UNIVERSAL WISER PUBLISHER

Research Article

A Novel Approach for Energy-Efficient Container Migration Using GNBO

Rukmini S[1*], Shridevi Soma[2], Rajkumar Buyya[3]

[1]Department of Computer Science, Government Women's Polytechnic, Kalaburagi, Karnataka, India
[2]Department of Computer Science, Poojya Dodappa Appa College of Engineering, Kalaburagi, Karnataka, India
[3]School of Computing and Information Systems, The University of Melbourne, Melbourne, Australia
Email: rukminis@pdaengg.com

Received: 26 May 2023; **Revised:** 26 June 2023; **Accepted:** 4 August 2023

Abstract: Cloud services are increasingly becoming available through containers because of their scalability, portability, and reliable deployment, particularly in microservices and smart vehicles. Due to the diversity of workloads and cloud resources, the scheduler component of cloud containers plays a crucial role in optimizing energy efficiency and minimizing costs. The growing demand for cloud services poses a challenge in terms of energy consumption. It is possible to optimize energy consumption on servers by utilizing live migration technology. This study aims to propose a hybrid model that will facilitate the migration of containers from one server to another using gradient descent Namib beetle optimization (GNBO) algorithms and thereby reduce the amount of energy consumed by cloud servers. The work is carried out by cloud simulation with physical machines (PMs), virtual machines (VMs), and containers. The tasks are allocated to VM in a round-robin manner. The actor-critic neural network (ACNN) is used to predict the load of PMs. Overloading and underloading conditions are determined based on the load. A hybrid optimization algorithm, GNBO, calculates the optimal solution based on predicted load, migration costs, resource utilization, energy consumption, and network bandwidth. This results in a load of 0.177 millions of instructions per second (MIPS), migration costs of 10.146 J, and energy consumption optimized to 0.068 W.

***Keywords*:** container migration, cloud computing, energy consumption optimization, ACNN, VM migration, live migration

MSC: 32C05, 32C07, 32C22, 32C30

1. Introduction

Cloud computing elevates software computation when compared to traditional computing methods by providing on-demand online computing scalable infrastructure and storage, which are extensions of storage-based devices like desktops and phones, and offering apparent access to obtain them anytime and anywhere [1]. The cloud model involves various key modules, like the user interaction interface, the management of resources and services, and the resource provisioning phase. The system resource management component controls a huge network of servers executing in parallel. It also utilizes virtualization models for dynamically allocating and de-allocating computing resources. These days, the computing domain is capable of envisioning transitioning into cloud computing platforms due to breathtaking advancements in computing and information techniques over the previous three decades [2]. A growing demand for cloud resources has led cloud providers to handle warehouse-sized data centers. This large-scale cloud data center (CDC) is equipped with thousands of computing servers connected by high-speed communication links. It consumes a lot of electricity. As part of this process, approximately 30% of servers remain in idle mode, while approximately 12-15% are used for completing resource requests. The workload requests for cloud applications, including interactive ones, are frequently changing, which causes dynamic resource demands, resulting in service level agreements (SLAs) being violated and performance degradation [28]. Cloud computing is an eminent technique that offers the services of information technologies to address several issues [3, 4]. Cloud computing with virtualization offers end-user services, such as application levels and hardware resources [4-6]. Meanwhile, cloud computing poses several features that can help scientists. By utilizing the cloud computing model, one can offer a huge number of distributed platforms using the opposite platform [4, 7, 8]. The requirement of cloud computing has simplified the dynamic operation of computing, storage resources, and networking to offer desired services [9].

Cloud computing commonly refers to hosted services. It represents the physical hardware virtualization [10], wherein data is arranged in particular centers. In cloud computing services, the virtual cluster permits the active allocation of visualized resources. The management of users and virtual storage represents a common cloud model. In cloud platforms, the input represents job association, and its arrangement is imperative for the competency of complete cloud services [7]. With the progression of virtualization, one of the crucial virtualization techniques utilized for hosting cloud services is virtual computers, which share networking, processing, and storage resources with real machines [9]. The standard of containers facilitates a reliable and lightweight platform, which permits the software program to begin sharing and underlies the operating model. Thus, the containers are utilized in highly effective deployments of the same applications, which utilize similar kernel code [11]. The container technique has various industrial applications. A few organizations where containers are employed include [12, 13] containers of Internet of Things (IoT) tools, containerization in gaming, healthcare, web applications, financial services providers, micro-services models, scientific workflows, edge computing, marketing, and advertising [13].

Thus, it is imperative to model a sensible container migration model for realizing load balancing, considering edge networks, and reducing the influence of container migration. To address the aforementioned issue, a container migration selection model and sensible migration timing are required, and meanwhile, the container migration techniques need normal selection. Regarding the issue of migration timing, some classical models rely on static thresholds, whereas others utilize prediction models like machine learning [14, 15] and linear regression [15, 18, 30-33]. Some techniques are provided as heuristic techniques for getting the approximate best solution, but it is easy to fall into local optima. Bio-motivated meta-heuristic techniques, like the genetic algorithm (GA), ant colony system (ACS), and particle swarm optimization (PSO), cannot avoid falling into the local optimum solution but also acquire high-quality approximations [15, 20]. Some of the meta-heuristic models, like PSO and GA, need extraordinary encoding for combinatorial optimization issues as they devised continuous issues. ACS is devised to discover the optimum mapping of virtual machines on servers in the data center to search for Pareto-optimal solutions considering two dimensions: waste of resources and consumption of power. However, the aforementioned technique is utilized in container redeployment and considers delay and migration costs in edge computing [15]. There have been several studies conducted on both container migrations and virtual machine (VM) migrations, and it has been determined that container migrations are better than virtual machine migrations because of their smaller size, resulting in a faster migration time or reduced migration cost [29].

The goal is to devise a model for container migration to optimize energy consumption in the cloud. The simulation of the cloud is done, and the tasks are allocated to the VM in a round-robin manner. Here, the load of physical machines (PMs) is predicted using actor-critic neural network (ACNN). If the load is higher than the threshold value, then the containers are optimally migrated from overloaded PM to underloaded PM using gradient descent Namib beetle optimization (GNBO) by considering multi-objectives like predicted load, migration cost, resource utilization, energy consumption, and network bandwidth.

The major contributions include:

• Proposed GNBO for container migration: The GNBO is utilized for container migration in cloud computing by considering multi-objectives, like predicted load, migration cost, resource utilization, energy consumption, and network bandwidth. Here, GNBO is obtained by blending the gradient descent optimization algorithm and Namib beetle optimization (NBO).

The paper organization is done as follows: Section 2 describes priority-devised container migration techniques. Section 3 offers a cloud model. Section 4 describes the container migration model. Section 5 discusses efficiency, and Section 6 concludes the paper.

2. Motivation

Intellectual management of tasks in huge-scale infrastructures is complex because of the extensively volatile platform of contemporary workload applications. Container migration has emerged to alleviate the issue by utilizing heuristics to rapidly reach decisions or artificial intelligence (AI)-driven techniques for dynamic scenarios.

2.1 *Related work*

An overview of previous research on container migration is presented in this section, along with its merits and challenges, as well as an explanation of the details of the work.

Table 1. Summary of related work

Work	Objective achieved	Energy optimization considered [YES/NO]	Demerits/GAP
Benomar et al. [19]	Cloud management middleware named OpenStack.	NO	The work does not consider various policies for managing and migrating the containers.
Bhardwaj and Krishna [26]	LXD/CR container-based migration.	NO	Security in container migration.
Saboor et al. [13]	Micro-service execution for performing the container migration.	NO	Not tested in a simulation environment.
Ma et al. [15]	Container migration-based decision-making (CMDM) model for container migration in power IoT.	NO	This method did not adapt mobility and enhance the migration model.
Al-Tarawneh [16]	Mobility-aware container migration.	NO	Migration time was high.
Singh and Singh [9]	Container migration to migrate the host front to back.	NO	Could not execute in other cloud platforms.
Chhikara et al. [11]	Energy-efficient container migration.	YES	Did not consider multi-objective criteria for migration.
He et al. [17]	Container migration using dynamic resource dependency.	NO	The work does not minimize the count of live migrations.
Buyya et al. [24]	Multi-objective-based container migration.	YES	Migration cost was not considered as a parameter.

Benomar et al. [19] presented de facto cloud management middleware named OpenStack for enabling the control of the remote platform to provision and control the containers. However, this method did not consider various policies for managing and migrating the containers. To adopt other policies, Bhardwaj and Krishna [26] devised the LXD/CR container-based migration technique for container migration. Here, the experimental set-up was adapted for executing and analyzing the efficiency of the method. The checkpoint/restore mechanism was adapted for container migration. However, this method did not consider security. To imply security, Saboor et al. [13] devised a conceptual model for managing the huge quantity of micro-service execution for performing the container migration. Here, four key agent services were considered, which include intellectual partitioning and mutation actions. However, this method was not examined and tested on the simulation platform. To test in simulation infrastructure, Ma et al. [15] devise a CMDM model for container migration in power IoT. However, this method did not adapt mobility or enhance the migration model. To adapt to mobility, Al-Tarawneh [16] developed a mobility-aware container migration technique for container migration in a cloud-based IoT model. The method was devised based on an integrated MCDM approach. However, the expected time of migration was very high. To reduce migration time, Singh and Singh [6] devised a technique for container migration to migrate the host from front to back. The method aids in minimizing the quantity of data transmission over the network. However, the method was not able to execute on other cloud platforms. To execute on other platforms, Chhikara et al. [9] developed an energy-efficient container migration model for container migration using the source host server to the destination host server to meet container resource needs. Here, the new technique was utilized for determining the better destination host for container placement to address host over-load or under-load issues, considering the best-fit container placement. The need for extra resources for one container was not utilized. To acquire extra resources, He et al. [11] devised a technique for container migration. Here, the resource competitions amidst the live migrations were modeled as a dynamic resource dependency graph. It did not minimize the number of live migrations.

2.2 *Challenges*

Some of the challenges of container migration are described below:

• A technique is devised by Benomar et al. [19] to deploy or manage the containers at fog levels to attain effective container migration. However, this technique did not examine the system's scalability by combining the S4T platform with emulators.

• An approach for container migration was implemented by Saboor et al. [13], but this method did not combine expert scheduling models and learning techniques for containers and microservices.

• The developed method by Ma et al. [15] did not use definite mathematical techniques for approximately improving the algorithm's stability.

• The technique devised by Singh et al. [9] for container migration for migrating back to a similar host failed to minimize the data transfer between the source and destination hosts.

• The definite migration of runtime applications must take place in real-world cases, and thus it is essential for accessing the migration process by employing different workloads.

3. System model

Cloud computing is a type of computing model that has attracted huge attention because of its various applications and its simple nature. The VM resources use various configurations with various amounts of memory, storage, and power. Due to the complete control of cloud functions, a performance bottleneck may appear and may lead cloud infrastructure to become ineffective. Thus, devising a container migration strategy is important. Figure 1 exposes the cloud model for revealing the container migration process. The system model is a network model of cloud data center networks (DCNs), which are designed to provide efficient and scalable communication within large-scale data centers. They aim to optimize resource utilization, minimize latency, and improve overall network performance.

Changes are made in Figure 1.

Assume a cloud model with a number of PMs, such that $T = \{T_1, T_2, ..., T_x, ..., T_y\}$. Each PM contains one or several VMs, such that $R = \{R_1, R_2, ..., R_r, ..., R_s\}$. Furthermore, each VM contains one or several containers, such that $S = \{S_1, S_2, ..., S_w, ..., S_z\}$. Each VM needs the processing elements, like millions of instructions per second (MIPS), memory, and the applications or tasks that run on containers, such that

$$H = \{H_1, H_2, H_3, \cdots, H_m, \cdots, H_n\}. \tag{1}$$

The container parameters include CPU, memory, and MIPS.

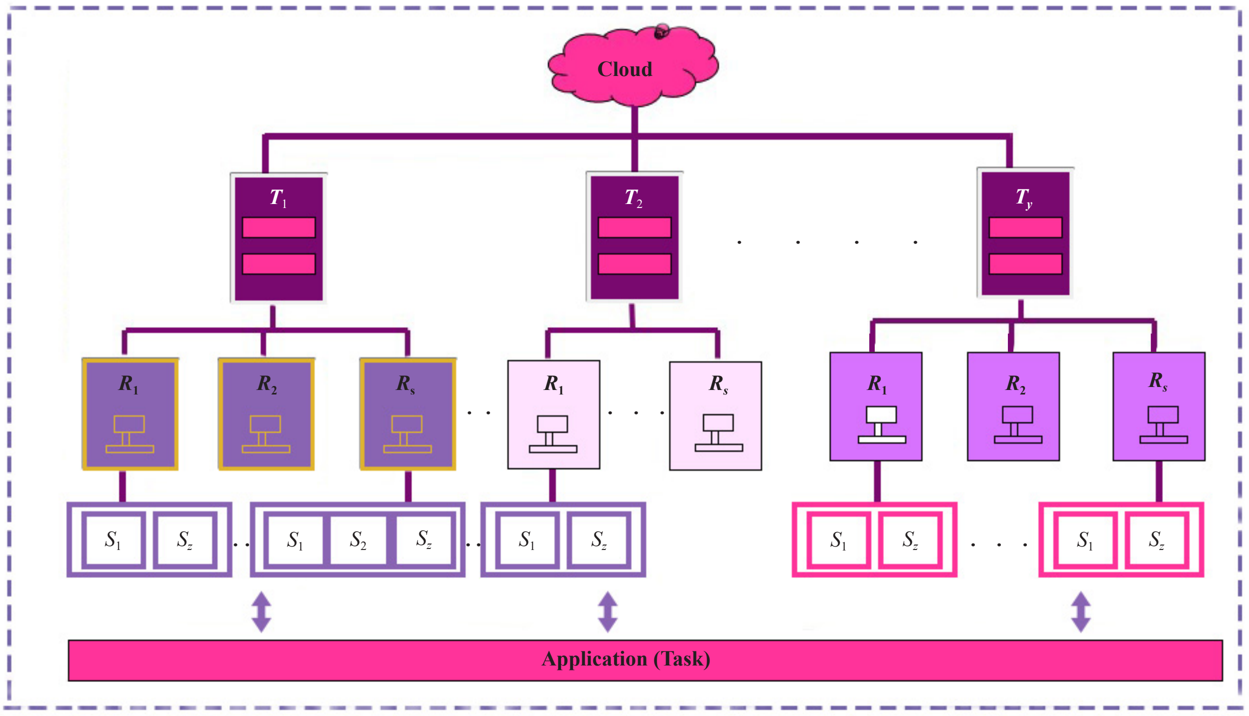


Figure 1. System model

4. Proposed GNBO for container migration in cloud computing

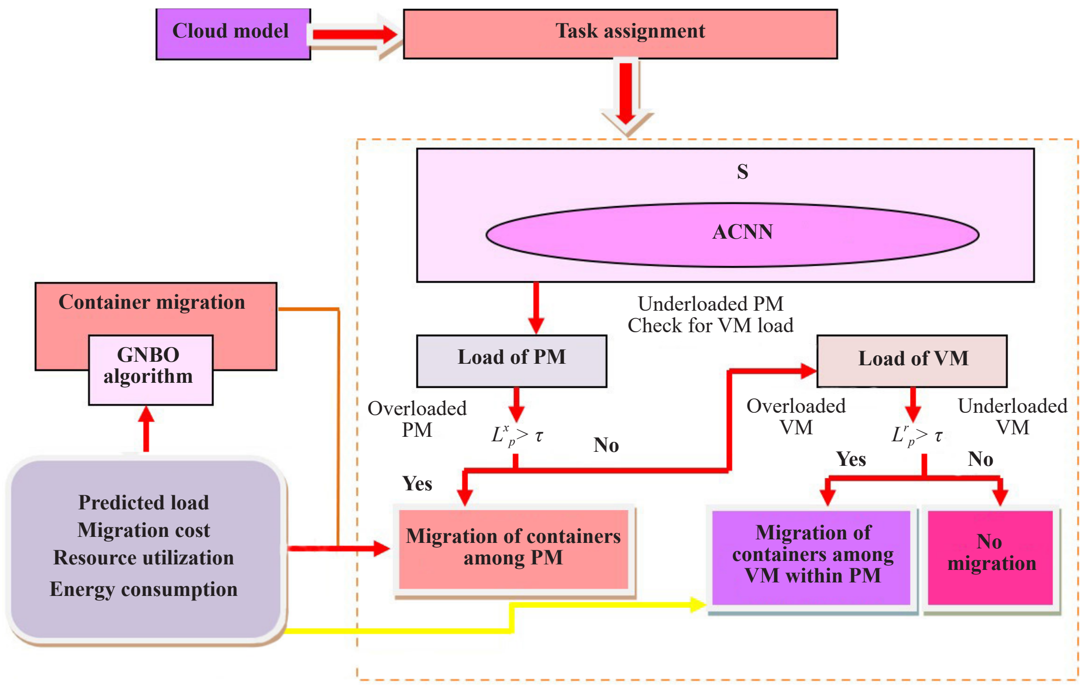


Figure 2. Structure of proposed GNBO for container migration in cloud computing

The objective is to present an effective model for container migration in cloud computing. Initially, cloud simulation with PM, VM, and containers is done, and then tasks are assigned to the VM in a round-robin manner. Thereafter, the load of PM is predicted utilizing ACNN [21]. If the load is greater than the threshold value, then the containers are optimally migrated from overloaded PM to underloaded PM using GNBO by considering multi-objectives like predicted load, migration cost, resource utilization, energy consumption, and network bandwidth. Here, the GNBO is obtained by blending the gradient descent algorithm [10] and the NBO [20]. Figure 2 shows the structure of container migration model in cloud computing using the proposed GNBO.

4.1 *Algorithm steps*

Step 1) Initially the task is assigned to the container in a round-robin manner.
Step 2) Predict a load of VM and PM using the ACNN.
Step 3) If $L^x_p > \tau$, then migrate the container among PM.
Step 4) If $L^r_p > \tau$, then migrate the container among VM within the PM.
Step 5) Container migration is done optimally using the proposed GNBO.

4.2 *Load prediction of VM and PM using actor-critic neural network*

A load of rth VM and xth PM is provided as ACNN input to predict the load. ACNN [21] represents a supervised classifier that mostly comprises two operating modules, namely actor and critic modules. The ACNN is utilized to predict the state of switching, and it provides the processing speed delivered with good accuracy. Meanwhile, the critic module provides the control action that offers effective prediction. ACNN has two modules: the actor module and the critic module. The actor module selects the container to be migrated based on the load pattern, and the critic module selects the most overloaded container among the containers selected in the actor phase to migrate. The procedure is repeated to find the optimal output.

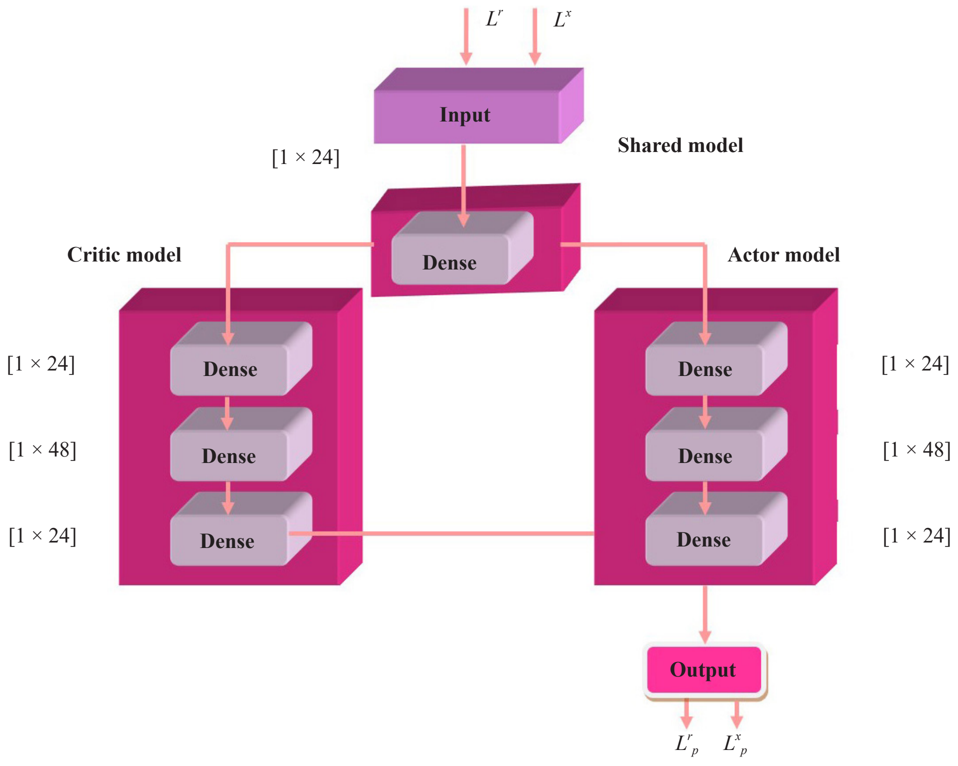


Figure 3. ACNN model

Figure 3 depicts the ACNN model.

(i) Actor module: Assume actor mode input is denoted by $P(o)$ and is a system state that aids in estimating the best action, and the output obtained is denoted by $J^{Aa}(o)$. The actor module executes absolutely with the parameterized model, such as a neural network. In the actor module, a straightforward feed-forward neural model with one hidden layer is employed for functioning. The actor output is modeled as

$$J^{Aa}(o) = Y^u_{Aa}(o)\gamma(U^u_{Aa}(o)P_{Aa}(o)) = Y^u_{Aa}(o)\chi_{Aa}(o), \tag{2}$$

where $P_{Aa}(o)$ represents input, $U_{Aa}(o)$ denotes weight matrix amidst the input and hidden layer while $Y_{Aa}(o)$ signifies weight amidst the hidden and output layer. In addition, the activation attribute $\mathcal{X}(\cdot)$ is expressed as follows, and it represents a hyperbolic tangent function.

$$\chi(v) = \frac{e^v - e^{-v}}{e^v + e^{-v}} \tag{3}$$

(ii) Critic module: Critic module provides the action of control using the newest criterion by modeling the further accumulative reward-to-go measure represented as

$$\text{Reward}(o) = \sum_{g=0}^{u} \lambda^g W(o + g + 1) \tag{4}$$

where λ^g represents a discount factor that relies in $0 < \lambda^g < 1$, o expresses time, and u denotes terminal step. Thus, a similar three-layer feed-forward neural model is adapted for approximating Reward(o) in equation (4).

The critic module output represents an approximation of Reward(o), and is expressed as

$$C(o) = Y^u_{Cc}(o)\chi(U^u_{Cc}(o)P_{Cc}(o)) = Y^u_{Cc}(o)\chi_{Cc}(o), \tag{5}$$

where P_{Cc} symbolizes critic input, U^u_{Cc} is the weight matrix amidst input and hidden layer. Meanwhile, the weight matrix amidst the hidden and output layer is denoted by $Y^u_{Cc}(o)$. The predicted load of rth VM is denoted as L^r_p, and the predicted load of x^{th} PM is denoted as L^x_p.

4.3 *Container migration using proposed GNBO*

Container migration is termed an effective lightweight virtualization model for managing the load. In container migration, the container is migrated between two models and performs several applications to control through physical models. The container can handle the delay sensitivity of mobile applications for usage and provisioning. The containers are devised in a way to facilitate quick migration and the effective making of decisions. Here, the container migration is done using the proposed GNBO, considering two conditions. The first condition includes migrating containers among VMs within the PM, and the second condition includes migrating containers among PMs. These conditions are briefly illustrated below.

4.3.1 *Condition 1: Migrate container among VM within the PM*

The first condition is the migration of containers among the VMs through the PM. Here, the overloaded containers are shifted to underloaded containers amongst the VMs by checking the load of each VM. The solution encoding and fitness are described below.

4.3.1.1 *Solution encoding*

Figure 4 represents the solution representation process. Here, R_3 seems to be overloaded. Hence, the container migration is done among VMs within the PM in such a way that each VM contains two containers. Thus, R_1 contains containers 1 and 2, R_2 contains containers 6 and 3, and R_3 contains containers 4 and 5 to balance the load by performing container migration among VMs within the PM.

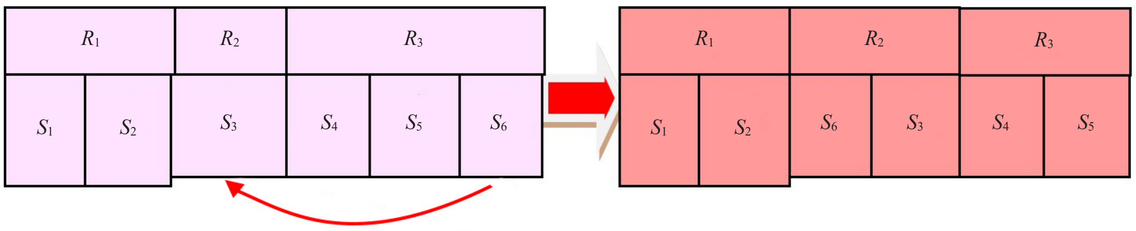


Figure 4. Solution encoding of container migration among VM within the PM

4.3.1.2 *Fitness function*

Fitness is a maximization function that comprises the load of VM, the energy consumption of VM, resource utilization, and migration. Here, fitness is manifested as

$$\Re = \frac{1}{4}\sum_{r=1}^{s}\left(L_p^r + S + P_{VM} + RU\right), \tag{6}$$

where S represents migration cost, P_{VM} symbolizes energy consumption of VM, RU denotes resource utilization and L_p^r expresses predicted load of rth VM, and s represents total VMs, such that $1 \le r \le s$.

Here, the migration cost is given by

$$S = \frac{1}{z}\sum_{w=1}^{z}\frac{V}{k * l}, \tag{7}$$

where z depicts total containers, V refers to the number of migration of containers, k symbolizes migration constant, and l is a normalizing factor.

The energy consumption [22] of VM is manifested as

$$P_{VM} = \sum_{w=1}^{z} P_{container}, \tag{8}$$

where $P_{container}$ refers energy consumed by a specific container. It is formulated as

$$P_{container} = \left(\frac{P_{VM_{idle}}}{s}\right) + V_c * \left(P_{VM_{\max}} - P_{VM_{idle}}\right) * U_c, \tag{9}$$

where U_c is the utilization level of the container, and V_c is a fraction of VM resources allocated to the container, $P_{container}$ refers to energy consumed by a specific container and, $P_{VM_{\max}}$ and $P_{VM_{idle}}$ refer to energy consumption when the CPU is 100% utilized and idle in VM.

The resource utilization is formulated as

$$RU = \frac{CPU_w^V * mem_w^V * MIPS_w^V}{CPU_w^T * mem_w^T * MIPS_w^T}. \tag{10}$$

4.3.2 *Condition 2: Migrate container among PM*

The second condition is the migration of containers among PMs. Here, the overloaded containers are shifted to underloaded containers amongst the PMs by checking the load of each PM. The solution encoding and fitness are described below.

4.3.2.1 *Solution encoding*

Figure 5 represents the solution representation process. Here, R_3 seems to be overloaded. Thus, the container migration is done among PMs in such a way that each VM contains two containers. In T_1, the R_1 contains containers 1 and 2, while R_2 contains containers 3 and 4, and R_3 contains containers 5 and 6, whereas in T_2, the R_1 contains containers 8 and 9, while R_2 contains containers 7 and 10, to balance load by performing container migration among PM. Here, the container migration takes place amongst the PM.

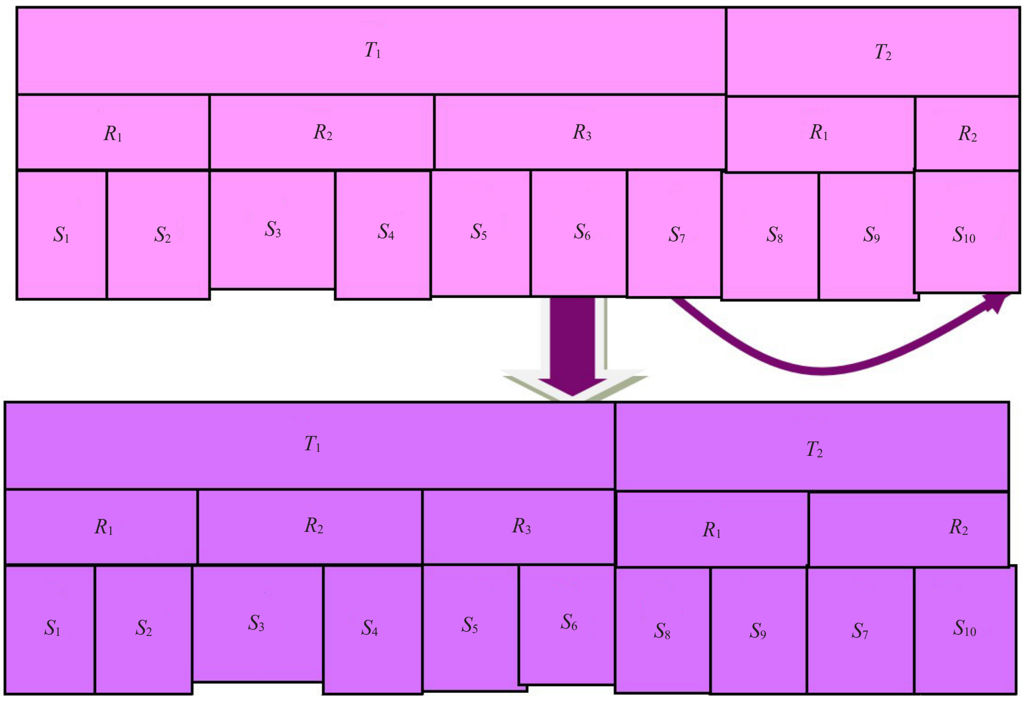


Figure 5. Solution encoding of container migration among PM

4.3.2.2 *Fitness function*

Fitness is a maximization function that comprises the load of PM, energy consumption of PM, resource utilization, and migration. Here, fitness is manifested as

$$\Re = \frac{1}{4}\sum_{x=1}^{y}\left(L_p^x + S + P_{PM} + RU\right), \tag{11}$$

where S represents migration cost, P_{PM} symbolizes energy consumption of PM, RU denotes resource utilization and L^x_p expresses predicted load of xth PM obtained from ACNN, and y represents total PMs, such that $1 \le x \le y$.

Here, the formula of migration cost, and resource utilization is obtained from Section 4.3.1.2.

The energy consumption is manifested as

$$P = \sum_{w=1}^{z} P_{PM} \tag{12}$$

$$P_{PM} = \left(\frac{P_{idle}}{y}\right) + \omega_{PM} * \left(P_{\max} - P_{idle}\right) * u, \tag{13}$$

where u refers to CPU utilization level and, P_{idle} and $P_{\max}$ refer to energy consumption when the CPU is idle in VM and 100% or maximum utilized.

4.3.3 *Proposed GNBO algorithm*

A comparison between the behavior of the Namib beetle and its application in the proposed GNBO algorithm is provided in Table 2, followed by a detailed description of the algorithm with pseudocode.

Table 2. Analogy between Namib beetle and proposed GNBO

Namib beetle	Proposed GNBO
Initial population	Initialize matrix with possible solutions for container migration
Suitability of area for water collection	Find a suitable VM or PM based on the load prediction by ACNN
Moving toward wet area	Migrate container
Move the wet mass and removal of population	Remove repeated allocation
Return back safely	Repeat the procedure and fine-tune with gradient descent optimization

An efficient GNBO algorithm is used for container migration, which is obtained by combining gradient descent and NBO. The NBO [20] is inspired by the behavior of Namib beetles, which are utilized for collecting water in the desert and are considered an intellectual part of generating water.

The behavior of beetles is modeled, which moves to heights for absorbing steam and air humidity and is utilized for solving optimization issues. It poses the highest capability to minimize dimensions and feature selection space, has high stability, and ranks first in finding optimal solutions with less error. Meanwhile, the gradient descent algorithm [25] aimed to offer intuitions concerning the behavior of various algorithms. It offers better balance amidst the accuracy of parameter updates and time, and it converges to the global optimal solution. In addition, it is more robust to poor initialization and aids in the training of deep and complicated networks. Hence, the blending of gradient descent and NBO offers a better solution. The steps of the proposed GNBO are explained below:

Step 1) Initialization

As per NBO [20], each solution is modeled as a beetle, encoded with B dimensions, and given as

$$G = \left[h_1, h_2, h_3, \cdots, h_B\right]. \tag{14}$$

Each beetle possesses B decision attributes. Some of it is arbitrarily produced as K in the range M and N, modeled as the initial population given by

$$Q = \begin{bmatrix} G_{1,1} & G_{1,2} & \cdots & G_{1,B} \\ G_{2,1} & G_{2,2} & \cdots & G_{2,B} \\ \vdots & \vdots & \vdots & \vdots \\ G_{K,1} & G_{K,2} & \cdots & G_{K,B} \end{bmatrix}. \tag{15}$$

Here, Q represents initial beetle's population. Here, $G_{a,b}$ expresses bth component linked to ath beetle. The equation to randomly build any solution in problem space is given by

$$G_{a,b} = M + (N - M) \cdot rand(0,1). \tag{16}$$

Step 2) Find fitness function

Here, the fitness function is explained in Sections 4.3.1.2 and 4.3.2.2. The first fitness is utilized when the container is to be migrated among VMs within the PM, and the second fitness is used when the container is to be migrated among PMs.

Step 3) Suitability of each region for the collection of water

It is assumed that the beetle is positioned in an optimal region, and this optimal region can be a focus for other beetles and administer them to these regions for accumulating water. Each region wherein the beetle, like G_a is positioned can pose a capacity to receive various beetles which is represented as

$$F_a = F_{\max} . \sin\left(\frac{\Re(G_a) - \Re_{\min}}{\Re_{\max} - \Re_{\min}} \cdot \frac{\pi}{2} \right), \tag{17}$$

where F_a represents the capacity of beetle count in the region wherein the beetle G_a is positioned, $F_{\max}$ refers maximum capacity of beetles, $\Re(G_a)$ refers competence of beetle G_a, and $\Re_{\min}$ and $\Re_{\max}$ represent the minimum and maximum competencies of population beetles.

The count of new beetles that are in the process of being built and that follow the prior beetles that are accumulating water is represented by

$$F = \sum_{a=1}^{K} F_a = F_1 + F_2 + \cdots + F_K \tag{18}$$

$$F = \sum_{a=1}^{K} F_{\max} . \sin\left(\frac{\Re(G_a) - \Re_{\min}}{\Re_{\max} - \Re_{\min}} \cdot \frac{\pi}{2} \right) = F_{\max} . \sin\left(\frac{\Re(G_a) - \Re_{\min}}{\Re_{\max} - \Re_{\min}} \cdot \frac{\pi}{2} \right) + \cdots + F_{\max} . \sin\left(\frac{\Re(G_K) - \Re_{\min}}{\Re_{\max} - \Re_{\min}} \cdot \frac{\pi}{2} \right). \tag{19}$$

Step 4) Moving towards wet regions

Each beetle requires choosing regions with adequate moisture to discover water. This focus on receiving moisture decreases with increasing distance. Consider a beetle G_a is in one region and other beetle, like G_b is in problem space, and F_a is a beetle that desires to move towards beetle G_a. The distance amidst these two beetles can be expressed as

$$\partial_{a,b} = \|G_a - G_b\| = \sqrt{\sum_{\ell=1}^{B} \left(G_{a,\ell} - G_{b,\ell}\right)^2}. \tag{20}$$

The quantity of focus in a region to attract beetles is modeled. Here, the beetle G_a is considered to have entered the region, in which the beetle G_a is positioned as

$$Humidity(d) = \varepsilon * Humidity_0.\exp\left(-\partial_{a,b}^{c}\right), \tag{21}$$

where $Humidity_0$ refers initial humidity, $Humidity(d)$ signifies moisture quantity that G_b feels from regions of G_a and c symbolizes power, which can be either 1.5 to 2, $\partial_{a,b}$ represents distance between two beetles and ε is a coefficient.

The coefficient can be modeled as

$$\varepsilon = \varepsilon_{\max} - \varepsilon_0 \cdot \left(1 - \frac{\varpi}{\varpi_{\max}}\right) \cdot rand\left(0,1\right), \tag{22}$$

where ε_0 symbolizes initial coefficient of humidity increase, $rand(0, 1)$ refers a random number between 0 and 1, $\varepsilon_{\max}$ signifies maximum coefficient of humidity increase, $\varpi_{\max}$ is maximum iteration, and ϖ is current iteration.

The method of attracting one beetle to another beetle, wherein the present position of one beetle and the moisture sensing coefficient are utilized, which is expressed as

$$G_b^{new} = G_b^{old} + Humidity \cdot \left(G_a - G_b^{old}\right) + Levy, \tag{23}$$

where current and new position of beetle are given by G_b^{old} and G_b^{new} and $Levy$ represents random vector for beetle movement, G_a refers to position of a beetle attracting another beetle, and it is formulated as

$$Levy = \frac{e}{|f|^{\frac{1}{\alpha}}} \cdot \left| \frac{\Gamma\left(1+\alpha\right) \cdot \sin\left(\frac{\pi}{2}\alpha\right)}{\Gamma\left(\frac{1+\alpha}{2}\right) \cdot \alpha \cdot 2^{\frac{\alpha-1}{2}}} \right|^{\frac{1}{\alpha}}, \tag{24}$$

where e and f symbolize two single random vectors that alter in range (0, 1), and α represents a constant which is equal to 1.5.

Assume $G_b^{new} = G_b(\vartheta + 1)$, $G_b^{old} = G_b(\vartheta)$ and $Humidity = J$,

$$G_b\left(\vartheta+1\right) = G_b\left(\vartheta\right) + J \cdot \left(G_a - G_b\left(\vartheta\right)\right) + Levy, \tag{25}$$

$$G_b\left(\vartheta+1\right) = G_b\left(\vartheta\right) + JG_a - JG_b\left(\vartheta\right) + Levy, \tag{26}$$

$$G_b\left(\vartheta+1\right) = G_b\left(\vartheta\right)\left(1-J\right) + JG_a + Levy. \tag{27}$$

From gradient descent algorithm [21], the update equation is given by

$$G_b\left(\vartheta+1\right) = G_b\left(\vartheta\right) - \mu\nabla f\left(G_b\left(\vartheta\right)\right), \tag{28}$$

where $G_b(\vartheta + 1)$ refers to new position of solution, and $G_b(\vartheta)$ is current solution, μ signifies parameter, and $\nabla f(G_b(\vartheta))$ is gradient function of current solution.

$$G_b\left(\vartheta\right) = \mu\nabla f\left(G_b\left(\vartheta\right)\right) + G_b\left(\vartheta+1\right) \tag{29}$$

Substitute equation (29) in equation (27)

$$G_b(\vartheta+1) = \left[\mu\nabla f\left(G_b(\vartheta)\right) + G_b(\vartheta+1)\right](1-J) + JG_a + Levy, \tag{30}$$

$$G_b(\vartheta+1) - G_b(\vartheta+1)(1-J) = \mu\nabla f\left(G_b(\vartheta)\right)(1-J) + JG_a + Levy, \tag{31}$$

$$G_b(\vartheta+1)\left[1-(1-J)\right] = \mu\nabla f\left(G_b(\vartheta)\right)(1-J) + JG_a + Levy. \tag{32}$$

Thus, the final update expression of GNBO is manifested as

$$G_b(\vartheta+1) = \frac{1}{J}\left[\mu\nabla f\left(G_b(\vartheta)\right)(1-J) + JG_a + Levy\right]. \tag{33}$$

Step 5) Move to wet mass

The space that poses an elevated chance of discovering water and can be in regions of optimality and gravity is formulated as

$$G_a^{new} = G_a^{old} + rand \cdot \left(G^* - \overline{G}\right) + Levy, \tag{34}$$

where G^* refers a position that has high moisture and $\overline{G}$ is position of water gravity.

The water gravity position is formulated as

$$\overline{G} = \frac{\sum_{a=1}^{K} G_a}{K} = \frac{G_1 + G_2 + \cdots + G_K}{K} \tag{35}$$

$$G_a^{new} = G_a^{old} + rand \cdot \left(\varepsilon \cdot G^* - (1-\varepsilon)\overline{G}\right) + Levy. \tag{36}$$

Step 6) Hunting or population removal

The beetle tries to return to its nest after being watered, but here some of them are hunted by lizards. Thus, it is essential to present a possibility for beetle removal. Thus, the roulette wheel mechanism is utilized for removing the solution. Here, the improper solution is more susceptible to removal and hunting. Table 3 depicts the pseudocode of the proposed GNBO.

Step 7) Re-evaluation of fitness

The fitness is re-evaluated to obtain a better solution.

Step 8) Termination

The steps are repeated until the maximum count of iterations is reached to acquire the best solution.

Table 3. Pseudocode of proposed GNBO

```
Input: Beetle population G
Output: Best beetle $G_a^{new}$
Begin
Initialize
  Fix basic attributes such as $Q$, $K$, $Humidity$
  Each solution is termed as Namib beetle
  Initialize population with arbitrary position
for $a = 1$ to $q$ do
for $a = 1$ to $B$ do
       Update using equation (16)
     End for
   End for
  Evaluate using fitness
for $a = 1$ to $q$ do
    Evaluate using fitness
  End for
while ($\vartheta <= \vartheta_{\max}$) do
     Compute coefficient $\varepsilon$ using equation (22)
     for $a = 1$ to $q$ do
        Compute $F_a$ using equation (17)
     End for
 for $a = 1$ to $q$ do
for $j = 1$ to $F_a$ do
             Search around $G_a$
End for
End for
  for $a = 1$ to $q$ do
for $j = 1$ to $q$ do
            Evaluate distance using equation (20)
            Evaluate humidity using equation (21)
            Evaluate new beetle position using equation (33)
End for
End for
for $a = 1$ to $q$ do
            Evaluate new beetle position using equation (36)
End for
$\vartheta = \vartheta + 1$
End while
Return $G_a^{new}$
End
```

5. Performance evaluation

The propensity of GNBO is revealed by examining the techniques using certain evaluation metrics by altering task size.

5.1 *Experimental set-up*

The operation of GNBO has been performed on the Windows 10 OS with 2 GB of RAM and an Intel Core processor and is scripted in Python. Here, four set-ups are considered, wherein the first set-up includes PM = 5, VM = 10, and container = 15, the second set-up includes PM = 10, VM = 15, and container = 20, the third set-up includes PM

= 15, VM = 20, and container = 25, and the fourth set-up includes PM = 20, VM = 25, and container = 30. The resources allocated to a VM typically include CPU, memory, bandwidth, and frequency. The RAM of VMs varies from 2 GB to 4 GB.

5.2 *Dataset description*

The HCSP-based dataset [26], which consists of cloudlets and VMs, could be utilized to schedule tasks in cloud computing. The dataset consists of four dataset instances: c-hilo, c-lohi, i-hilo, and i-lohi. There are 1,024 cloudlets and 32 VMs per dataset. Any simulation can use these datasets as input by directing the files to a certain location.

5.3 *Evaluation measures*

The evaluation of GNBO is performed with certain measures and is defined below. The migration cost is measured in joules, energy is measured in kWh, and load is the number of cloudlets, where each cloudlet is of length 1,000 MIPS.

5.3.1 *Load*

The load used here is number of tasks executed, which is measured in MIPS. A load of PM and VM is computed using ACNN, which is briefly explained in Section 4.2.

5.3.2 *Migration cost*

Migration cost is the time required to migrate the containers. The migration cost is described in equation (7). It is measured in joules (J).

5.3.3 *Energy consumption*

Energy consumption is the energy consumed by the server. The energy consumption of PM and VM is described in equations (13) and (8). It is measured in watts (W).

5.4 *Comparative methods*

The proposed GNBO is analyzed with prior methods that involve Linux container migration (LXD/CR-CM) [26], CMDM [15], energy-efficient container migration (CM) [11], and migration technique [9].

5.5 *Comparative analysis*

The assessment of techniques is done using four set-ups, considering migration cost, energy consumption, and load by altering task size (Figures 6 to 9).

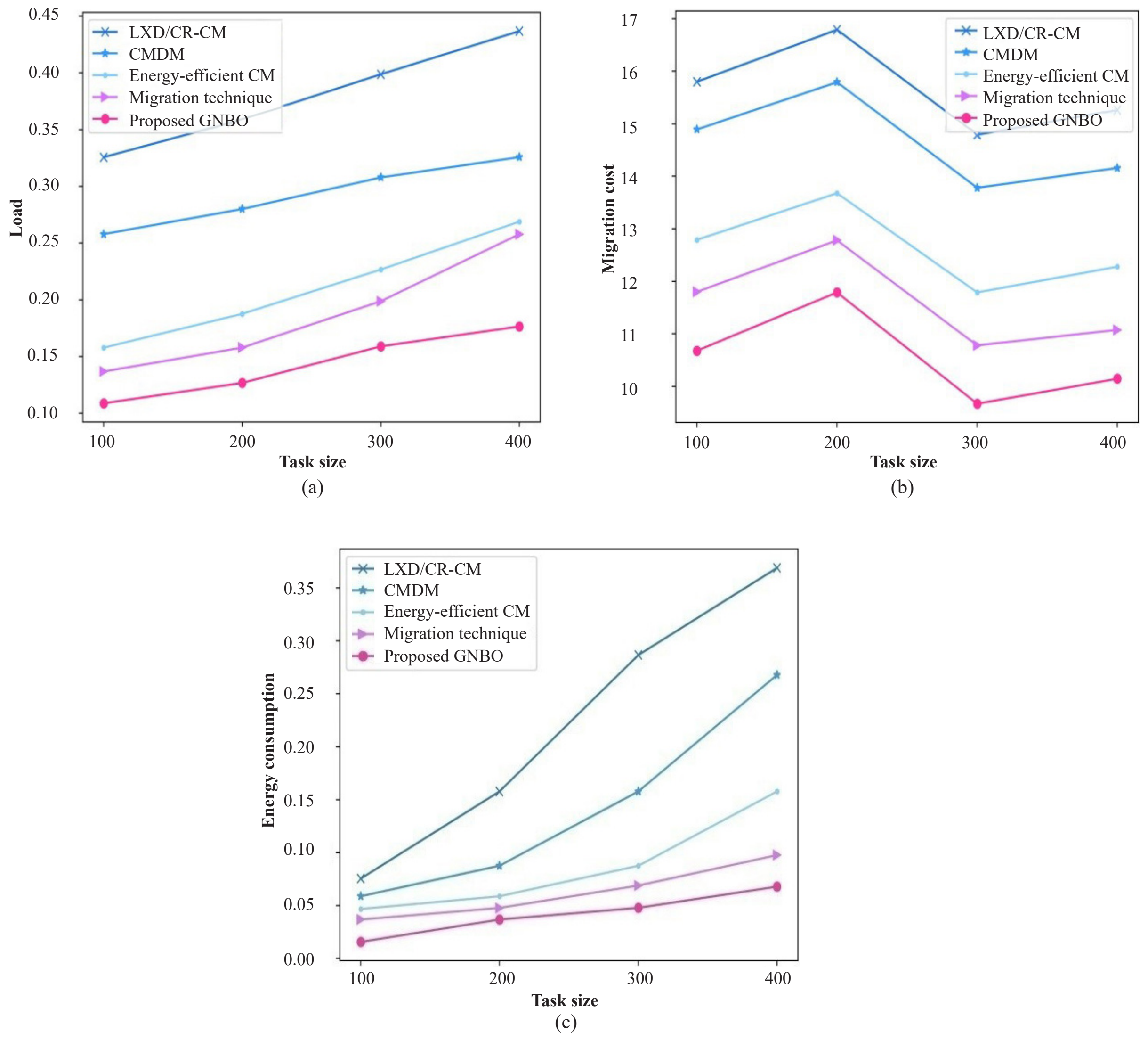
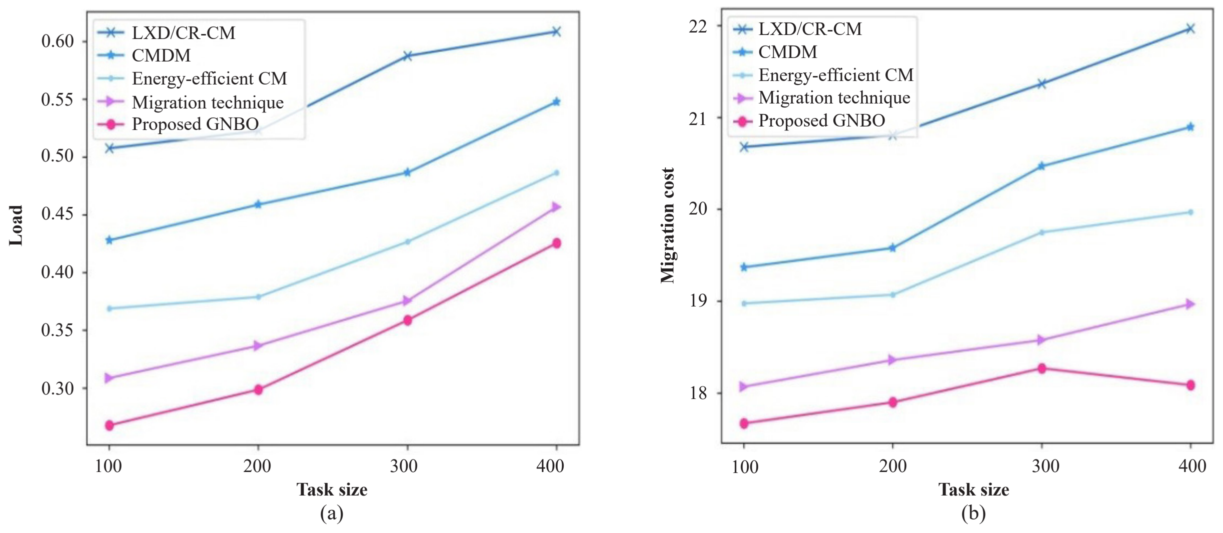
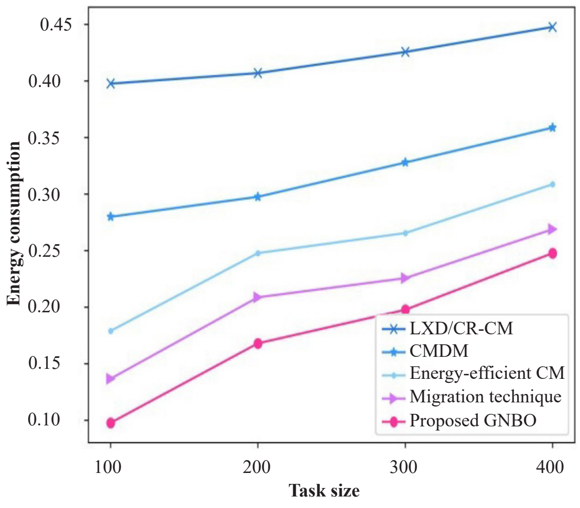


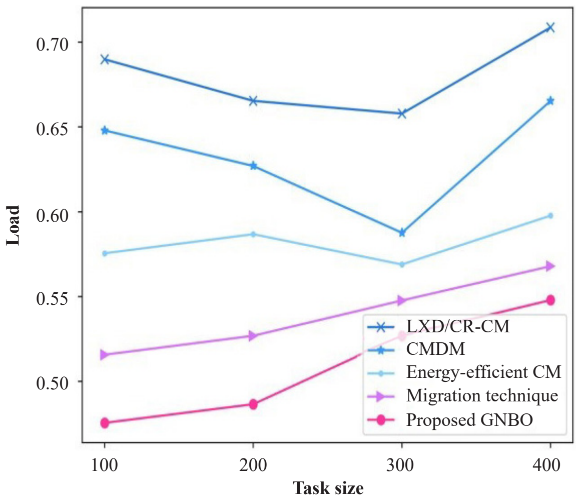
Figure 6. Technique assessment with set-up = 1 considering (a) load, (b) migration cost, and (c) energy consumption



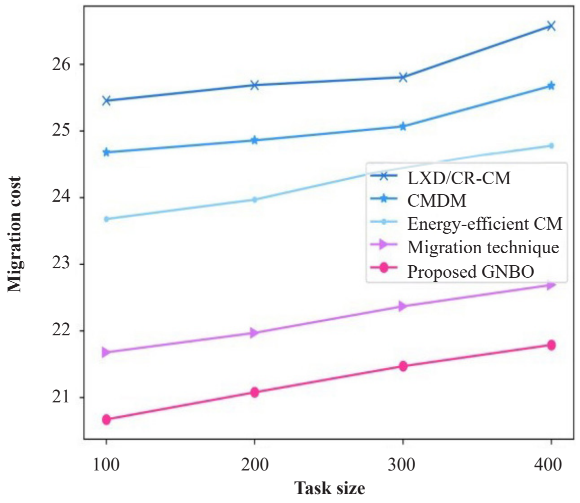


(c)

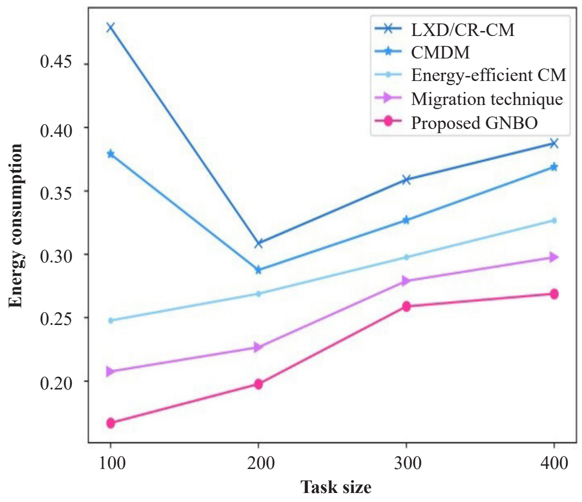
Figure 7. Technique assessment with set-up = 2 considering (a) load, (b) migration cost, and (c) energy consumption



(a)

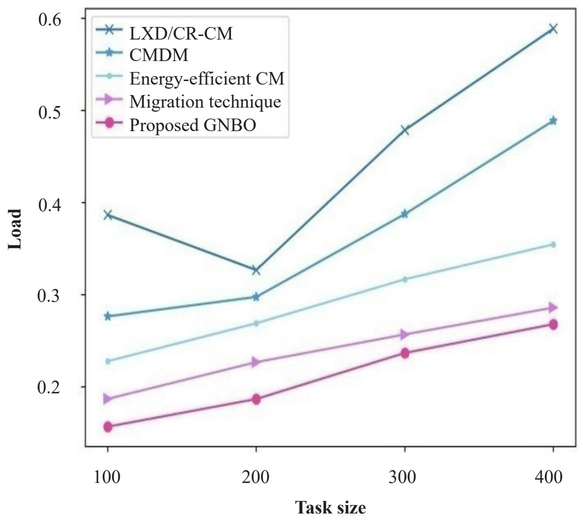


(b)

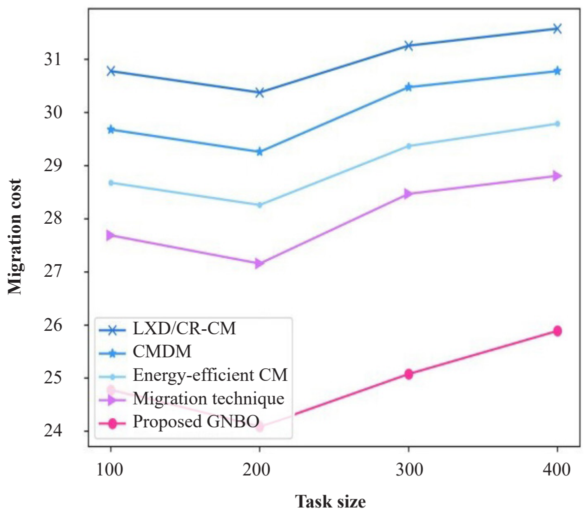


(c)

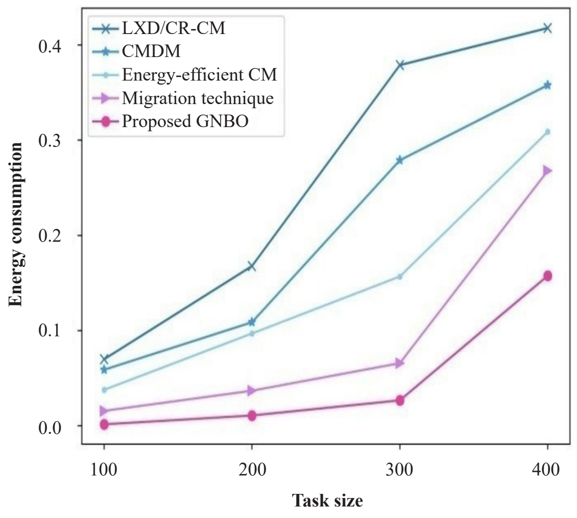
Figure 8. Technique assessment with set-up = 3 considering (a) load, (b) migration cost, and (c) energy consumption



(a)



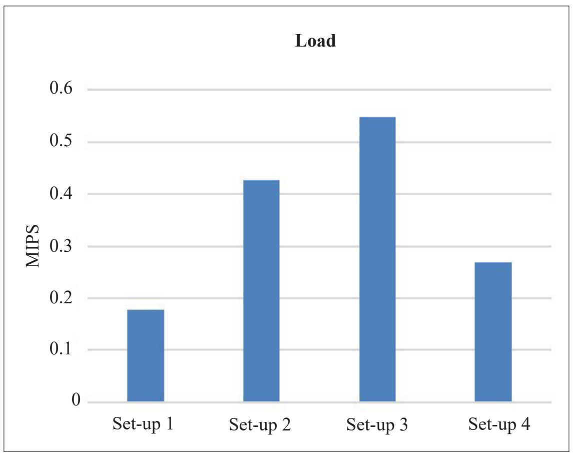
(b)



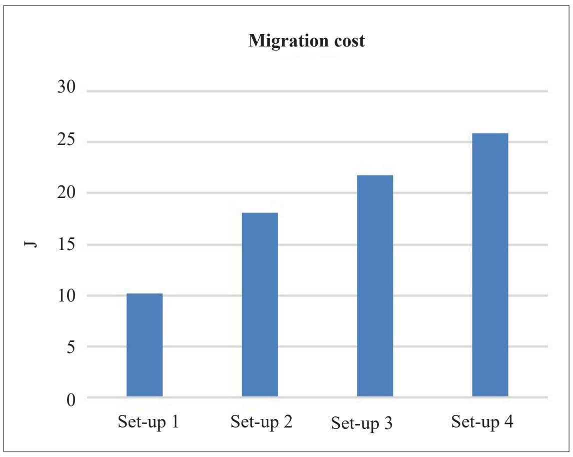
(c)

Figure 9. Technique assessment with set-up = 4 considering (a) load, (b) migration cost, and (c) energy consumption

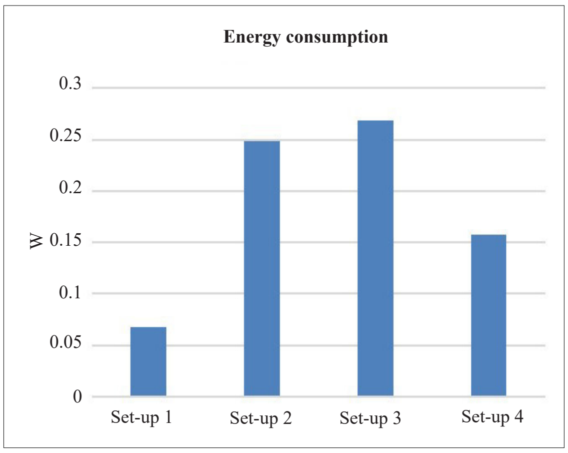
In Figure 10, GNBO's behavior is compared for four different configurations. Figure 10(c) demonstrates that the energy consumption of GNBO is directly proportional to its load, since load balancing is crucial for achieving energy efficiency in a cloud environment. In this situation, if the load is not evenly distributed, some servers will always be busy, whereas others will be idle or underloaded, which will result in improper energy consumption. Further migration costs increase with an increase in the number of PMs, VMs, and containers (as can be concluded from Figure 10(b)).



(a)



(b)



(c)

Figure 10. Assessment of proposed GNBO (a) load, (b) migration cost, and (c) energy consumption

5.6 *Comparative discussion*

This section provides the collective analysis from assessment set-up = 1 to assessment set-up = 4 and is summarized in Table 4. A performance analysis of GNBO with algorithms LXD/CR-CM, CMDM, energy efficiency, and migration technique is done using metrics: load, migration cost, and energy consumption, with task variation from 100 to 400 in each set-up. It can be observed from the assessments that GNBO outperforms in migration cost, load, and energy efficiency. As the number of tasks increases in all four set-ups, the average reduced value of GNBO for load is 29.7%, which improves workload allocation. The average reduction value of migration cost by 15.54% helps in reducing the downtime of the migrating container and hence improves SLA. A reduction of average energy consumption by 35.5% of GNBO optimizes the energy consumption of the server or the PM.

Table 4. Comparative analysis of proposed GNBO with existing system

Set-up	Metrics	LXD/CR-CM	CMDM	Energy efficient CM	Migration technique	Proposed GNBO
Set-up 1	Load	0.437	0.326	0.269	0.258	0.177
	Migration cost	15.257	14.157	12.278	11.076	10.146
	Energy consumption	0.369	0.268	0.158	0.098	0.068
Set-up 2	Load	0.609	0.548	0.487	0.457	0.426
	Migration cost	21.970	20.897	19.970	18.970	18.087
	Energy consumption	0.448	0.359	0.309	0.269	0.248
Set-up 3	Load	0.709	0.665	0.598	0.568	0.548
	Migration cost	26.580	25.680	24.780	22.690	21.790
	Energy consumption	0.388	0.369	0.327	0.298	0.269
Set-up 4	Load	0.589	0.489	0.355	0.286	0.268
	Migration cost	31.580	30.780	29.790	28.809	25.890
	Energy consumption	0.418	0.358	0.309	0.268	0.158

6. Conclusion

Due to the huge augmentation in cloud services, the migration of containers can help reduce huge performance bottlenecks. Hence, it helps in optimizing energy consumption, migration costs, and load reduction. Here, the migration is dependent on the competence of the data center. Cloud providers are considered in a quest for a solution that can address these overheads and provide adaptability to cloud platforms. Here, the cloud simulation is executed using the PM, VM, and containers, and then tasks are allocated to the VM in a round-robin manner. A load of PM is predicted by utilizing ACNN. If the load is higher than the threshold value, then the containers are optimally migrated from the overloaded PM to the underloaded PM using GNBO by considering the multi-objectives. It provides a huge enhancement in the migration process, and it aids and serves as a key migration to improve the efficiency of the CDC. The proposed GNBO provided enhanced efficiency with the smallest load of 0.177 MIPS, a migration cost of 10.146 J, and an energy consumption of 0.068 W. Other optimization models may be adapted in the future to expose the feasibility of models in real-time. It is possible to use the proposed GNBO to optimize energy consumption while taking into account more parameters, such as network bandwidth and the data sets of other data centers.

Conflict of interest

The authors declare no conflict of interest.

References

[1] Chard K, Caton S, Rana O, Bubendorfer K. Social cloud: Cloud computing in social networks. *Proceedings of*

2010 IEEE 3rd International Conference on Cloud Computing. IEEE; 2010. p.99-106.

[2] Kim W. Cloud computing: Today and tomorrow. *Journal of Object Technology*. 2009; 8(1): 65-72.

[3] Qaqos NN, Zeebaree SRM, Hussan BKH. Opnet based performance analysis and comparison among different physical network topologies. *Academic Journal of Nawroz University*. 2018; 7(3): 48-54.

[4] Ahmed OM, Haji LM, Shukur HM, Zebari RR, Abas SM, Sadeeq MAM. Comparison among cloud technologies and cloud performance. *Journal of Applied Science and Technology Trends*. 2020; 1(2): 40-47.

[5] Ahmed OM, Abduallah WM. A review on recent steganography techniques in cloud computing. *Academic Journal of Nawroz University*. 2017; 6(3): 106-111.

[6] Zebari RR, Zeebaree SRM, Jacksi K, Shukur HM. E-business requirements for flexibility and implementation enterprise system: A review. *International Journal of Scientific and Technology Research*. 2019; 8(11): 655-660.

[7] Rashid ZN, Zebari SR, Sharif KH, Jacksi K. Distributed cloud computing and distributed parallel computing: A review. *Proceedings of 2018 International Conference on Advanced Science and Engineering (ICOASE).* IEEE; 2018. p.167-172.

[8] Zeebaree SR, Zebari RR, Jacksi K, Hasan DA. Security approaches for integrated enterprise systems performance: A review. *International Journal of Scientific and Technology Reserach*. 2019; 8(12): 2485-2489.

[9] Singh G, Singh P. A container migration technique to minimize the network overhead with reusable memory state. *International Journal of Computer Networks and Applications*. 2022; 9(3): 350-360.

[10] Malhotra L, Agarwal D, Jaiswal A. Virtualization in cloud computing. *Journal of Information Technology and Software Engineering*. 2014; 4(2): 1-3.

[11] Chhikara P, Tekchandani R, Kumar N, Obaidat MS. An efficient container management scheme for resource-constrained intelligent IoT devices. *IEEE Internet of Things Journal*. 2020; 8(16): 12597-12609.

[12] Watada J, Roy A, Kadikar R, Pham H, Xu B. Emerging trends, techniques and open issues of containerization: A review. *IEEE Access*. 2019; 7: 152443-152472.

[13] Saboor A, Hassan MF, Akbar R, Shah SNM, Hassan F, Magsi SA, et al. Containerized microservices orchestration and provisioning in cloud computing: A conceptual framework and future perspectives. *Applied Sciences*. 2022; 12(12): 5793.

[14] Farahnakian F, Pahikkala T, Liljeberg P, Plosila J, Tenhunen H. Utilization prediction aware VM consolidation approach for green cloud computing. *Proceedings of 2015 IEEE 8th International Conference on Cloud Computing*; New York, NY, USA: IEEE; 2015. p.381-388.

[15] Ma Z, Shao S, Guo S, Wang Z, Qi F, Xiong AO. Container migration mechanism for load balancing in edge network under power Internet of Things. *IEEE Access*. 2020; 8: 118405-118416.

[16] Al-Tarawneh MA. Mobility-aware container migration in cloudlet-enabled IoT systems using integrated multicriteria decision making. *International Journal of Advanced Computer Science and Applications*. 2020; 11(9): 239-246.

[17] He TZ, Toosi AN, Buyya R. Efficient large-scale multiple migration planning and scheduling in SDN-enabled edge computing. *arXiv preprint arXiv:2111.08936*. 2021.

[18] Li L, Dong J, Zuo D, Wu J. SLA-aware and energy-efficient VM consolidation in cloud data centers using robust linear regression prediction model. *IEEE Access*. 2019; 7: 9490-9500.

[19] Benomar Z, Longo F, Merlino G, Puliafito A. Cloud-based enabling mechanisms for container deployment and migration at the network edge. *ACM Transactions on Internet Technology (TOIT)*. 2020; 20(3): 1-28.

[20] Chahardoli M, Eraghi NO, Nazari S. Namib beetle optimization algorithm: a new meta-heuristic method for feature selection and dimension reduction. *Concurrency and Computation: Practice and Experience*. 2022; 34(1): e6524.

[21] Zhao D, Wang B, Liu D. A supervised actor-critic approach for adaptive cruise control. *Soft Computing*. 2013; 17(11): 2089-2099.

[22] Khan AA, Zakarya M, Khan R, Rahman IU, Khan M. An energy, performance efficient resource consolidation scheme for heterogeneous cloud datacenters. *Journal of Network and Computer Applications*. 2020; 150: 102497.

[23] Liu F, Ma Z, Wang B, Lin W. A virtual machine consolidation algorithm based on ant colony system and extreme learning machine for cloud data center. *IEEE Access*. 2019; 8: 53-67.

[24] Gholipour N, Arianyan E, Buyya R. A novel energy-aware resource management technique using joint VM and container consolidation approach for green computing in cloud data centers. *Simulation Modelling Practice and Theory*. 2020; 104: 102127.

[25] Ruder S. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*. 2016.

[26] Bhardwaj A, Krishna CR. A container-based technique to improve virtual machine migration in cloud computing. *IETE Journal of Research*. 2022; 68(1): 401-416.

[27] IEEE DataPort. *Dataset for task scheduling in cloud using cloudsim*. Available from: https://ieee-dataport.org/

documents/dataset-task-scheduling-cloud-using-cloudsim#files [Accessed 6th August 2022].

[28] Choudhary A, Govil MC, Singh G, Awasthi LK, Pilli ES, Kapil D. A critical survey of live virtual machine migration techniques. *Journal of Cloud Computing*. 2017; 6: 23. Available from: https://doi.org/10.1186/s13677-017-0092-1.

[29] Soma S, Rukmini S. Virtual machine and container live migration algorithms for energy optimization of data centre in cloud environment: A research review. In: Joby PP, Balas VE, Palanisamy R. (eds.) *IoT Based Control Networks and Intelligent Systems. Lecture Notes in Networks and Systems, vol 528*. Springer, Singapore; 2023. Available from: https://doi.org/10.1007/978-981-19-5845-8_45.

[30] Liu JS, Lin CH, Hu YC, Donta PK. Joint data transmission and energy harvesting for MISO downlink transmission coordination in wireless IoT networks. *Sensors*. 2023; 23(8): 3900. Available from: https://doi.org/10.3390/s23083900.

[31] Kumar DP, Amgoth T, Annavarapu CSR. Machine learning algorithms for wireless sensor networks: A survey. *Information Fusion*. 2019; 49: 1-25. Available from: https://doi.org/10.1016/j.inffus.2018.09.013.

[32] Ramesh B, Suhashini G, Kalnoor BVVN, Rao DN. Cost optimization by integrating PV-system and battery energy storage system into microgrid using particle swarm optimization. *International Journal of Pure and Applied Mathematics*. 2017; 114(8): 45-55.

[33] Samuilik I, Sadyrbaev F, Ogorelova D. Comparative analysis of models of gene and neural networks. *Contemporary Mathematics*. 2023; 4(2): 217-229. Available from: https://ojs.wiserpub.com/indexphp/CM/article/view/2404.