes how these individuals, with their chosen variable values, make up the population matrix. It's a way for the algorithm to systematically handle and evaluate these proposed solutions.

$$X = \begin{bmatrix} X_1 \\ \vdots \\ X_i \\ \vdots \\ X_N \end{bmatrix}_{N \times m} = \begin{bmatrix} X_{1,1} \cdots & X_{1,d} \cdots & X_{1,m} \\ \vdots \ddots & \vdots \iddots & \vdots \\ X_{i,1} \cdots & X_{i,d} \cdots & X_{i,m} \\ \vdots \iddots & \vdots \ddots & \vdots \\ X_{N,1} \cdots & X_{N,d} \cdots & X_{N,m} \end{bmatrix}_{N \times m} \quad (1)$$

where X is the population matrix of CMBO, X_i is the ith search agent, $x_{i,d}$, is the value for the dth problem variable obtained by the ith search agent, N is the number of population members, and m is the number of problem variables.

In simpler terms, each member of the population proposes a set of values for solving the problem. These proposed solutions are then evaluated using an objective function, creating a vector of scores (Equation 2). This vector helps identify which solutions are more effective in addressing the problem.

$$F = \begin{bmatrix} F_1 \\ \vdots \\ F_i \\ \vdots \\ F_N \end{bmatrix}_{Nx1} \quad (2)$$

where F is the vector of objective function values and F_i is the objective function value for the ith search agent. Simply put, after evaluating solutions with the objective function, the best solutions are ranked first, and the worst ones are ranked last. Equations (3) and (4) define how the population matrix and objective function are organized to identify and prioritize the most effective solutions.

$$X^s = \begin{bmatrix} X_1^s \\ \vdots \\ X_i^s \\ \vdots \\ X_N^s \end{bmatrix}_{N\,x\,m} = \begin{bmatrix} X_{1,1}^s \cdots & X_{1,d}^s \cdots & X_{1,m}^s \\ \vdots \ddots & \vdots \iddots & \vdots \\ X_{i,1}^s \cdots & X_{i,d}^s \cdots & X_{i,m}^s \\ \vdots \iddots & \vdots \ddots & \vdots \\ X_{N,1}^s \cdots & X_{N,d}^s \cdots & X_{N,m}^s \end{bmatrix}_{N\,x\,m} \quad (3)$$

$$F^s = \begin{bmatrix} F_1^s \; min(F) \\ \vdots \quad \vdots \\ F_N^s \; max(F) \end{bmatrix}_{N\,x\,1} \quad (4)$$

where $X^{\$}$ is the sorted population matrix based on objective function value, $X^{\$}_i$ is the ith member of sorted population matrix, $X^{\$}_{i,d}$, is the value for the dth problem variable obtained by the ith search agent of sorted population matrix, and $F^{\$}$ is the sorted vector of an objective function.

The Cat and Mouse-Based Optimizer (CMBO) divides its population matrix into two groups: mice and cats. It assumes that the top-performing half of the population, with better objective function values, forms the mouse group. The remaining lower-performing half constitutes the cat group, as defined in Equations (5) and (6). This separation helps focus on the more promising solutions for further optimization.

$$M = \begin{bmatrix} M_1 = X_1^s \\ \vdots \\ M_i = X_i^s \\ \vdots \\ M_{N_m} = X_{N_m}^s \end{bmatrix}_{N_m x\, m} = \begin{bmatrix} X_{1,1}^s & \cdots & X_{1,d}^s & \cdots & X_{1,m}^s \\ \vdots & \ddots & \vdots & \iddots & \vdots \\ X_{i,1}^s & \cdots & X_{i,d}^s & \cdots & X_{i,m}^s \\ \vdots & \iddots & \vdots & \ddots & \vdots \\ X_{N_m,1}^s & \cdots & X_{N_m,d}^s & \cdots & X_{N_m,m}^s \end{bmatrix}_{N_m X\, m} \quad (5)$$

$$C = \begin{bmatrix} C_1 = X_{N_m+1}^s \\ \vdots \\ C_j = X_{N_m+j}^s \\ \vdots \\ C_{N_m} = X_{N_m+N_c}^s \end{bmatrix}_{N_c x\, m} = \begin{bmatrix} X_{N_m+1,1}^s & \cdots & X_{N_m+1,d}^s & \cdots & X_{N_m+1,m}^s \\ \vdots & \ddots & \vdots & \iddots & \vdots \\ X_{N_m+j,1}^s & \cdots & X_{N_m+j,d}^s & \cdots & X_{N_m+j,m}^s \\ \vdots & \iddots & \vdots & \ddots & \vdots \\ X_{N_m+N_c,1}^s & \cdots & X_{N_m+N_c,d}^s & \cdots & X_{N_m+N_c,m}^s \end{bmatrix}_{N_c X\, m} \quad (6)$$

where M is the population matrix of mice, N_m number of mice, M_i is the jth mouse, C is the population matrix of cats, N c is the number of cats, and C j is the ith cat. CMBO updates cat positions towards mice by simulating their natural behavior, mathematically expressed through Equations (7)–(9).

$$C_j^{new} : C_{j,d}^{new} = C_{j,d} + rx\left(m_{k,d} - Ix\, C_{j,d}\right) \wedge j = 1:N_c, d = 1:m, kE\,1:N_{m,} \quad (7)$$

$$I = \text{round}(1+\text{rand}), \quad (8)$$

$$C = \left\{ C_j^{new}, \vee F_j^{c,new} < F_j^c, C_j \vee else \right\} \quad (9)$$

Here, C_j^{new} is the new status of the jth cat, $C_{j,d}^{new}$,is the new value for the dth problem variable obtained by the jth cat, r is a random number in interval [0,1], $m_{k,d}$ is the dth dimension of the kth mouse, $F_j^{c,new}$,is the objective function value based on new status of the jth cat.

In the second phase of CMBO, mice simulate escaping to random havens for safety, assumed to exist for each mouse. Havens' positions are randomly generated based on the patterns of algorithm members. Equations (10)–(12) mathematically represent this phase, updating the positions of mice in the search space.

$$H_i : h_{i,d} = x_{l,d} \;\&\; i = 1 : N_{m,d} = 1 : m,\ l \in 1 : N, \tag{10}$$

$$M_i^{new} : m_{i,d}{}^{new} = m_{i,d} + r \times (h_{i,d} - I \times m_{i,d}) \times sign(F_i^m - F_i^H)\ \&\ i = 1 : N_m,\ d = 1{:}m \tag{11}$$

$$M_i = \{M_i^{new},\ |\ F_i^{m,new} < F_i^m \qquad M_i,\ |else, \tag{12}$$

Here, H_i is the haven for the ith mouse and F_i^H is its objective function value. M_i^{new} is the new status of the ith mouse and $F_i^{m,new}$,is its objective function value.

After updating all algorithm members, CMBO proceeds to the next iteration, persisting until a specified stopping condition is met, which could be a set number of iterations, unacceptable error level, or a predefined time limit. Once the iterations conclude, the algorithm provides the best quasi-optimal solution for the given optimization problem. Flowcharts and pseudo code detailing CMBO's stages are depicted in Figure 1 and Pseudo code 1, respectively.

ALGORITHM :

The algorithm involves two main components:

1. **Cats (Predators):** Represent potential solutions or candidates in the search space.
2. **Mice (Prey):** Represent the optimal solution or the target that the algorithm is trying to find.

Start CMBO.

Input problem information: variables, objective function, and constraints.

Set number of search agents (N) and iterations (T).

Generate an initial population matrix at random.

Evaluate the objective function.

For t = 1:T

Sort population matrix based on objective function value using Equations (3) and (4).

Select population of mice M using Equation (5).

Select population of cats C using Equation (6).

Phase 1: update status of cats.

For j=1:N_c

Update status of the jth cat using Equations (7)–(9).

end

Phase 2: update status of mice.

For i = 1:N m

Create haven for the ith mouse using Equation (10).

Update status of the ith mouse using Equations (11) and (12).

end

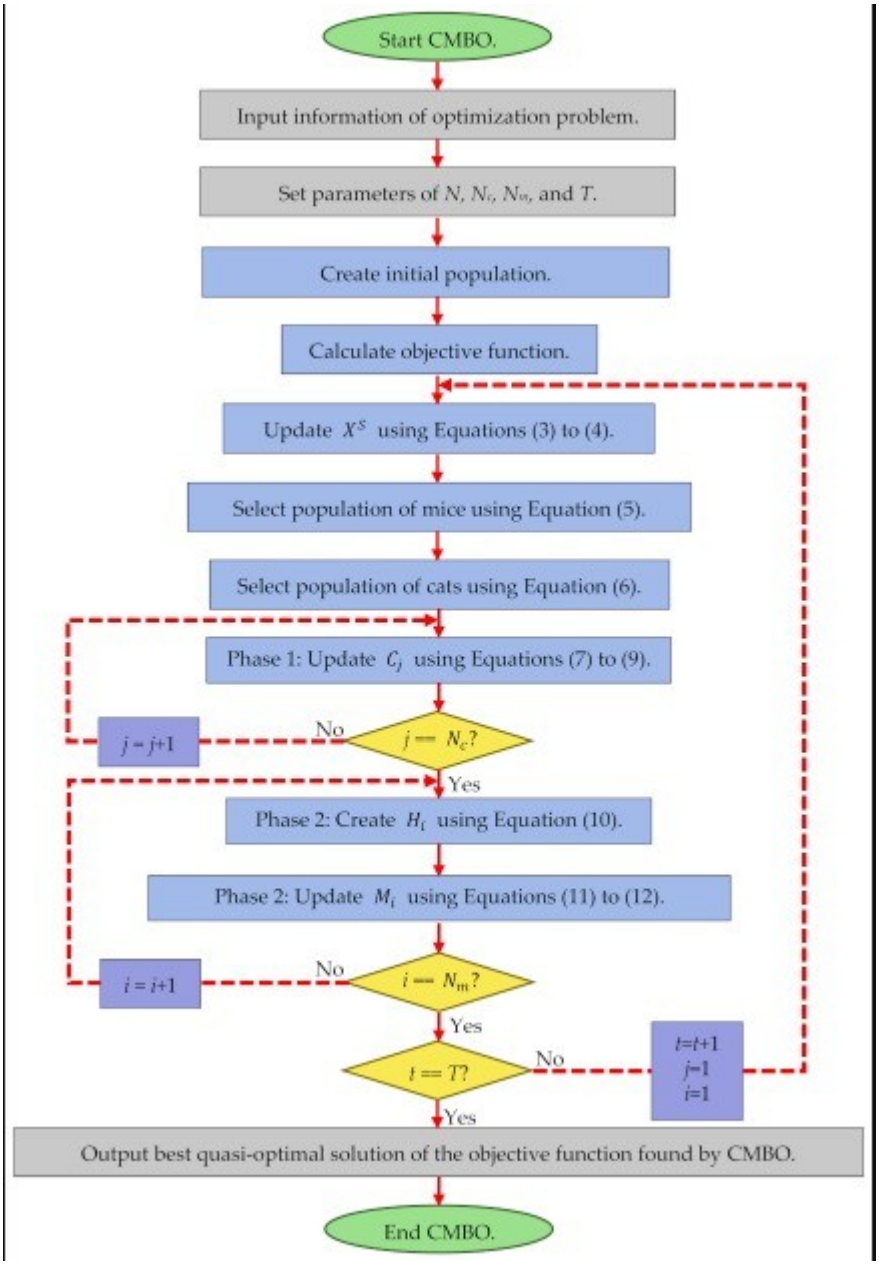
End

Output best quasi-optimal solution obtained with the CMBO.

End CMBO

The interaction between cats and mice is used to guide the search process towards the optimal solution. Cats try to catch mice, and mice try to escape from cats, creating a balance that helps explore the solution space effectively.

FLOWCHART



ADVANTAGES :

Efficient exploration and exploitation: CMBO balances exploration and exploitation by combining cat's search for prey and mouse's escape to havens.

Simple and transparent: The algorithm is easy to understand and implement, requiring minimal parameter tuning.

Flexibility and adaptability: CMBO can be easily adapted to solve various optimization problems in cloud computing with minor modifications.

Robustness and convergence: The algorithm exhibits good robustness against noise andconverges quickly towards optimal solutions.

Bio-inspired and relatable: The cat-and-mouse metaphor makes the algorithm intuitive and relatable, facilitating its understanding and application.

Parallelizable: CMBO can be easily parallelized for faster execution on multi-core and distributed computing platforms.

Memory efficient: The algorithm requires less memory compared to other population-based optimization algorithms.

Effective for uni modal problems: CMBO performs well in solving uni modal optimization problems with a single global optimum.

Promising results in cloud computing: CMBO has shown promising results in various cloud optimization tasks like resource allocation, task scheduling, and data placement.

Potential for further development and improvement: CMBO is a relatively new algorithm with the potential for further development and improvement to address its limitations.

Disadvantages:

Limited exploration: CMBO might get stuck in local optima due to its reliance on previous successes, reducing exploration potential.

Parameter tuning: Finding the optimal values for algorithm parameters can be challenging and time-consuming.

High computational cost: The iterative nature of the algorithm can be computationally expensive, especially for complex problems with large search spaces.

Premature convergence: Early convergence towards suboptimal solutions can occur if initial population diversity is insufficient.

Lack of theoretical foundation: Compared to other established algorithms, the theoretical basis of CMBO is still under development.

Limited applications: Although promising, CMBO's application in cloud computing remains limited compared to well-established algorithms.

Sensitive to initial population: The algorithm's performance heavily depends on the quality and diversity of the initial population. Black-box nature: The internal workings of the algorithm can be complex and difficult to interpret, hindering its widespread adoption.

Limited scalability: The performance of CMBO might degrade with increasing problem size and complexity.

Susceptible to manipulation: Malicious actors could potentially manipulate the algorithm to exploit vulnerabilities in the cloud system.

GREY WOLF ALGORITHM

The Grey Wolf Optimization (GWO) algorithm is a nature-inspired optimization technique based on the social structure and hunting behavior of grey wolves. Grey Wolf Optimization was proposed by Marajhali Mohammad and Lewis in 2014. In the context of cloud computing, GWO is employed to optimize various aspects of cloud systems, including resource allocation, task scheduling, energy efficiency, load balancing, and fault tolerance. We have different rank of wolves in pack using alpha, beta, delta, and omega wolves to represent potential solutions in the optimization process. We have alpha wolf that is the pack leader it can be

1.male wolf

2.female wolf.

4 Alpha wolf=Responsibility of alpha wolf is the season making is hunting,sleeping,eating, wake up and all important seasons are made by alpha wolf.

5 Beta Alpha=best candidate to be Alpha wolf.

6 Delta wolf=provide food to the pack and work for the pack in case of any danger.

7 Omega=caretaker,Older wolf,Scouts and the hunter.

By iteratively updating the positions of these wolves, GWO aims to converge toward optimal solutions, contributing to the efficient utilization of resources and improved overall performance in cloud-based environments.

Main steps of grey wolf hunting are:

1.Searching:searching for the pray

2.pursuing:tracking,chasing and approaching the pray

3.encircling and harassing the pray until it's moving

4.Attacking:attacking the prey.

Mathematical model of GWO algorithm

fittest solution=Alpha solution

Second solution=beta wolf

Third solution=Delta wolf

ALGORITHM:

1.Initialize grey wolf population

2.Initialize a,A and C.

3.calculate the fitness of each segment agent.

4.X_{alpha}=best search agent.

5.X_{beta}=second best search agent.

6.X_{gama}=third best search agent.

while(t<max number of iterations)

for each search agent

update the position of current search agent by above equations

end for

7.update a,A and C

8.calculate the fitness of all search agent

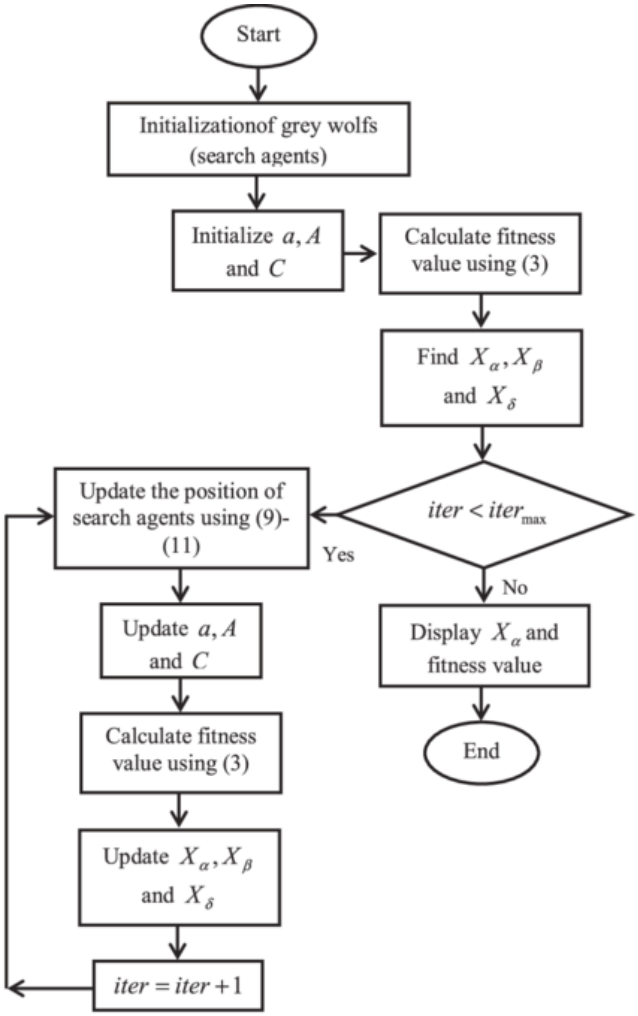
9.update $X_{alpha,}x_{beta,}X_{delta}$

10.t=t+1

11.end while

12.return x_{alpha}

FLOWCHART



WORKING MODEL OF GREY WOLF ALGORITHM:

The Grey Wolf Optimizer (GWO) is a nature-inspired optimization algorithm that simulates the social behavior of grey wolves. The algorithm was proposed by Mirjalili et al. in 2014. It is a heuristic optimization algorithm used for solving optimization problems, and it draws inspiration from the hunting and social hierarchy of grey wolves in the wild. Here's a detailed explanation of the GWO algorithm:

1) **Initialization:**

1) Define the objective function to be optimized.
2) Initialize the population of wolves. The position of each wolf represents a solution to the optimization problem.
3) Randomly initialize the positions of the alpha, beta, and delta wolves, which represent the top three solutions found so far. These wolves will guide the rest of the pack.

2) Objective Function Evaluation:

1) Evaluate the objective function for each wolf in the population.

3) Update Alpha, Beta, and Delta Positions:

1) Identify the alpha, beta, and delta wolves based on their fitness (objective function values). Alpha is the best, followed by beta and delta.
2) Update the positions of alpha, beta, and delta wolves using the following formulas: Dalpha=|2·rand()·alpha position−current position|
3) Dbeta=|2·rand()·beta position−current position|
4) Ddelta=|2·rand()·delta position−current position| where rand() is a random number between 0 and 1.

4) Update the Positions of the Rest of the Wolves:

1) Update the positions of the other wolves using the following formula: new position=alpha position−A·Dalpha where A is a coefficient that controls the exploration-exploitation trade-off. It is a decreasing function with time.

5) Boundary Handling:

1) If a wolf goes beyond the search space boundaries, it should be repositioned within the feasible region.

6) Update Coefficient A:

1) Update the coefficient A using a linearly decreasing function to balance exploration and exploitation.

7) Repeat:

1) Repeat steps 2-6 until a termination criterion is met (e.g., a maximum number of iterations or a satisfactory solution is found).

The GWO algorithm is designed to converge towards an optimal solution over successive iterations. It leverages the social hierarchy and collaboration among wolves to explore the solution space effectively. Adjusting parameters like the population size, the maximum number of iterations, and the coefficient A can impact the performance of the algorithm for specific problems.

Advantages:

Easy to Understand and Implement:

- The algorithm is straightforward and doesn't require complex mathematical understanding. Its simplicity makes it accessible for use without extensive expertise.

Global Optimization Capability:

- The algorithm is proficient at finding solutions close to the best possible outcome on a global scale, which is crucial for obtaining high-quality solutions across a wide range of problems.

Few Parameters to Adjust:

- Users don't need to tweak many parameters to make the algorithm work effectively, making it user-friendly and easy to apply to different scenarios.

Disadvantages:

Limited Robustness:

- The algorithm may not perform optimally in every scenario or problem. Its effectiveness can vary depending on the nature of the optimization problem, and it might not always outperform other algorithms.

Not Suitable for All Types of Problems:

- While versatile, the Grey Wolf Optimization Algorithm may not be the best choice for certain types of optimization problems, especially those with specific characteristics or constraints.

Lack of Consistency:

- The algorithm's performance may lack consistency across different types of optimization problems. It might excel in some scenarios but not perform as well in others.

When do we use the grey wolf algorithm?

The GWO is used to solve different problems such as global optimization problems, electric and power engineering problems, scheduling problems, power dispatch problems, control engineering problems, robotics and path planning problems, environmental planning problems and many others.

A Hybridized Approach: Cat-Wolf Optimization for Cloud Efficiency

Problem: Cloud resource management faces challenges like dynamic workloads, underutilized resources, and inefficient scheduling.

Solution: Hybrid Cat-Mouse and Grey Wolf Optimization (CMGWO) tackles these issues by blending local search (CMO) and global exploration (GWO).

Cat Phase: VMs act as "cats" chasing "mice" representing optimal resource allocation. Local search refines resource usage within clusters.

Wolf Phase: Packs of VMs ("wolves") mimic Grey Wolf hunting behavior. Alpha wolves (leaders) guide the pack towards efficient resource utilization across clusters.

Hybridization: A dynamic switch between Cat and Wolf phases balances exploration and exploitation, preventing stagnation and finding global resource optimums.

Scalability: CMGWO adapts to large cloud environments with efficient parallelization on multiple nodes.

Benefits: Improved resource utilization, reduced costs, enhanced service quality, and efficient workload scheduling.

Improved resource efficiency: Efficiently allocate resources and minimize wastage by finding near-optimal configurations.

Enhanced performance: Improve application performance by dynamically adjusting resource allocation based on workload demands.

Cost optimization: Reduce cloud computing costs by optimizing resource utilization and avoiding unnecessary spending.

Dynamic adaptation: Handle the dynamic nature of cloud environments by continuously adapting resource allocation to changing demands.

Applications: Large-scale data processing, scientific computing, and dynamic web hosting platforms.

Security: Address security concerns by implementing access control and resource isolation mechanisms.

Potential Impact: CMGWO can significantly improve cloud resource management, leading to cost savings, performance gains, and a more sustainable cloud ecosystem.

CONCLUSION

Cloud load balancing algorithms are critical for ensuring optimal performance and resource utilization in cloud environments. Overall,cloud load balancing algorithms are essential for building efficient,scalable and reliable cloud applications. By understanding the strengths,challenges and future trends we can choose the right algorithm to optimize our cloud infrastructure and maximize performance. CMGWO presents a promising hybrid approach for cloud resource optimization, offering benefits for both cloud providers and users. Hybrid CMOGWO has the potential to significantly improve resource optimization in cloud computing environments. However, careful implementation, parameter tuning, and problem-specific adaptation are crucial for achieving optimal results.

REFERENCES

1. P. Chawla and A. Kumar, “A Systematic literature review on load balancing algorithms of Virtual Machines in a Cloud computing environment,” International Conference on Innovative Computing and Communications (2020).

2. X. Fu, G. Xu and J. Pang, “A load balancing model based on cloud partitioning for the public cloud,” **Tsinghua Science and Technology**18, 34-39(2013).

3. R. Zaman Khan and M. Haris, “A Systematic Review on Load Balancing Issues in Cloud Computing,” Sustainable Communication Networks and Application, 297 - 303(2020).

4. P. Kumar and R. Kumar, ‘‘Issues and challenges of load balancing techniques in cloud computing: A survey,” ACM Computing Surveys 51, 1 – 35 (2019)

5. M. AsimShahid, N. Islam, M. Mansoor Alam, Mazliham and S. Musa, “A Comprehensive Study of Load Balancing Approaches in the Cloud Computing Environment and a Novel Fault Tolerance Approach,” **IEEE Access,** 8, 130500 - 130527 (2020).

6. Suman and Sangeeta, “Load Balancing In Cloud Computing: A Review,” International Journal of Advanced Research in Computer Science 15, 22 - 29(2018).

7**.** Mohammad Dehghani and Stepan Hubalovsky,”cat and mouse based optimizer:a new nature inspired optimization algorithm”.2021.

8. K. Vinod Kumar, R. Balakrishna and Santhosh Kumar,”Survey on various load balancing algorithms in cloud computing “AIP Conference Proceedings 2463, 020002 (2022).

9. Kethineni Vinod Kumar & A. Rajesh (2022): Multi-Objective Load Balancing in Cloud Computing: A Meta-Heuristic Approach, Cybernetics and Systems.

10. Rafi and K.Vinod Kumar,” Assessment of performance load balancing in the cloud using Optimization Techniques” 2023.