Virtual Machine Allocation in Cloud Computing Environments using Giant Trevally Optimizer

Hai-yu zhang*

School of Information, Shanxi College of Finance and Taxation, Taiyuan, 030024, China
School of Finance and Economics, Taiyuan University of Technology, Taiyuan 030024, China

***Abstract*—Cloud computing has gained prominence due to its potential for computational tasks, but the associated energy consumption and carbon emissions remain significant challenges. Allocating Virtual Machines (VMs) to Physical Machines (PMs) in cloud data centers, a known NP-hard problem, offers an avenue for enhancing energy efficiency. This paper presents an energy-conscious optimization approach utilizing the Giant Trevally Optimizer (GTO) which is inspired by the hunting strategies of the giant trevally, a proficient marine predator. Our study mathematically models the trevally's hunting behavior when targeting seabirds. The trevally's approach involves strategic selection of optimal hunting locations based on food availability, including pursuing seabird prey in the air or seizing it from the water's surface. Through extensive simulations, our method demonstrates superior performance in terms of skewness, CPU utilization, memory utilization, and overall resource allocation efficiency. This research offers a promising avenue for addressing the energy consumption challenges in cloud data centers while optimizing resource utilization for sustainable and cost-effective cloud operations.**

Keywords—Cloud computing; resource allocation; virtualization; Giant Trevally Optimizer

I. Introduction

The flexibility of cloud computing enables the provision of infrastructure, platforms, and software services. It has gained increasing popularity in private and public institutions due to its pay-per-use pricing scheme [1]. Cloud computing offers numerous advantages, such as scalability, flexibility, and cost efficiency. However, one pressing issue associated with cloud computing is its significant energy consumption [2]. Cloud data centers, which host the infrastructure and servers powering the services, consume a substantial amount of energy to handle computing tasks and store vast amounts of data. This energy-intensive operation contributes to environmental concerns, including carbon emissions and strain on power grids [3, 4]. To mitigate this problem, efforts are underway to develop energy-efficient practices such as server consolidation, virtualization, and green data center designs. By addressing the energy consumption challenge, cloud computing can become more sustainable and cost-effective while minimizing environmental impact [5]. According to the 2020, state of the data center report, data centers exhibit rack densities of 8.2 kW, with the potential to achieve 43 kW per rack through the implementation of effective water-cooling methods [6]. In the United States alone, data centers consume an estimated 140 billion kWh of energy annually [7]. In contrast, the global energy consumption of data centers is projected to range from 200 TWh to 500 TWh [8], accounting for approximately 1% of global electricity consumption. Predictions from [9] indicate that by 2030, data centers are expected to consume 3-13% of the world's electricity [10].

The convergence of Internet of Things (IoT), smart grids, meta-heuristic algorithms, machine learning, Artificial Intelligence (AI), association rule mining, and urban public transportation plays a pivotal role in revolutionizing the landscape of cloud computing. IoT sensors and devices generate an unprecedented volume of data, which smart grids harness to optimize energy distribution [11-13]. Meta-heuristic algorithms are essential for efficiently allocating resources in cloud data centers to manage this influx of data [14, 15]. Machine learning and AI algorithms analyze this data, predicting energy demands and enabling proactive resource allocation in cloud infrastructure [16-18]. Additionally, association rule mining identifies patterns and correlations within IoT-generated data, aiding in predictive maintenance and energy optimization [19]. Urban public transportation systems leverage IoT for real-time data collection and route optimization. Cloud computing serves as the backbone for processing, analyzing, and delivering information to commuters and traffic management systems, enhancing urban mobility [20].

Inefficient utilization of computing resources in cloud data centers is a significant concern that leads to excessive energy consumption. Despite the growing demand for cloud services, many data centers operate at low resource utilization levels, with an average utilization of less than 30%. This inefficiency leads to a significant amount of energy being consumed by idle nodes, accounting for more than 70% of the peak energy consumption [21]. This wastage of energy results in increased ownership costs and reduced returns on investments in cloud infrastructure. Cloud service providers increasingly recognize the importance of enhancing energy efficiency in their data centers. They are actively seeking strategies to optimize resource utilization and minimize energy wastage in order to meet the growing demand for sustainable operations. By implementing energy-efficient practices and optimizing resource allocation, they aim to achieve a more sustainable and cost-effective operation while meeting the increasing demands of cloud services [22].

This paper introduces a novel strategy for cloud computing resource allocation based on the Giant Trevally Optimizer (GTO). By adopting a cloud-based model, data can be processed, recorded, and retrieved simultaneously, ensuring efficient resource allocation. This approach optimizes resource allocation by considering the user's request while maintaining system performance. Task assignment to virtual machines (VMs) is primarily determined by factors such as cost, deadline, and runtime. The structure of the paper is outlined as follows: Section II offers a detailed review of existing cloud resource allocation techniques. Section III elaborates on the proposed algorithm, providing details on its methodology. Section IV presents the experimental results obtained by implementing the algorithm. Finally, Section V provides a comprehensive summary of the paper and offers suggestions for future research directions in the field of cloud resource allocation.

II. Related Work

Hanini, et al. [23] introduced a novel approach that combines a virtual machine utilization scheme with a mechanism to regulate access to the virtual machine monitor for incoming requests. The number of active virtual machines is determined based on the workload, while the access control is determined by the number of requests. A mathematical model is utilized to describe the studied process and parameter values, and a power consumption model is developed and assessed. The evaluation of the proposed mechanism includes the use of numerical data to assess the quality of service (QoS) parameters. Additionally, the impact of the method on energy consumption behavior is thoroughly analyzed. The results of this analysis indicate a positive and beneficial influence of the proposed mechanism. Cloud computing brings forth various valuable services but also introduces security concerns related to user information privacy and the optimization of virtual machine allocation to enhance resource utilization. Dubey and Sharma [24] aim to address these challenges by developing a secure VM allocation algorithm based on an extended version of the Intelligent Water Drop (IWD) algorithm, which leverages natural phenomena. The implementation of their proposed algorithm was conducted using the CloudSim simulation toolkit. To evaluate its effectiveness, a comparison was performed against established VM allocation policies in the field of cloud computing. The experimental results from the simulations demonstrated that the proposed VM allocation policy outperformed existing approaches.

Samriya, et al. [25] have introduced a novel algorithm called the multi-objective Emperor Penguin Optimization (EPO) algorithm to optimize the allocation of virtual machines in a heterogeneous cloud environment, focusing on resource utilization. The proposed approach incorporates elements from the Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), and Binary Gravity Search (BGS) algorithms to ensure its suitability for virtual machines in data centers. A comprehensive evaluation of the proposed system was conducted using a JAVA simulation platform, which demonstrated its energy efficiency and significant advantages compared to other strategies. The results revealed that the EPO-based system effectively reduces energy consumption, minimizes SLA violations, and enhances QoS requirements, thereby providing a capable cloud service.

Devi and Kumar [26] have introduced a new VM allocation approach that effectively addresses SLA violation concerns and optimally allocates VMs to the most suitable hosts using the Improvised Grey Wolf Optimization (IGWO) algorithm. It considers various host characteristics, including CPU utilization and power consumption, to determine the most appropriate hosts for VM allocation. Additionally, the host's unused CPU and RAM resources are evaluated to maximize resource utilization. The experimental evaluation involved a random dataset with different virtual machines, and the proposed method was evaluated in comparison with existing methods such as ACO and Power-Aware Best Fit Decreasing (PABFD). The results demonstrate that the approach significantly minimizes the number of VM migrations, reduces SLA violations, and improves energy consumption. Consequently, the proposed VM allocation method promotes a green computing environment by consuming less power and maintaining a higher level of SLA compliance.

Xing, et al. [27] have formulated a VM allocation problem that aims to minimize the network bandwidth resources consumed by VMs and the total amount of power consumed by Physical Machines (PMs). To tackle this challenge, they propose the energy- and traffic-aware ACO (ETA-ACO) algorithm, which incorporates three innovative strategies for improved performance. The first strategy involves a two-step PM selection process that prioritizes PMs with lower power consumption and selects PMs that consume the least bandwidth. In the second strategy, VMs are arranged in descending order according to traffic demand. The third strategy generates a new solution by distributing components of optimal solutions across multiple solutions. Simulation outcomes validate the effectiveness of these three strategies in adapting ETA-ACO to the VM allocation problem. Addressing the challenging and critical issue of VM allocation for highly reliable cloud applications, Sheeba and Uma Maheswari [28] propose an improved Firefly algorithm-based approach. A K-means clustering algorithm is used to reduce migration time. Moreover, for optimal cluster selection in VM placement, adaptive PSO with the coyote optimization algorithm is applied. The suggested method is evaluated by examining the number of VMs, packet size, execution time, and transmission overhead. Under different constraints, the proposed method achieves improved performance and an optimal virtual machine placement scheme.

We have selected GTO as the basis for our research into VM allocation within cloud computing environments due to its unique and promising attributes that set it apart from other optimization algorithms. GTO draws inspiration from the hunting tactics of the giant trevally, a natural predator known for its exceptional hunting prowess in targeting seabirds and other prey. This distinctive approach to optimization allows us to model the allocation of VMs with a fresh perspective. The key benefits of GTO lie in its ability to effectively navigate complex optimization spaces, adapt to dynamic resource allocation scenarios, and converge towards superior solutions. Unlike conventional algorithms, GTO excels in its capacity to strategically select the optimal allocation locations for VMs based on factors such as food availability, mirroring the trevally's hunting strategy. Furthermore, GTO dynamically adapts to its pursuit, seizing opportunities whether they arise in the air or near the water's surface, mirroring the trevally's agile tactics. This adaptability makes it exceptionally well-suited to the inherently dynamic and multifaceted challenges of VM allocation in cloud data centers. Moreover, GTO offers the advantage of enhanced exploration and exploitation capabilities, striking a delicate balance between exploiting known promising solutions and exploring new allocation possibilities.

III. Energy-Aware VM Allocation Approach

A virtualization strategy based on the GTO is discussed in this section. Resource allocation in cloud environments depends on the architecture of the system, which allows different methods of access to the resources. Datacenter infrastructure can be provisioned by using a variety of methods and schemes. Fig. 1 illustrates the suggested architecture for energy-efficient resource allocation, comprising three fundamental elements: service providers, users, and data center resource management. Users submit their requests to the cloud service provider first, and then the broker returns a response based on the user's requirements, the date line, and the operation of the resource services. The Cloud Information System (CIS) resource manager reviews the broker's request as soon as it reaches the data center, assesses its suitability, and makes the appropriate decision. Requests are accepted by CIS based on the availability of the system and are passed on to the allocation scheme to determine the global optimal solution. GTO is responsible for the initial placement of VMs as well as monitoring the solution.

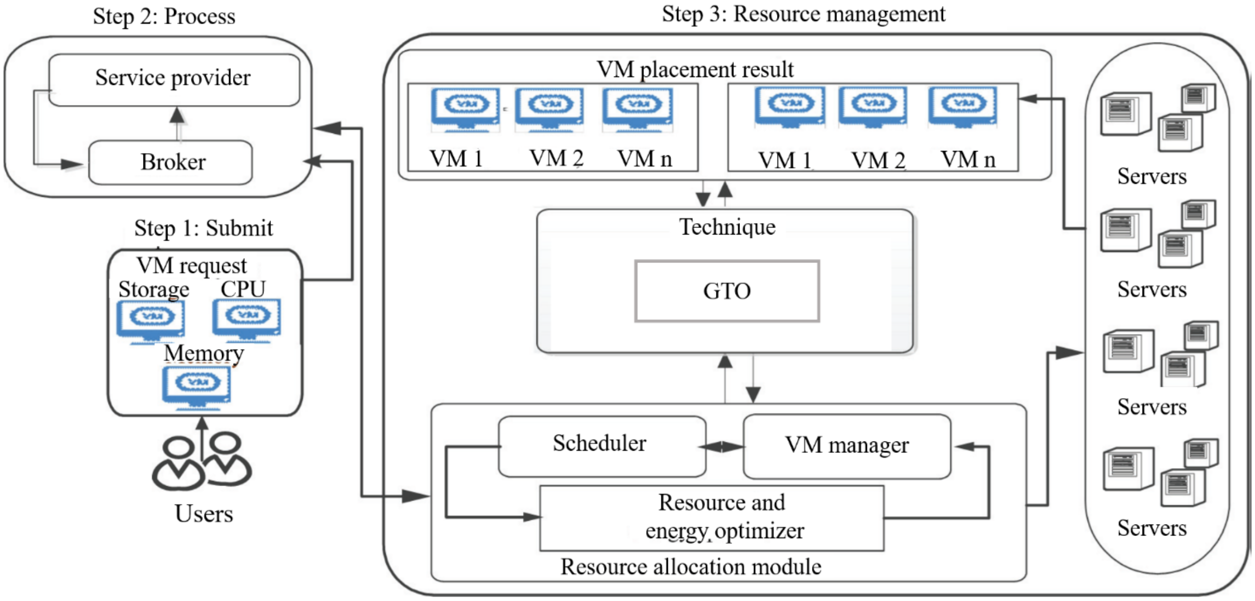

Fig. 1. Resource allocation model.

A. Giant Trevally Optimizer

GTO draws inspiration from nature, mimicking the behavior and strategies of giant trevallies in their pursuit of seabirds. The giant trevally belongs to the Jack family of marine predators. It is also known as the giant kingfish. The giant trevally, known as a dominant predator in its habitats, employs sophisticated hunting techniques that demonstrate its intelligence and adaptability. The giant trevally exhibits a hunting behavior that can be observed both in solitary individuals and in coordinated group efforts. It is most effective for predators to capture schooled prey when they are grouped. In a group or school, the leader, or first predator, is the most effective at capturing prey. When hunting, the giant trevally employs a remarkable strategy where it launches itself out of the water to surprise and capture its prey, often targeting seabirds.

Similarly, to other population-based meta-heuristic algorithms, GTO generates random initialization solutions termed giant trevallies. A potential or candidate solution to an optimization problem is represented by each giant trevally. These vectors, seen from a mathematical perspective as members of a population, make up the algorithm's population matrix [29]. Eq. (1) is used to model the GTO population members.

$$X = \begin{bmatrix} X_1 \\ \vdots \\ X_i \\ \vdots \\ X_N \end{bmatrix} = \begin{bmatrix} x_{1,1} \cdots x_{1,j} \cdots x_{1,Dim} \\ \vdots \quad \ddots \quad \vdots \quad \iddots \quad \vdots \\ x_{i,1} \cdots x_{i,j} \cdots x_{i,Dim} \\ \vdots \quad \iddots \quad \vdots \quad \ddots \quad \vdots \\ x_{N,1} \cdots x_{N,j} \cdots x_{N,Dim} \end{bmatrix} N \times Dim \quad (1)$$

where, X_i represents the i^{th} candidate solution of GTO, N denotes the number of GTO members, Dim denotes the number of decision parameters and $x_{i,j}$ indicates the value of the j^{th} variable provided by the i^{th} candidate solution. When the population's size and dimensions are determined, they will not change during the experiment. Eq. (1), as originally presented, continues to serve as the foundational model for representing the GTO population members. It encapsulates the critical elements of the algorithm, wherein each giant trevally, symbolizing a potential solution, contributes to the algorithm's population matrix. Eq. (1) remains constant and integral throughout the GTO process. Every trevally in the solution space of the problem is assigned a random position prior to its operation. All feasible regions must be covered by this random assignment in the $N \times Dim$ search space, as indicated in Eq. (2).

$$X_{i,j} = Minimum_j + (Maximum_j - Minimum_j) \times R \quad (2)$$

where, *R* represents a random number between 0 and 1, *Minimum$_j$* and *Maximum$_j$* indicate the limits of the described problem for the j^{th} dimension, i.e., the minimum and maximum values of population members. Each member of the GTO population is a potential solution to the VM allocation problem. Consequently, each candidate solution can be evaluated in terms of its objective function. In accordance with Eq. (3), these values are represented by a vector:

$$F = \begin{bmatrix} F_1 \\ \vdots \\ F_i \\ \vdots \\ F_N \end{bmatrix} = \begin{bmatrix} F(X_1) \\ \vdots \\ F(X_i) \\ \vdots \\ F(X_N) \end{bmatrix} N \times 1 \quad (3)$$

where, F_i refers to the i^{th} member's value of the objective function, as well as *F* represents the vector that contains these values.

The GTO algorithm simulates the giant trevallies' behavior while hunting for seabirds. To calculate the optimal optimization procedure of the suggested GTO algorithm, three steps are required: extensive search using Levy flight, choosing the hunting area, as well as jumping out of the water to chase and attack prey. The first and second steps represent the exploration phase of the GTO, as well as the third one represents the GTO's exploitation phase. Due to their nature, giant trevallies can travel long distances in search of food. Therefore, Eq. (4) is used in this step to simulate the foraging movements of giant trevallies.

$$X(t+1) = Best_P \times R + ((Maximum - Minimum) \times R + Minimum) \times Levy(Dim) \quad (4)$$

where *X(t+1)* denotes the position vector of the next-iteration giant trevally, *Best$_P$* signifies giant trevallies' current search space determined by their best position, *R* refers to a random number ranging from 0 to 1, *Levy(Dim)* stands for the Levy flight, a non-Gaussian stochastic process whose step sizes follow the Levy distribution. The algorithm is able to perform a global search due to its occasional large steps. Moreover, the levy flight increases the diversity of the population, prevents premature convergence, and enhances the ability to jump out of local optimal solutions. The recent literature has demonstrated that many animals, including marine predators, exhibit the behavior of Levy flight. Eq. (5) is used to calculate the levy (Dim).

$$Levy(Dim) = step \times \frac{u \times \sigma}{|\upsilon|^{1/\beta}} \quad (5)$$

where *step* refers to the step size, set to 0.01 in this case, β represents the Levy flight distribution index, a variable ranging from 0 to 2, set to 1.5 in this study, u as well as υ correspond to random numbers normally was distributed between 0 and 1. σ is derived from Eq. (6).

$$\sigma = \left(\frac{\Gamma(1+\beta) \times sine(\frac{\pi\beta}{2})}{\Gamma(\frac{1+\beta}{2}) \times \beta \times 2^{(\frac{\beta-1}{2})}} \right) \quad (6)$$

Giant trevallies determine and choose the best hunting area based on the number of food (seabirds) present in the chosen search space. This behavior is mathematically simulated by Eq. (7).

$$X(t+1) = Best_P \times \mathrm{A} \times \mathcal{R} + Mean_{Info} - Xi(t) \times \mathcal{R} \quad (7)$$

where *A* refers to a parameter that controls position change in the range of 0.3 and 0.4, *Xi(t)* indicates the location of the i^{th} giant trevally in a given frame of time *t* (at the present iteration). *Mean_Info* confirms that all of the information from the previous points has been utilized by these giant trevallies and is determined by Eq. (8).

$$Mean_{Info} = \frac{1}{N} \sum_{i=1}^{N} Xi(t) \quad (8)$$

Trevally starts chasing its prey during the attacking the GTO's phase. At this point, the trevally attacks the bird by jumping out of the water and catching it. During chasing and attacking prey, GTO presumed that giant trevallies experience visual distortion, which is primarily caused by the refraction of light. Refraction of light occurs as light travels from one material to another, where its direction changes at the interface. As depicted in Fig. 2, the light from point *A* in the first medium enters the second medium at the intersection point *S*. Hence refraction occurs and arrives at point *B* at the end of the process. The light bends toward the normal as it enters the denser medium as light travels from a rare medium, like air, to a denser medium, like water. There must be an angle between the incident and refracted rays at the point of refraction. Light rays are also affected by the medium in which they are traveling. Snell's law clarifies this connection using refractive indices, fixed values for certain media.

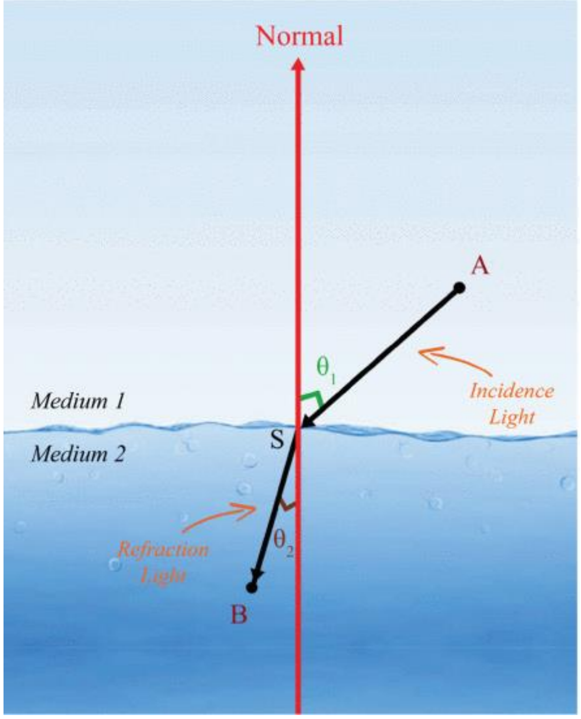

Fig. 2. Refraction of light principle.

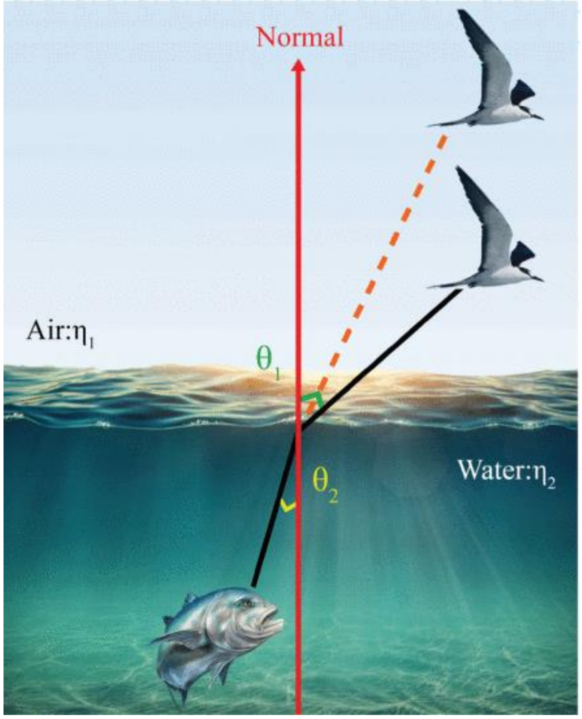

Fig. 3. Visual distortion in GTO.

As shown in Fig. 3, the giant trevally acts as an observer, as well as the bird behaves as an object. Due to the refraction of light, birds appear taller than their actual height, as indicated by the dashed line.

The relationship between the angle of incidence and the angle of refraction can be predicted using Snell's law. If we know the angle of incidence, we can determine the angle of refraction and vice versa. This relationship is demonstrated by Eq. (9), which represents Snell's law.

$$\eta_1 sin\theta_1 = \eta_2 sin\theta_2 \tag{9}$$

where, $\eta_1=1.0002$ and $\eta_2=1.3$ represent air's and water's absolute refractive indices, respectively. θ_1 and θ_2 refer to angles of incidence and refraction, respectively. θ_2 denotes a random number between 0 and 360, derived from Eq. (10).

$$sin\theta_1 = \frac{\eta_2}{\eta_1} sin\theta_2 \tag{10}$$

Eq. (11) is used to calculate the visual distortion.

$$\mathcal{V} = \sin(\theta_1^{\circ}) \times \mathcal{D} \tag{11}$$

where, *sin* stands for the sine of a variable in degrees, and *D* refers to prey-attacker distance, determined by Eq. (12).

$$\mathcal{D} = |(Best_P - Xi(t))| \tag{12}$$

where, $Best_P$ indicates the best solution gained so far, representing the prey's location. Eq. (13) is then used to simulate giant trevally behavior during jumping as well as chasing.

$$X(t+1) = \mathcal{L} + \mathcal{V} + \mathcal{H} \tag{13}$$

where, $\mathcal{L}$ is the launch speed for simulating the pursuit of the bird, as determined by Eq. (14), and $\mathcal{H}$ is the jump slope function used by the algorithm for the adaptive transition from exploration to exploitation, derived from Eq. (15).

$$\mathcal{L} = Xi(t) \times \sin(\theta_1^{\circ}) \times F\_obj(Xi(t)) \tag{14}$$

$$\mathcal{H} = \mathcal{R} \times (2 - t \times \frac{2}{\mathrm{T}}) \tag{15}$$

In Eq. (15), *R* stands for a random number used to denote the various motion senses of the giant trevally during the exploitation step, *t* signifies the current iteration, and *T* refers to the maximum number of iterations.

B. User Request Model

Users request resources, commonly referred to as VMs, from the data center via a broker or cloud provider. Each resource (VM) consists of a variety of components coordinated to fulfill a certain function. UR stands for users' requests. It is possible for users to submit multiple UR requests at the same time, which are executed on a First-Come-First-Served (FCFS) basis. VMs encompass three categories of resources: storage, memory, and CPU. *i* and *s* indicate the number of resources and their measuring capacities. Eq. (16) can be used to express the request mathematically.

$$A_i \subset UR \; and \; a_s^1, \beta_s^1, \gamma_s^1 \subset A_i$$

$$a_s^1, \beta_s^1, \gamma_s^1 \subset A_i \subset UR \Rightarrow a_s^1, \beta_s^1, \gamma_s^1 \subset UR \tag{16}$$

Eq. (17) and Eq. (18) will be used to express the request for a single resource in this case.

$$UR^1 = A_i \tag{17}$$

$$A_i = (a_s^1, \beta_s^1, \gamma_s^1) \tag{18}$$

where, *i* represents the number of resources required, when a user submits multiple requests, they are represented by Eq. (19) and Eq. (20).

$$UR^n = \sum_{i=1}^{n} = A_i = A_1 + A_2 + A_3 + \cdots A_n$$

$$= (a_s^1, \beta_s^1, \gamma_s^1) + (a_s^2, \beta_s^2, \gamma_s^2) + \cdots + (a_s^n, \beta_s^n, \gamma_s^n) \tag{19}$$

$$UR^n = \sum_{i=1}^{n}(a_s^i) + \sum_{i=1}^{n}(\beta_s^i) + \sum_{i=1}^{n}(\gamma_s^i) \tag{20}$$

C. Resource Utilization and Energy Model

CPU and memory utilization are calculated using Eq. (21) and Eq. (22), where *i* represents the number of tasks assigned to *n* VMs. rpu_{ijk} and rmu_{ijk} represent the CPU and memory utilization of k tasks running on *j* VMs on the i^{th} node, respectively.

$$RPU_i = \sum_{j=1}^{n} \sum_{k=1}^{l} rpu_{ijk} \tag{21}$$

$$RMU_i = \sum_{j=1}^{n} \sum_{k=1}^{l} rmu_{ijk} \tag{22}$$

The power consumption of the i^{th} PM in terms of memory and CPU utilization can be calculated using Eq. (23), where *t* is the unit of time and *C* is the number of memory units.

$$PC_i = \frac{(RPU_i)(RMU_i)}{C} \times t \tag{23}$$

Another objective of this research is to optimize the time required to assign VMS to relevant hosts. Allocation operations are influenced by the capacity of the hosts. A numerically generated data set is generated for each source between 0.1 and 10 milliseconds. The total allocation time is calculated by adding the CPU time associated with each host using Eq. (24).

$$Time = \sum_{i=1}^{n} T_i \tag{24}$$

D. step-by-step algorithmic explanation

The proposed VM allocation approach follows the following steps:

- Initialization: Initialize the GTO algorithm by generating a population of random solutions, referred to as "giant trevallies." Each giant trevally represents a potential solution to the VM allocation problem. Define parameters: N (number of giant trevallies), Dim (number of decision parameters), and set the population size and dimensions.
- Random position assignment: Assign each giant trevally a random position within the feasible search space of the problem, ensuring coverage across the N×Dim search space.
- Objective function evaluation: Evaluate the objective function for each giant trevally, representing the quality of their respective VM allocation solutions. This result in a vector F containing these objective function values.
- Exploration phase: Simulate the exploration behavior of giant trevallies by employing Levy flights. This phase allows for extensive search and occasional large steps. Update the position of each giant trevally using Eq. (4), where Levy flight is used to determine the next iteration's position.
- Choosing the hunting area: Giant trevallies select their hunting areas based on the number of seabirds (food) in those areas. This is simulated using Eq. (7), which determines the new position based on a combination of the best search space and previous positions.
- Chasing and attacking prey (exploitation phase): During this phase, giant trevallies pursue and attack prey, simulating their behavior when capturing seabirds. Visual distortion is considered due to the refraction of light, which affects the perceived size of prey. This is calculated using Snell's law Eq. (9) to determine the angle of refraction. The position update during the chase is determined by Eq. (13), where L represents the launch speed, ν is the visual distortion, and H is the jump slope function.
- Iteration and convergence: Repeat the above steps for a specified number of iterations or until convergence criteria are met (as defined by T, the maximum number of iterations).

IV. Experimental Results

In this section, we conduct a comparison between the performance of our proposed resource allocation algorithm and previous approaches. Additionally, we perform several experiments to evaluate the effectiveness of our algorithm. The suggested algorithm is implemented and simulated using Matlab simulator 2016b. To assess the effectiveness of the optimization algorithm, we utilize key performance indicators such as skewness, CPU utilization, memory utilization, and resource utilization. These metrics allow us to quantitatively evaluate the efficiency of our algorithm and make comparisons with other algorithms.

- Skewness: Skewness measures the asymmetry or unevenness in a probability distribution. It provides an indication of the uneven utilization of multiple resources on a server. The concept of skewness is derived from the observation that if a PM runs numerous memory-intensive virtual machines with a light load, resources may be lost due to insufficient memory to accommodate an additional virtual machine. Skewness quantifies the unevenness in resource utilization across a server by applying Eq. (25). Here, *R* represents the resource utilization of the n^{th} virtual machine, and *A* represents the average resource utilization.

$$W = (\frac{R_n}{A} - 1)^2 \tag{25}$$

- CPU utilization: This metric represents the average amount of CPU consumed by all servers while handling user requests. It is computed using Eq. (26), where H_i denotes the total number of available CPU resources and E_i represents the CPU resources requested for task execution.

$$C = \sum_{i=1}^{y} \frac{E_i}{H_i} \tag{26}$$

- Memory utilization: Memory utilization refers to the fraction of the memory resource that is used over time for processing all submitted tasks. It is calculated using Eq. (27), where v_i represents the total available memory and u_i indicates the memory requested for task execution.

$$M = \sum_{i=1}^{y} \frac{u_i}{v_i} \tag{27}$$

- Resource utilization: Resource utilization is defined as the ratio of the number of allocated resources to the total number of available resources. It provides an assessment of how effectively resources are utilized and is calculated accordingly.

$$R = \frac{C}{W} \tag{28}$$

The proposed method exhibits performance enhancements compared to existing approaches when considering 15 virtual machines. Specifically, when compared to PSO [30], genetic [31], and GWO [32] algorithms, the proposed algorithm consistently outperforms them, as depicted in Fig. 4. It achieves lower skewness values faster and maintains them even with increased iterations. This superiority is attributed to the proposed algorithm's ability to adapt swiftly and accurately to different datasets, thanks to its improved learning rate and parameter tuning. As a result, it enables more efficient optimization and better overall performance.

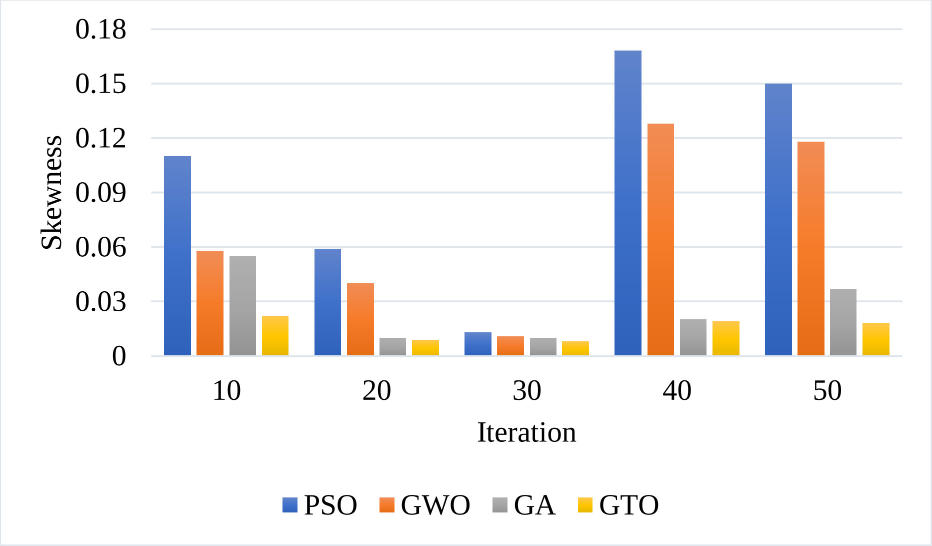

Fig. 4. Skewness comparison.

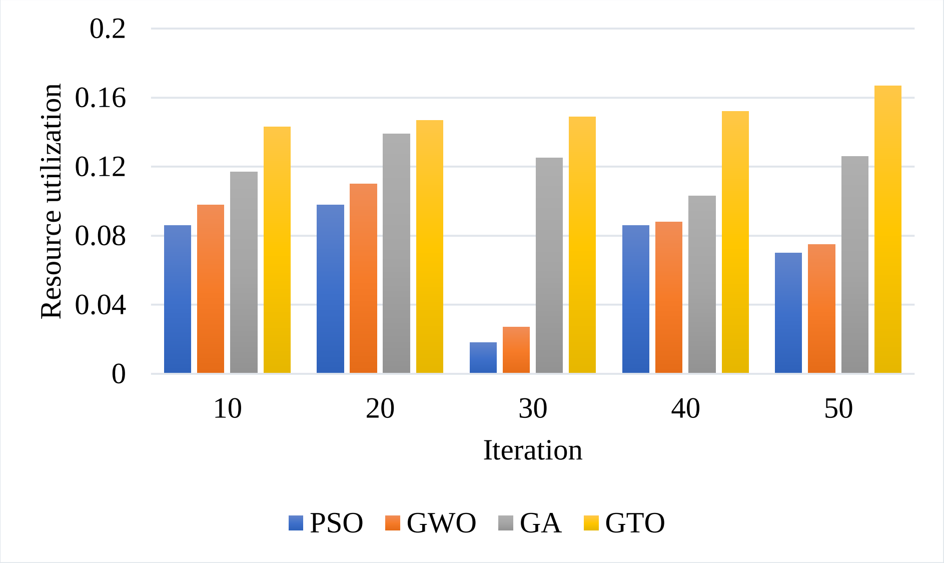

Fig. 5. Resource utilization comparison.

Fig. 5 demonstrates that the proposed algorithm utilizes more resources than existing techniques within the same number of iterations, indicating its enhanced efficiency and ability to achieve superior results with less iteration. Furthermore, Fig. 6 illustrates that the proposed algorithm exhibits improved memory utilization efficiency, requiring significantly less memory than existing techniques for the same number of iterations. Finally, Fig. 7 presents that the proposed algorithm accomplishes tasks more efficiently than existing models like GA, GWO, and PSO, as it achieves task completion in less time with the same number of iterations.

GTO in the context of VM allocation within cloud computing environments introduces a unique set of trade-offs and benefits that distinguish it from other optimization algorithms. While it may appear that GTO consumes more computational resources within the same number of iterations compared to some existing techniques, a closer examination reveals that the unique strengths of GTO can significantly outweigh the increased resource usage, ultimately leading to improved performance, efficiency, and sustainability in various aspects of VM allocation. GTO's use of extensive search techniques, such as Levy flights, might lead to higher resource consumption in terms of computation power and time. However, this trade-off is justified by its ability to explore a wider solution space, often resulting in superior VM allocations. The increased resource usage can be considered an investment in finding more energy-efficient and effective allocation solutions. GTO strikes a balance between exploration (discovering new allocation possibilities) and exploitation (refining promising solutions). This duality is vital in tackling the NP-hard problem of VM allocation. While some algorithms might prioritize one over the other, GTO excels in both, thereby enhancing the likelihood of finding optimal or near-optimal allocations.

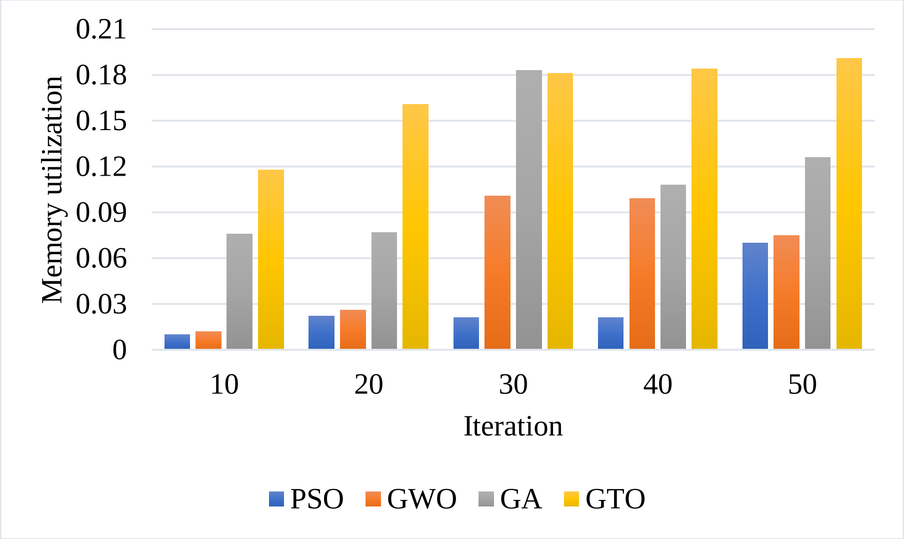

Fig. 6. Memory utilization comparison

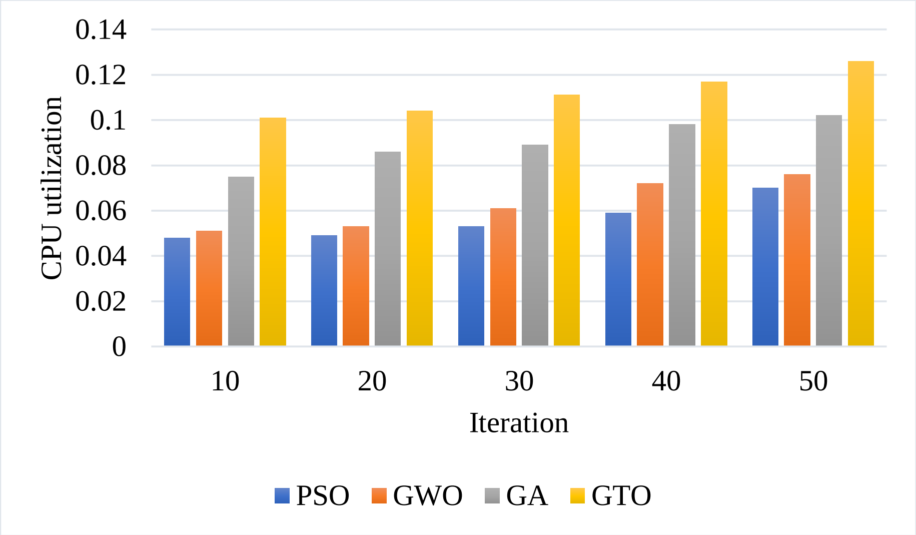

Fig. 7. CPU utilization comparison

GTO's ability to adapt its search behavior, mirroring the trevally's hunting tactics, allows it to respond effectively to changing conditions and evolving VM allocation demands. This adaptability is especially valuable in dynamic cloud environments. GTO's exploration phase, facilitated by Levy flights, enables it to perform global searches, effectively avoiding local optima. This global perspective ensures that VM allocations are not limited to suboptimal solutions, ultimately improving resource utilization and efficiency. GTO's incorporation of visual distortion due to the refraction of light is a unique feature that enhances its performance. This consideration ensures that VM allocations are not only optimal but also take into account real-world conditions, leading to more reliable and realistic allocation solutions. GTO's exploration phase increases the diversity of the population, preventing premature convergence. This diversity is crucial in avoiding stagnation and enabling the algorithm to jump out of local optima, which can be a common issue in other optimization techniques.

V. Conclusion

This paper introduced an energy-conscious optimization approach based on the GTO for VM allocation. It has been compared to existing methods, including GWO, genetic, and PSO algorithms. The experimental results and performance evaluations have demonstrated the superiority of the proposed algorithm in several aspects. Firstly, the proposed algorithm consistently outperforms other algorithms in terms of skewness. It achieves lower skewness values more rapidly and maintains them even with increased iterations. This improvement is attributed to the algorithm's enhanced learning rate and parameter tuning, allowing it to adapt more effectively to different datasets. Furthermore, the proposed algorithm exhibits improved resource utilization efficiency by effectively utilizing a greater number of resources compared to existing techniques within the same number of iterations. This indicates its enhanced efficiency and ability to achieve better results with less iteration. Moreover, the algorithm demonstrates superior memory utilization efficiency by requiring significantly less memory compared to existing techniques for the same number of iterations. This feature is valuable in resource-constrained environments where memory usage optimization is crucial. The findings highlight the promising performance and potential of the proposed GTO-based approach for VM allocation in cloud computing environments. Future research directions can explore the algorithm's applicability in different scenarios and consider additional parameters to address the complexities of diverse cloud computing environments.

While our study leverages the GTO to address VM allocation challenges in cloud computing, it is essential to acknowledge certain limitations. Firstly, GTO's resource-intensive nature, particularly in terms of computation, may pose practical constraints in real-time cloud environments where swift decision-making is crucial. Secondly, the effectiveness of the GTO algorithm may vary depending on the specific characteristics of a given cloud data center, such as size, workload, and infrastructure, which could limit its universality. Additionally, our study primarily focuses on energy efficiency and resource utilization aspects, potentially overlooking other critical performance metrics relevant to cloud service quality. Furthermore, while we account for visual distortion in VM allocation, the real-world applicability and accuracy of this consideration warrant further exploration. Despite these limitations, our research provides valuable insights into enhancing cloud sustainability and efficiency, offering a foundation for future investigations in the field.

REFERENCES

[1] B. Pourghebleh, A. A. Anvigh, A. R. Ramtin, and B. Mohammadi, "The importance of nature-inspired meta-heuristic algorithms for solving virtual machine consolidation problem in cloud environments," Cluster Computing, pp. 1-24, 2021.

[2] S. Iftikhar et al., "HunterPlus: AI based energy-efficient task scheduling for cloud–fog computing environments," Internet of Things, vol. 21, p. 100667, 2023.

[3] Y. Kumar, S. Kaul, and Y.-C. Hu, "Machine learning for energy-resource allocation, workflow scheduling and live migration in cloud computing: State-of-the-art survey," Sustainable Computing: Informatics and Systems, vol. 36, p. 100780, 2022.

[4] J. Liu, A. S. Prabuwono, A. W. Abulfaraj, S. Miniaoui, and N. Taheri, "Cognitive cloud framework for waste dumping analysis using deep learning vision computing in healthy environment," Computers and Electrical Engineering, vol. 110, p. 108814, 2023.

[5] J. A. Jeba, S. Roy, M. O. Rashid, S. T. Atik, and M. Whaiduzzaman, "Towards green cloud computing an algorithmic approach for energy minimization in cloud data centers," in Research Anthology on Architectures, Frameworks, and Integration Strategies for Distributed and Cloud Computing: IGI Global, 2021, pp. 846-872.

[6] P. Huang et al., "A review of data centers as prosumers in district energy systems: Renewable energy integration and waste heat reuse for district heating," Applied energy, vol. 258, p. 114109, 2020.

[7] J. Ni and X. Bai, "A review of air conditioning energy performance in data centers," Renewable and sustainable energy reviews, vol. 67, pp. 625-640, 2017.

[8] M. Koot and F. Wijnhoven, "Usage impact on data center electricity needs: A system dynamic forecasting model," Applied Energy, vol. 291, p. 116798, 2021.

[9] A. S. Andrae and T. Edler, "On global electricity usage of communication technology: trends to 2030," Challenges, vol. 6, no. 1, pp. 117-157, 2015.

[10] V. Hayyolalam, B. Pourghebleh, A. A. P. Kazem, and A. Ghaffari, "Exploring the state-of-the-art service composition approaches in cloud manufacturing systems to enhance upcoming techniques," The International Journal of Advanced Manufacturing Technology, vol. 105, no. 1-4, pp. 471-498, 2019.

[11] P. He, N. Almasifar, A. Mehbodniya, D. Javaheri, and J. L. Webber, "Towards green smart cities using Internet of Things and optimization algorithms: A systematic and bibliometric review," Sustainable Computing: Informatics and Systems, vol. 36, p. 100822, 2022, doi: https://doi.org/10.1016/j.suscom.2022.100822.

[12] R. Singh et al., "Analysis of Network Slicing for Management of 5G Networks Using Machine Learning Techniques," Wireless Communications and Mobile Computing, vol. 2022, 2022.

[13] T. Taami, S. Azizi, and R. Yarinezhad, "Unequal sized cells based on cross shapes for data collection in green Internet of Things (IoT) networks," Wireless Networks, pp. 1-18, 2023.

[14] T. Gera, J. Singh, A. Mehbodniya, J. L. Webber, M. Shabaz, and D. Thakur, "Dominant feature selection and machine learning-based hybrid approach to analyze android ransomware," Security and Communication Networks, vol. 2021, pp. 1-22, 2021.

[15] S. Mahmoudinazlou and C. Kwon, "A Hybrid Genetic Algorithm for the min-max Multiple Traveling Salesman Problem," arXiv preprint arXiv:2307.07120, 2023.

[16] C. Han and X. Fu, "Challenge and Opportunity: Deep Learning-Based Stock Price Prediction by Using Bi-Directional LSTM Model," Frontiers in Business, Economics and Management, vol. 8, no. 2, pp. 51-54, 2023.

[17] R. Soleimani and E. Lobaton, "Enhancing Inference on Physiological and Kinematic Periodic Signals via Phase-Based Interpretability and Multi-Task Learning," Information, vol. 13, no. 7, p. 326, 2022.

[18] B. M. Jafari, M. Zhao, and A. Jafari, "Rumi: An Intelligent Agent Enhancing Learning Management Systems Using Machine Learning Techniques," Journal of Software Engineering and Applications, vol. 15, no. 9, pp. 325-343, 2022.

[19] M. Shahin et al., "Cluster-based association rule mining for an intersection accident dataset," in 2021 International Conference on Computing, Electronic and Electrical Engineering (ICE Cube), 2021: IEEE, pp. 1-6, doi: 10.1109/ICECube53880.2021.9628206.

[20] S. Saeidi, S. Enjedani, E. Alvandi Behineh, K. Tehranian, and S. Jazayerifar, "Factors Affecting Public Transportation Use during Pandemic: An Integrated Approach of Technology Acceptance Model and Theory of Planned Behavior," Tehnički glasnik, vol. 18, pp. 1-12, 09/01 2023, doi: 10.31803/tg-20230601145322.

[21] G. J. Ibrahim, T. A. Rashid, and M. O. Akinsolu, "An energy efficient service composition mechanism using a hybrid meta-heuristic algorithm in a mobile cloud environment," Journal of parallel and distributed computing, vol. 143, pp. 77-87, 2020.

[22] V. Hayyolalam, B. Pourghebleh, M. R. Chehrehzad, and A. A. Pourhaji Kazem, "Single-objective service composition methods in cloud manufacturing systems: Recent techniques, classification, and future trends," Concurrency and Computation: Practice and Experience, vol. 34, no. 5, p. e6698, 2022.

[23] M. Hanini, S. E. Kafhali, and K. Salah, "Dynamic VM allocation and traffic control to manage QoS and energy consumption in cloud computing environment," International Journal of Computer Applications in Technology, vol. 60, no. 4, pp. 307-316, 2019.

[24] K. Dubey and S. C. Sharma, "An extended intelligent water drop approach for efficient VM allocation in secure cloud computing framework," Journal of King Saud University-Computer and Information Sciences, vol. 34, no. 7, pp. 3948-3958, 2022.

[25] J. K. Samriya, S. Chandra Patel, M. Khurana, P. K. Tiwari, and O. Cheikhrouhou, "Intelligent SLA-aware VM allocation and energy minimization approach with EPO algorithm for cloud computing environment," Mathematical Problems in Engineering, vol. 2021, pp. 1-13, 2021.

[26] N. N. Devi and S. V. Kumar, "SLAV Mitigation and Energy-Efficient VM Allocation Technique Using Improvised Grey Wolf Optimization Algorithm for Cloud Computing," in 2022 8th International Conference on Advanced Computing and Communication Systems (ICACCS), 2022, vol. 1: IEEE, pp. 155-160.

[27] H. Xing, J. Zhu, R. Qu, P. Dai, S. Luo, and M. A. Iqbal, "An ACO for energy-efficient and traffic-aware virtual machine placement in cloud

computing," Swarm and Evolutionary Computation, vol. 68, p. 101012, 2022.

[28] A. Sheeba and B. Uma Maheswari, "An efficient fault tolerance scheme based enhanced firefly optimization for virtual machine placement in cloud computing," Concurrency and Computation: Practice and Experience, vol. 35, no. 7, p. e7610, 2023.

[29] H. T. Sadeeq and A. M. Abdulazeez, "Giant Trevally Optimizer (GTO): A Novel Metaheuristic Algorithm for Global Optimization and Challenging Engineering Problems," IEEE Access, vol. 10, pp. 121615-121640, 2022.

[30] D. H. Phan, J. Suzuki, R. Carroll, S. Balasubramaniam, W. Donnelly, and D. Botvich, "Evolutionary multiobjective optimization for green clouds," in Proceedings of the 14th annual conference companion on Genetic and evolutionary computation, 2012, pp. 19-26.

[31] N. Moganarangan, R. Babukarthik, S. Bhuvaneswari, M. S. Basha, and P. Dhavachelvan, "A novel algorithm for reducing energy-consumption in cloud computing environment: Web service computing approach," Journal of King Saud University-Computer and Information Sciences, vol. 28, no. 1, pp. 55-67, 2016.

[32] C. T. Joseph, K. Chandrasekaran, and R. Cyriac, "A novel family genetic approach for virtual machine allocation," Procedia Computer Science, vol. 46, pp. 558-565, 2015.